Windows 2000 • Microsoft Network Monitor
Policy sablon-fájlok szerkesztése
BackOffice • Biztonságos Exchange levelezés
Business Internet • Component Load Balancing
Dupla KV
Developer • Transact SQL
Összetett lekérdezések

tech.net

A Microsoft Magyarország szakmai magazinja

Kellemes Ünnepeket és
Boldog Új Évet Kívánunk
Minden Kedves Olvasónknak!

2000.12.

I. Évfolyam 4.szám / Ár: 899 Ft

KÖSZÖNTŐ
Kovács Barbara training manager

Kedves olvasóink!

Mint majd az újság átlapozásakor észre fogják venni, került e decemberi számba egy-két olyan oldal, amely nem annyira a szoftverek használatára, mint inkább a karácsony utáni ünnepekre, a szilveszterre és az újévre „készít fel". A Dupla KV mellett ebben a hónapban átmenetileg Dupla Pezsgő rovat is van.

Nem feledve lapunk küldetését, e hónapban is sok szakmai érdekességről olvashatnak, hisz célunk, hogy a magyarországi IT szakemberek ismeretei továbbra is összemérhetőek legyenek más, fejlettebb országok mérnökeivel. Ennek a feladatnak ellátására mi magunk kicsik vagyunk, nem véletlen, hogy a Microsoft nagy hangsúlyt helyez Hivatalos Oktatóközpontjai *(MSCTEC)* által tartott tanfolyamainak színvonalára. Az utóbbi időben azonban úgy látszik, valami nem stimmel ezen a tájon. Nem a Microsoft elkötelezettsége hagyott alább, nem fogytak el, sőt megszaporodtak az elsajátítandó ismeretek... Akkor mi történt?

Nem lehet egyértelműen kijelenteni, hogy Magyarország ezen piaca alvó piac lenne, de egyértelműen azt sem lehet állítani, hogy olajozottan működne a gépezet. Ugyan régen túlléptünk a tudathasadásos állapoton, egyre többet beszélünk arról, hogy milyen lényeges az emberi erőforrás, és annak fejlesztése egy szervezet, sőt egy egész ország életében. Vajon hány olyan cég van, amely örömmel küldi szakembereit egy-egy tanfolyamra? Vajon mennyi idő kell ahhoz, hogy rájöjjenek a cégek: hogyha az a személy, akit delegálnak a tréningre, a tudást nem, esetleg a tréning anyagát tudja továbbadni a munkatársainak? A megszerzett tudással rögtön dolgozni kezd, hogy kamatoztassa azt. Viszont egy lábon természetesen egy cég sem áll meg, vagy csak addig a pillanatig, amíg az a bizonyos láb ott tartózkodik.

Egyértelműen kimutatható, hogy hazai szakembereinknek legjobban az éri meg anyagilag, ha itthon képzik magukat. Nagy a különbség nyugat és kelet Európa árai között, viszont az információ, a tanfolyam magja mindenhol ugyanaz. Egy tréning színvonalát alapvetően az oktató személyisége határozza meg. Egy jó oktató előadói képessége nemcsak az adott anyag mély ismeretén alapszik, a teljes tudásbázisra betekintése kell legyen, ismernie kell az új trendeket, a várható fejlesztéseket.

A jó oktatót azonban a honlapokon található információk alapján nem könnyű kiszűrni, hiszen nem találhatóak meg ott a tanfolyami értékelőlapok eredményei. Még ha ezek az adatok elérhetőek is lennének, akkor sem lenne garancia arra, hogy mindenkit a saját igényeinek megfelelő oktató fog képezni. A tanfolyamon résztvevő saját tanfolyamélménye, tapasztalata fogja meghatározni, hogy mely oktatótól tanul a leghatékonyabban. Talán nem túlzás azt állítani, hogy a rendszeresen tanfolyamot látogató szakemberek ezért lehetnek lépéselőnyben, hiszen ők már tapasztalták, hogy mely oktatóktól kapják a tudást a nekik megfelelő színvonalon, és hatékonysággal.

Egyébként a számok tükrében döbbenetesek a különbségek Nyugat- és Kelet-Európa között az oktatási piacon. Nem áll módomban összevetni a kelet és nyugati államok társadalmi, gazdasági és politikai mutatóit, de egyértelmű különbségeket a számok, a tények alapján is felfedezhetünk anélkül, hogy mélyebbre ásnánk. Óriási különbség van Nyugat- és Kelet-Európa között, ha az oktatóközpontok számát tekintjük. Nyugat-Európában a Microsoft tanfolyamok több mint 400 oktatóközpontja működik *(az élen Németország áll 134 oktatóközponttal, utána szorosan az Egyesült Királyság)*. A népességre vetített oktatóközpontszám több mint duplája a hazainak. Ennek ellenére több céget találunk, melyek – valószínűleg a nagy kereslet miatt – több telephellyel is rendelkeznek az adott ország különböző pontjain. Ez térségünkre egyáltalán nem jellemző. A kelet-európai oktatóközpontok összesített száma nem éri el a 150-et sem! *(A kelet- és közép-európai térség élén Lengyelország áll a maga 28 oktatóközpontjával.)*

Tény az is, hogy ha áttekintünk egy kicsit a tengerentúlra, felfedezhetjük, az USA-ban még ennél is rózsásabb a helyzet. Csak New York államban 106 nyilvántartott oktatóközpont működik. Képzelhetjük mennyi lehet az oktatóközpontok száma az USA egész területén!

Nagy különbségek vannak országonként a tanfolyamok árában is. Ugye mondani sem kell, hogy ebben a kategóriában is az USA vezet? Itt a tanfolyam költsége átlagosan 431 USD/nap, ami egy ötnapos tanfolyam esetén egy magyar tanulni vágyónak 668.000,-forintjába kerülne, természetesen szállás és útiköltségek nélkül! Németországban közel 500.000,- forintért, az osztrákoknál 400-500 ezer forintért ülhetnénk be egy ötnapos tanfolyamra. Ausztriától és Németországról keletebbre azonban az árak mintha laposkúszásban haladnának. Még a térségünkben fejlettnek mondható lengyelek sem adják napi 123 USD-nél drágábban a tanfolyamaikat. Magyarországon a tanfolyamok 90 USD körül mozognak naponként.

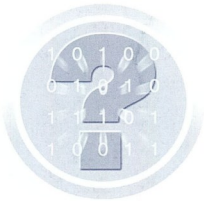
Megnyugtatásképpen azért hozzáfűzném, hogy ezek a számok nem mutatják az oktatás minőségének színvonalát. Nem jelenti, hogy a mi tanfolyamaink nem vagyunk elmaradva. Európa minden része tartja a lépést a Microsoft iramával. A curriculumban felsorolt közel 140 tanfolyam felöleli a teljes kínálatot a Windows NT 4.0-tól a Windows 2000-ig, az Exchange 2000-től az SQL tanfolyamokig. A fenti listából választhatják ki az oktatóközpontok az erőforrásuknak és profiluknak megfelelő kínálatukat. Dicséretes, hogy a magyarországi oktatóközpontok is képesek megújulni, hiszen az innováció, a folyamatos lépéstartás, az erőforrások fejlesztése ezekben a cégekben éppúgy zajlik, mint nyugaton.

Tehát leszögezhetjük, hogy nincs baj a tanfolyami árakkal, ha csak az nem, hogy túl alacsonyak. Nincs baj a kínált tanfolyamok változatosságával és az oktatás minőségével sem. Nincs túl sok oktatóközpont sem a magyar piacon.

Csak nem az a baj, hogy nem akarunk tanulni!?

Kovács Barbara
Training manager
kovacs.barbara@netacademia.net

Microsoft



tech.net

A Microsoft Magyarország Szakmai Magazinja

Szerkesztőség
Főszerkesztő: **Fóti Marcell**
marcellf@netacademia.net
Főszerkesztő-helyettes: **Fülöp Miklós**
mick@netacademia.net
Szerkesztőség címe:
1105 Budapest, Ihász utca 13.
Tel.: 263-2732
technet@netacademia.net
Nyilvános levelezési lista:
technet@lyris.netacademia.net

Kiadja a **Microsoft Magyarország**
1031 Budapest, Graphisoft park 3.
Tel.:437-2800
A kiadásért felel:
Arany Tóth László
v-laarto@microsoft.com

Terjeszti a **NetAcademia Kft.**
Terjesztési, előfizetési információ:
Tel.: 263-2732
terjesztes@netacademia.net
Megjelenik havonta, ára 899 Ft

Előfizethető megrendelőlevélben a szerkesztőségnél:
1105 Budapest, Ihász utca 13.
Fax: 261-7145
http://technet.netacademia.net/subs

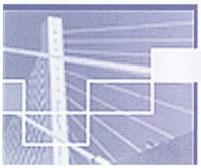
Hirdetésfelvétel:
Bársonykalapács Marketing
Felelős: **Udvarev Rita**
Tel./Fax: 214-0923
velvethammer@ahol.com
1027 Budapest, Fő utca 67. V. 1.
Grafikai tervezés, kivitelezés, nyomdai előkészítés:
Bársonykalapács Marketing
Művészeti vezető: **Balogh Zoltán**

Nyomda:
Partner's 2000 Kft.
1124 Budapest, Sion lépcső 7.
Felelős nyomdász: **Galambos Sándor**

ISSN 1586-5185

Small Business Server 2000

A .NET kiszolgálócsalád debütálása után már szinte készen van a kisvállalatoknak szánt csomag, a Small Business Server 2000 is. A **http://www.microsoft.com/sbserver/productinfo/SBS2000.htm** címen már most olvashatunk a termék újdonságairól, ami a hírek szerint 2000 vége helyett csak a jövő év elején fog megjelenni, és a Windows 2000 mellett szinte a teljes .NET palettát *(Exchange 2000, SQL Server 2000, ISA Server)* felvonultatja majd. A Windows 2000 Magazine értesülései szerint az SBS2K-ban addigra már a Service Pack 2-vel ellátott Windows 2000-et találunk.



Javítócsomagok a láthatáron

Ha már a javítócsomagoknál tartunk: megjelent az Internet Explorer 5.5 SP1 és az Office 2000 Service Pack 2. Mindkét javítócsomag elsősorban biztonsági javításokat tartalmaz. Az Office 2000 Service Pack 2 két változatban tölthető le: a kilenc megabájtos változat az otthoni felhasználók számára, a harminc megabájtos példány pedig vállalati környezetbe, központi telepítéshez szükséges. A **http://www.microsoft.com/windows/ie** illetve **http://www.microsoft.com/office/ork/2000/journ/OffSP2.htm** címről letölthető csomagok angol nyelvhez készültek, de mindkettőből várható a magyar nyelvű változat is.

Itt a DirectX 8!

Megjelent a Microsoft multimédia-könyvtárának legújabb változata, a DirectX 8 is. A DirectX 8 az új hardverek támogatása mellett fotóminőségű 3D animációt, fejlettebb hálózati játéktámogatást és 3D hangkezelést biztosít. A teljes fejlesztői változat, a DirectX 8 SDK a **http://msdn.microsoft.com/directx** címről kiindulva tölthető le, mindegy 145 megabájtot kóstál. A felhasználók megüsszák kevesebbel: a Windows 95/98/ME változat 11, a Windows 2000-re készült verzió pedig 8 megabájt. A felhasználói letöltéseket a **http://www.microsoft.com/directx** címről lehet elérni.

A bővítés egyelőre magyar nyelven nem érhető el, de az oldalon már most is számos nyelvi verzió található, így joggal várhatjuk a magyar változat megjelenését is. *(A tapasztalat azt mutatja, hogy az eddigi verziók esetében nem okozott különösebb gondot, ha magyar nyelvű operációs rendszerre angol nyelvű DirectX-et telepítettünk. Ha van kedvünk kockáztatni, talán érdemes megpróbálni.)*

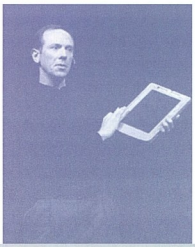
Microsoft Visual Studio .net beta 1

Letölthető a Visual Studio.NET és a Whistler Beta 1

A http://msdn.microsoft.com/vstudio/nextgen/beta.asp címről az MSDN előfizetők már letölthetik a Microsoft új generációs fejlesztőkészletének első nyilvános béta változatát. Ugyancsak itt érhetik el az MSDN Professional és MSDN Universal csomag tulajdonosai a Whistler első nyilvános bétáját is. Szintén ehhez a hírhez tartozik, hogy a Microsoft az új C# programozási nyelvet és a .NET keretrendszer egy részét felterjesztette szabványosításra az ECMA *(European Computer Manufacturer's Association)* szervezethez. Ugyanez a szervezet végezte a JavaScript *(nem a Java)* szabványosítását is.

Mi az a TabletPC?

A COMDEX alkalmával, Las Vegasban mutatta be Bill Gates a TransMeta ***(http: // www.transmeta.com)*** Crusoe mobil processzorára alapuló, Windows-t futtató TabletPC prototípusát. A gépen már a bemutatón Whistler futott, és minden bizonnyal ez az igazi várományosa az új platformnak. „A TabletPC egyesíti a számítógép a papír és ceruza előnyeit" – mondja a szlogen. A felhasználóinak nem kell kompromisszumokat kötni: ez egy teljesértékű számítógép. Az első TabletPC megjelenése 2002 végére, talán 2003-ra várható.



Bert Keely rendszermérnök és a TabletPC prototípusa

A Microsoft bejelentette a FrontPage 10-et

A FrontPage webszerkesztő-eszköz új változata hű marad a fejlesztések irányához: az új változat többek között fejlettebb képkezelést, gazdagabb galériákat, automatikus webes eszközöket *(MSN keresőt, Expedia térképet)*, rendszerfelügyeleti összetevőket *(használati analízis, toplisták)* és teljes csoportmunka-támogatást tartalmaz majd. A FrontPage 10 jelenleg Beta 2 állapotban tart, és az Office 10-zel karöltve valószínűleg jövő év közepén jelenik majd meg. Az új FrontPage szolgáltatásairól addig is a **http://www.microsoft.com/presspass/features/2000/nov00/11-28frontpage.asp** címen olvashatunk.

Microsoft Certified Partner program

Mint azt már novemberi számunkban is említettük, januártól megújul a Microsoft partneri programja *(a volt MCSP program)*. A Microsoft Certified Partner program két MCP minősítésű, a Microsoft Gold Certified Partner pedig specializálódástól függően *(elektronikus kereskedelem, vállalati rendszerek, Internet- és alkalmazás-szolgáltatók, ügyféltámogatás)* 4-7 MCSE rendszermérnök munkatárs bejegyzését és természetesen az éves díj befizetését tűzi feltételként a belépéshez. A regisztráció máris megkezdődött, a partneri program további részleteiről a Microsoft Magyarország weboldalán, a **http://www.microsoft.com/hun/partner/certpartner.htm** címen tájékozódhatunk.

Elkészült a Microsoft Szerviz CD 7

Megjelent a Microsoft Magyarország negyedévi rendszerességgel kiadásra kerülő szervízcsomag-sorozatának 7. tagja. A dupla CD-n a Microsoft Tudásbázis anyagai mellett megtalálhatók az általánosan használt Microsoft operációs rendszerek és irodai alkalmazások legfrissebb javítócsomagjai, valamint egyik legnépszerűbb játékának, az Age Of Kings-nek demo változata is. A csomag egy példányban ingyenesen rendelhető a Microsoft Ügyfélszolgálattól, a további példányok ára 1000 Ft.

Helyreigazítás

Novemberi számunk 31. oldalán, az NTLMv2-t tárgyaló cikkünkbe sajnálatos hiba csúszott. A hiba a Windows 9x registry kulcsának nevében volt, pontosan az az információ maradt ki a cikkből, amiért az tulajdonképpen íródott. A lényeg az, hogy a Windows 9x regisztrációs adatbázisában található, az NTLMv2-t engedélyező érték neve **LMCompatibilityLevel** helyett helyesen: **LMCompatibility**.

A telepítés után a Windows9x kompatibilitási okok miatt még az eredeti LM azonosítást használja, de a következő registry értékek módosításával le lehet erről beszélni:

```
HKLM\System\CurrentControlSet\control\LSA
Value Name: LMCompatibility
Data Type:     REG_DWORD
Value: 3
Valid Range: 0-3
```

A DNS ügyfelek beállítása

A Windows 2000 DNS ügyféloldalán általában kevés beállítani valónk akad, ami az esetek többségében kimerül egy elsődleges *(és esetleg egy másodlagos)* DNS kiszolgáló IP címének megadásában. Szerencsére ezt a feladatot is rábízhatjuk a DHCP kiszolgálóra.

Az ügyféloldali alapbeállítások rendszerint megfelelők, de előfordulhatnak olyan helyzetek, amelyekben ezeket célszerű megváltoztatni. Az alábbi rendszerleíró azonosítók ebben lesznek majd segítségünkre.

Az alapértelmezett TTL beállítása

A DNS ügyfelek az A és PTR rekordok TTL-jét alaphelyzetben 20 percre állítják be, amit megváltoztathatunk a következő rendszerleíró azonosítóval:

DefaultRegistrationTTL
Az azonosító helye:
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters
Adattípus: **REG_DWORD** *(másodpercben)*
Alapértelmezett érték: **0x4B0** *(decimális értéke 1200 másodpercnek, vagyis 20 percnek felel meg)*
Értelmezési tartomány: **0–0xFFFFFFFF**
(Alapértelmezésben nem létező azonosító)

A dinamikus frissítés kikapcsolása

Az automatikus frissítés látszólag egyértelmű hasznossága ellenére néha nemkívánatos is lehet. A következő rendszerleíró azonosítóval a dinamikus frissítés akár a teljes számítógépre, akár egyetlen kártyára is kiiktatható:

DisableDynamicUpdate
Az azonosító helye:
KEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services Tcpip\Parameters
vagy
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services Tcpip\Parameters\Interfaces\<interface>
Adattípus: **REG_DWORD** *(logikai)*
Értelmezési tartomány: **0** *(hamis)*, **1** *(igaz)*
Alapértelmezett érték: **0** *(hamis: dinamikus frissítés engedélyezve)*
(Alapértelmezésben nem létező azonosító)

Lekérdezés

Az ügyfélprogram egy tartománynevet tartalmazó lekérdezést küld a DNS kiszolgálónak, amelynek hatására az a nevet megpróbálja megtalálni és a hozzá tartozó erőforrásrekordokat visszaküldeni. A lekérdezésben a keresett tartomány nevén kívül a visszaküldendő rekordtípusok kódja is megtalálható.

Az alábbi Network Monitor *(NM)* napló bemutatja egy tartománynév lekérdezését. *(A sors akarta így, hogy bemelegítő NetMon cikkem mellé mindjárt itt egy durva trace. Mea culpa, forró ólmot a fülembe. Mentségemre legyen mondva, hogy a cikk hátralévő részében azért mezőnként és bitenként felsorolom mindazt, ami itt jön.)*

Kérdés:

```
1 4.866998 LOCAL 3COM  884403 DNS 0x1587:
Std Qry for masina.falatrax.hu. of type Host
DNS 10.10.2.200
+ Frame: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD
Internet Protocol
+ IP: ID = 0xEEA8; Proto = UDP; Len: 61
+ UDP: Src Port: Unknown, (4715); Dst Port: DNS
(53); Length = 41 (0x29)
 DNS: 0x1587:Std Qry for masina.falatrax.hu. of
type Host Addr on class INET addr.
      DNS: Query Identifier = 5511 (0x1587)
   + DNS: DNS Flags = Query, OpCode - Std Qry,
RD Bits Set, RCode - No error
      DNS: Question Entry Count = 1 (0x1)
      DNS: Answer Entry Count = 0 (0x0)
      DNS: Name Server Count = 0 (0x0)
      DNS: Additional Records Count =
0 (0x0)
      DNS: Question Section: masina.falatrax.hu.
of type Host Addr on class INET addr.
          DNS: Question Name: masina.falatrax.hu.
          DNS: Question Type = Host Address
          DNS: Question Class = Internet address
class
```

Válasz:

```
2 4.866998 3COM  884403 LOCAL DNS 0x1587:
Std Qry Resp. for masina.falatrax.hu. of
type
10.10.2.200 DNS IP
+ Frame: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP:
DOD Internet Protocol
+ IP: ID = 0x7BAA; Proto = UDP; Len: 77
+ UDP: Src Port: DNS, (53); Dst Port:
Unknown (4715); Length = 57 (0x39)
 DNS: 0x1587:Std Qry Resp. for masina.fa-
latrax.hu. of type Host Addr on class INET
addr.
      DNS: Query Identifier = 5511 (0x1587)
   + DNS: DNS Flags = Response, OpCode -
Std Qry, AA RD RA Bits Set, RCode - No
error
      DNS: Question Entry Count = 1 (0x1)
      DNS: Answer Entry Count = 1 (0x1)
      DNS: Name Server Count = 0 (0x0)
      DNS: Additional Records Count =
0 (0x0)
      DNS: Question Section: masina.falat-
rax.hu. of type Host Addr on class INET
addr.
```

nehézségi fok:

```
          DNS: Question Name:
masina.falatrax.hu.
          DNS: Question Type = Host Address
          DNS: Question Class = Internet
address class
     DNS: Answer section: masina.falat-
rax.hu. of type Host Addr on class INET
addr.
          DNS: Resource Name: masina.falat-
rax.hu.
          DNS: Resource Type = Host Address
          DNS: Resource Class = Internet
address class
          DNS: Time To Live = 1200 (0x4B0)
          DNS: Resource Data Length = 4
(0x4)
          DNS: IP address = 10.10.2.200
```

A fenti „párbeszédben" egy ügyfél DNS lekérdezést küld a DNS kiszolgálónak, amelyre a masina.falatrax.hu-hoz tartozó összes A rekordot várja válaszként. A válaszban a kért rekord*(ok)* mellett megtaláljuk a kérdést is. Példánk tartománynevéhez csak egy A rekord tartozik, amely a 10.10.2.200 címre mutat. A fordított lekérdezés lefolyása nagyon hasonló, ezért erre nem mutatunk be példát.

Aliasok lekérdezése

A kérdező nem tudhatja előre, hogy a felhasználó által megadott tartománynév A vagy CNAME rekordra vonatkozik, ezért a CNAME átalakítását is el kell végezni. A járulékos hálózati forgalom csökkentése érdekében a DNS kiszolgáló a CNAME rekorddal együtt az összes hozzárendelt A rekordot is beleteszi a válaszba.
Az alábbi NM napló bemutatja egy alias lekérdezését:

Kérdés:

```
1          6.559432      DNS Server       DNS
Client       DNS          0x1590:Std Qry for
ns1.falatrax.hu. of type Host A DNS
10.10.2.200
+ Frame: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP:
DOD Internet Protocol
+ IP: ID = 0xEFCD; Proto = UDP; Len: 60
+ UDP: Src Port: Unknown, (4761); Dst Port:
DNS (53); Length = 40 (0x28)
 DNS: 0x1590:Std Qry for ns1.falatrax.hu.
of type Host Addr on class INET addr.
     DNS: Query Identifier = 5520 (0x1590)
   + DNS: DNS Flags = Query, OpCode - Std
Qry, RD Bits Set, RCode - No error
     DNS: Question Entry Count = 1 (0x1)
     DNS: Answer Entry Count = 0 (0x0)
     DNS: Name Server Count = 0 (0x0)
     DNS: Additional Records Count = 0
(0x0)
     DNS: Question Section: ns1.falatrax.-
hu. of type Host Addr on class INET addr.
          DNS: Question Name: ns1.falatrax.-
hu.
          DNS: Question Type = Host Address
          DNS: Question Class = Internet
address class
```

Válasz:

```
2          6.569446      DNS Client       DNS
Server       DNS          0x1590:Std Qry
Resp. for
ns1.falatrax.hu. of type  10.10.2.200     DNS
IP
+ Frame: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP:
DOD Internet Protocol
+ IP: ID = 0x807B; Proto = UDP; Len: 95
+ UDP: Src Port: DNS, (53); Dst Port: Unk-
nown (4761); Length = 75 (0x4B)
 DNS: 0x1590:Std Qry Resp. for ns1.falat-
rax.hu. of type Canonical name on class
INET addr.
     DNS: Query Identifier = 5520 (0x1590)
   + DNS: DNS Flags = Response, OpCode -
Std Qry, AA RD RA Bits Set, RCode - No er-
ror
     DNS: Question Entry Count = 1 (0x1)
     DNS: Answer Entry Count = 2 (0x2)
     DNS: Name Server Count = 0 (0x0)
     DNS: Additional Records Count = 0
(0x0)
     DNS: Question Section: ns1.falatrax.-
hu. of type Host Addr on class INET addr.
          DNS: Question Name: ns1.falatrax.-
hu.
          DNS: Question Type = Host Address
          DNS: Question Class = Internet
address class
     DNS: Answer section: ns1.falatrax.hu.
of type Canonical name on class INET
addr.(2
records present)
        + DNS: Resource Record: ns1.falatrax.hu. of
type Canonical name on class INET addr.
        + DNS: Resource Record: masina.falatrax.hu.
of type Host Addr on class INET addr.
```

A példában a DNS ügyfél lekérdezést küld a DNS kiszolgálónak, melyben arra kéri, hogy küldje el neki az ns1.falatrax.hu névhez tartozó A rekordot. A név jelen esetben a masina.falatrax.hu aliasa, így a válaszban két rekord szerepel: az egyik az ns1.falatrax.hu CNAME rekordja, amely az aliast tartalmazza, a másik a masina.falatrax.hu A rekordja, amely megadja a hozzárendelt IP címet.

A DNS dinamikus frissítése

Az RFC 2136 definiálja a DNS zónák dinamikus frissítését. A DNS ügyfelek ezzel a zónákba maguk jegyezhetnek be vagy törölhetnek erőforrásrekordokat, illetve rekordcsoportokat. A frissítési kérelmek teljesítése bizonyos feltételekhez köthető, amelyek a frissítési művelettől függetlenül határozhatók meg. A végrehajtáshoz az összes előfeltételnek teljesülnie kell. A Windows 2000 TCP/IP ügyfél és a DHCP kiszolgáló dinamikus frissítési kérelmekkel jegyezteti be az A és PTR rekordokat. A következő NM napló egy A rekord dinamikus bejegyzésének folyamatát mutatja be.

Kérés:

```
1          6.270000       DNS Client       DNS Server       DNS            0x61:Dyn Upd PRE/UPD records to CSODAGEP.falatrax.hu 10.10.1.52 195.152.236.200
+ Frame: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD Internet Protocol
+ IP: ID = 0x1082; Proto = UDP; Len: 115
+ UDP: Src Port: Unknown, (3276); Dst Port: DNS (53); Length = 95 (0x5F)
 DNS: 0x61:Dyn Upd PRE/UPD records to CSODAGEP.-falatrax.hu. of type Canonical name
      DNS: Query Identifier = 97 (0x61)
   + DNS: DNS Flags = Query, OpCode - Dyn Upd, RCode - No error
      DNS: Zone Count = 1 (0x1)
      DNS: Prerequisite Section Entry Count = 2 (0x2)
      DNS: Update Section Entry Count = 1 (0x1)
      DNS: Additional Records Count = 0 (0x0)
   + DNS: Update Zone: falatrax.hu. of type SOA on class INET addr.
   + DNS: Prerequisite: CSODAGEP.falatrax.hu. of type Canonical name on class Unknown Class(2 records present)
      DNS: Update: CSODAGEP.falatrax.hu. of type Host Addr on class INET addr.
           DNS: Resource Name: CSODAGEP.falatrax.-hu.
           DNS: Resource Type = Host Address
           DNS: Resource Class = Internet address class
           DNS: Time To Live = 1200 (0x4B0)
           DNS: Resource Data Length = 4 (0x4)
           DNS: IP address = 10.10.1.52
```

Válasz:

```
2          6.270000       DNS Server       DNS Client       DNS            0x61:Dyn Upd Resp. PRE/UPD records to CSODAGEP.ka 195.152.236.200 10.10.1.52
+ Frame: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD Internet Protocol
+ IP: ID = 0x86BD; Proto = UDP; Len: 115
+ UDP: Src Port: DNS, (53); Dst Port: Unknown (3276); Length = 95 (0x5F)
 DNS: 0x61:Dyn Upd Resp. PRE/UPD records to CSO-DAGEP.falatrax.hu. of type Canonical name
      DNS: Query Identifier = 97 (0x61)
   + DNS: DNS Flags = Response, OpCode - Dyn Upd, RCode - No error
      DNS: Zone Count = 1 (0x1)
      DNS: Prerequisite Section Entry Count = 2 (0x2)
      DNS: Update Section Entry Count = 1 (0x1)
      DNS: Additional Records Count = 0 (0x0)
   + DNS: Update Zone: falatrax.hu. of type SOA on class INET addr.
   + DNS: Prerequisite: CSODAGEP.falatrax.hu. of type Canonical name on class Unknown Class(2 records present)
      DNS: Update: CSODAGEP.falatrax.hu. of type Host Addr on class INET addr.
           DNS: Resource Name: CSODAGEP.falatrax.-hu.
           DNS: Resource Type = Host Address
           DNS: Resource Class = Internet address class
           DNS: Time To Live = 1200 (0x4B0)
           DNS: Resource Data Length = 4 (0x4)
           DNS: IP address = 10.10.1.52
```

A példában a DNS ügyfél egy dinamikus kérelmet küld a DNS kiszolgálónak a csodagep.falatrax.hu-hoz tartozó A rekord frissítésére, melynek IP címe jelenleg 10.10.1.52.

Zónaátvitel

A zónaátvitel háromféleképpen valósulhat meg:

- A hagyományos zónaátvitel során a másodlagos kiszolgáló teljes zónamásolatot kér az elsődlegestől. Leírása az RFC 1034-ben olvasható. Kész sávszélesség-pazarlás, ha a változás kicsi a teljes zónához képest.
- Az RFC 1995-ben definiált növekményes zónaátvitelhez az elsődleges zónát tároló DNS kiszolgálónak emlékeznie kell a zónasorszám növekedései között bekövetkező változásokra. A másodlagos kiszolgálónak így elég a legutóbbi frissítés óta történt változásokat kérnie.
- Az AD zónaátvitel AD replikációval valósul meg, melyben az AD zónákat az összes tartományvezérlő megkapja.

Íme egy NM napló, amely bemutatja a falatrax.hu zóna átvitelét az elsődleges kiszolgálóról a másodlagosra:

```
1 60.1765 Secondary Primary TCP ....S., len: 0, seq:3436924871-3436924871, ack
2 60.1765 Primary Secondary TCP .A..S., len: 0, seq:2396712099-2396712099, ack
3 60.1765 Secondary Primary TCP .A...., len: 0, seq:3436924872-3436924872, ack
4 60.1765 Secondary Primary DNS 0x0:Std Qry for falatrax.hu. of type Req for zn Xfer on
```

```
class INET addr.
5 60.1865 Primary   Secondary DNS 0x0:Std Qry
Resp. for falatrax.hu. of type SOA on class
INET addr.
6 60.1865 Primary Secondary DNS 0x636F:Rsrvd for
_ of type Unknown Type on class
7 60.1865 Secondary Primary TCP .A...., len: 0,
seq:3436924904-3436924904, ack
8 60.2366 Secondary Primary TCP .A...F, len: 0,
seq:3436924904-3436924904, ack
9 60.2366 Primary Secondary TCP .A...., len: 0,
seq:2396714217-2396714217, ack
10 60.2366 Primary Secondary TCP .A...F, len: 0,
seq:2396714217-2396714217, ack
```

Példánkban a másodlagos DNS kiszolgáló először felépít egy TCP kapcsolatot az elsődleges kiszolgálóval, majd zónaátvitelt kér. Az elsődleges zóna DNS kiszolgálója erre a zóna erőforrásrekordjainak átküldésével válaszol. A zónaátvitel első és utolsó eleme a SOA rekord. A folyamat végén a TCP kapcsolat megszűnik.

A nagy és/vagy dinamikus zónák növekményes átvitele hatékonyabb a hagyományosnál. Ez azonban járulékos munkával terheli a DNS kiszolgálót: jegyeznie kell a változásokat, és csak a módosított rekordokat kell átküldenie. A standard zónák alaphelyzetben ezt az átvitelt használják, ha lehet.

A következő NM naplóban a falatrax.hu zóna növekményes átvitelét követhetjük nyomon:

```
1       563.890835   LOCAL       3COM   6B15C7
DNS       0x6000:Std Qry for falatrax.hu. of
type SOA on class INET addr.
2       563.890835    3COM   6B15C7     LOCAL
DNS       0x6000:Std Qry Resp. for
falatrax.hu. of type SOA on class INET addr.
3       563.890835    LOCAL       3COM   6B15C7
DNS       0x4000:Std Qry for falatrax.hu. of
type Req for incrmntl zn Xfer on class INET
addr.
4       563.890835    3COM   6B15C7     LOCAL
DNS       0x4000:Std Qry Resp. for
falatrax.hu. of type SOA on class INET addr.
```

A fenti párbeszédet kezdeményező DNS kiszolgáló először lekéri a SOA rekordot, majd egy növekményes zónaátvitelt kér. A negyedik csomagban található válasz egyetlen UDP adatrészben elfér. Más lett volna a helyzet akkor, ha a válasz csonkolva érkezik, ekkor ugyanis a kezdeményező kiszolgáló TCP kapcsolatot épített volna fel, és ezen keresztül kérte volna a zónaátvitelt.

Az AD replikáció csak Windows 2000 tartományvezérlőkkel használható. A standard és növekményes zónaátvitel a másodlagos zónát tároló kiszolgálókra épít, amelyek az elsődleges zóna változásait letöltik *(pull)*. Az AD replikáció ezzel szemben „push" jellegű. Az AD replikáció a keveset változó zónáknál biztosítja, hogy a zónát tároló DNS kiszolgálók gyorsan frissülnek, míg a dinamikusabb zónáknál egyenletesebbé teszi a replikációs forgalmat.

DNS erőforrásrekordok

Mik az erőforrásrekordok?

Az erőforrásrekord egy DNS tartományról hordoz információt: a címrekord például meghatározza egy munkaállomás IP címét. Vannak azonban minden rekordban megtalálható adatok is:

- A tulajdonos jelzi, hogy a rekord melyik DNS tartományban található.
- A TTL azt határozza meg, hogy mennyi ideig kell a rekordot más DNS kiszolgálóknak a gyorsítótárukban tartani. Ez a mező a legtöbb rekordnál nem kötelező. A TTL mértékegysége a másodperc, és a 0 érték azt jelenti, hogy a rekordot nem kell ideiglenesen tárolni. A SOA rekordok alapértelmezett TTL-je például 1 óra, ami megakadályozza a rekord hosszú időre történő gyorsítótárazását, következésképpen a változások késleltetett terjedését.
- A rekordok osztályokba sorolhatók, amelyeket mnemonikokkal jelölünk. Az IN például azt jelzi, hogy a rekord az Internet osztályhoz tartozik. Korábban több osztály is létezett (például CH - Chaos Net), de jelenleg csak az IN-t használjuk.
- A rekord típusát feltüntető mnemonik használata kötelező. Az A mnemonik például címrekordot jelöl.
- Az erőforrás további leírása adható meg a változó hosszúságú rekord specifikus adatmezőben, melynek formátuma a rekord típusától és osztályától függ.

A Windows 2000 DNS kiszolgálóknál az összes DNS adat automatikusan létrejön vagy meghagyható az alapértelmezett beállítása. Az erőforrásrekordok részletes ismerete elsősorban azoknak lehet hasznos, akik a Windows 2000 más rendszerekkel való integrálására, illetve hibajavításra kívánják használni.

A standard DNS zónafájlok egyszerű szövegfájlok. Minden erőforrásrekord külön sorban található, és az összes fenti adatot is tartalmazza, egymástól szóközzel elválasztott szövegmezők formájában. A zónafájl minden rekordja a fent említetett adatokból áll, egyedül a rekord specifikus adatmező formátuma lehet eltérő a rekordok típusának és osztályának megfelelően.

Így néz ki egy zónafájl

A korábban már felvázolt falatrax.hu zóna adatai a következőképpen festenek:

```
;  Database file falatrax.hu.dns for falatrax.hu
zone.
;  Zone version:  22508
;
@  IN  SOA masina.falatrax.hu. administrator.fa-
latrax.hu. (
   22508          ; serial number
   900            ; refresh
```

```
        600             ; retry
        86400           ; expire
        3600          ) ; minimum TTL
;
;  Zone NS records
;  There are two DNS servers holding this domain
@    NS       masina.falatrax.hu.
@    NS       csodagep.falatrax.hu.
;
;  Zone records for Falatrax.hu
;
@       600      A       10.10.1.52
@       600      A       10.10.2.200
@       600      A       10.10.2.211
hulla       900       A       10.10.2.211
csodagep              A       10.10.1.52
masina              A       10.10.2.200
DNS       1200       A       10.10.1.100
      1200       A       10.10.2.100
;
;  Delegated sub-zone:  jh.falatrax.hu.
;
jh                  NS       csodagep.falatrax.hu.
;  End delegation
```

Hol találhatók az erőforrásrekordok?

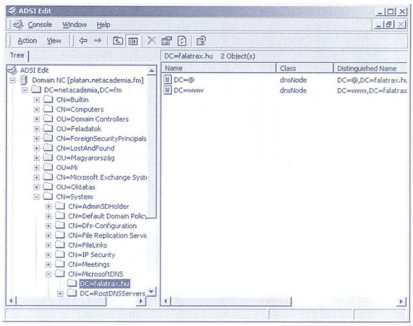
A standard zónák rekordjai a zónák nevét és a .dns kiterjesztést viselő fájlokban *(például falatrax.hu.dns)* találhatók a \winnt\system32\dns mappában.

Az Active Directoryval integrált zónák erőforrásrekordjai

Az AD-vel integrált zónák adatai AD objektumokban tárolódnak, amelyek két fő osztályba sorolhatók:

- A dnsZone osztály olyan AD zónákat reprezentál, amelyek dnsNode objektumokat tartalmaznak. Ez az osztály a szövegfájlban tárolt standard zónák megfelelője az AD-ben. A dnsZone objektumoknak van egy dnsProperty attribútuma, amely a zóna fontos jellemzőit határozza meg – például, hogy dinamikusan frissíthető-e?
- A dnsNode osztály a zóna egyes rekordjaira vonatkozik. Minden dnsNode objektumnak van egy dnsRecord attribútuma, amely az erőforrásra vonatkozó adatokat tárolja.

Az AD-vel integrált zónák tárolási helyét a Windows 2000 Support Toolsban megtalálható ADSI Edittel tekinthetjük meg. Felettébb érdekes, hogy a zóna a tartományi adatok között tárolódik, és nem a Configuration részben. Vajon miért?



A Windows 2000-ben támogatott erőforrásrekordok

A Windows 2000 az összes, RFC-ben definiált rekordtípust támogatja, bár ezek jórészét csak ritkán vagy sosem használjuk. Az alábbiakban ismertetjük a leggyakoribb rekordokat. A típust *(a név mellett zárójelben)* és a szintaxist is feltüntető leírásokat példa is kíséri.

Címrekord (A)

DNS tartománynevet 32 bites IPv4 címhez kapcsol.
szintaxis: tulajdonos A IPv4_cím
példa: csodagep A 10.10.2.200

IPv6 címrekord (AAAA)

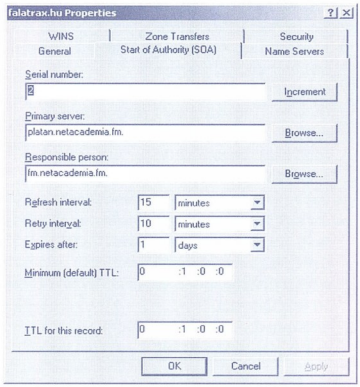
DNS tartománynevet 128 bites IPv6 címhez kapcsol.
szintaxis: tulajdonos AAAA IPv6_cím
példa: ipv6host AAAA 4321:0:1:2:3:4:567:89a:b

Alias, illetve kanonikus név (CNAME)

A tulajdonos mezőben álló aliast rendel egy másik aliashoz vagy egy valódi tartománynévhez. Ehhez a rekordhoz egy A rekordnak is tartoznia kell, amely a DNS tartománynévtér egy érvényes csomópontját adja vissza. A teljes alias ponttal *(„.")* végződik.
szintaxis: alias CNAME kanonikus_név
példa: ns1 CNAME masina.falatrax.hu

Start of Authority (SOA)

A zóna önleírója. Tartalmazza a zóna sorozatszámát, ami a replikációhoz kell, a replikáció gyakoriságát *(refresh)*, hiba esetén mennyi időnként próbálkozzon a secondary *(retry)*, s ha végképp nem tudja felvenni a kapcsolatot a primaryval, akkor mennyi idő múlva állítsa le a zónát, mert a rekordok frissessége több, mint kétséges *(expire)*. Apropó, expire. Egyik kedves olvasónk a múltkori, első rész után kérte, hogy majd említsem meg a SOA ismertetésénél az alapértelmezett értékeket:
Refresh=15 perc, retry=10 perc, expire=1 nap. Akinek ez nem tetszik, kattintson jobb gombbal a zónáján, Properties, ás állítsa át. Az 1 napnyi expire valóban elég kevés lehet, hisz így egy primary-secondary páros nem él túl egy hétvégét, ha a primary elhal.



Levelezés-szervező (MX)

Segíti az üzenetek útvonalának kiválasztását a céltartományban lévő levelezés-kiszolgálóhoz. Ha több ilyen is létezik, akkor a köztük fennálló sorrendiséget a kétjegyű preferencia-értékek határozzák meg. Mindegyik MX rekordhoz tartoznia kell egy ugyanabban a zónában található A rekordnak is.

szintaxis: tulajdonos MX preferencia-érték levelezés-kiszolgáló_neve

példa: falatrax MX 10 mail.falatrax.hu

Mutató (PTR)

Fordított névátalakításra használják, a tulajdonos mezőben szereplő IP címet egy DNS névhez kapcsolja. Többnyire csak az in-addr.arpa tartományban találkozhatunk vele a cím>név összerendelések fordított lekérdezésénél. Legtöbbször egy normál lekérdezési zónában lévő A rekordra mutat.

szintaxis: tulajdonos PTR céltartomány_neve

példa: 200 PTR masina.falatrax.hu

Szolgáltatás-azonosító (SRV)

Lehetővé teszi egy adott szolgáltatást nyújtó kiszolgáló, például egy Windows 2000 Active Directory tartományvezérlő azonosítását. Segítségével a hasonló, TCP/IP alapú szolgáltatásokat biztosító kiszolgálók listája egyetlen DNS lekérdezéssel kideríthető. Elsősorban a Windows 2000 AD igényli a jelenlétét, ahol az összes releváns DNS rekord automatikusan bekerülhet a DNS-be.

szintaxis: szolgáltatás.protokoll.név TTL SRV prioritás terheléselosztási_súly port cél

példák:

1. _ldap._tcp.dc._msdcs 600 SRV 0 100 389 masina.falatrax.hu
2. 600 SRV 0 100 389 csodagep.falatrax.hu
3. 600 SRV 0 100 389 hilo.falatrax.hu

DNS üzenetek

DNS üzenetet két DNS kiszolgáló, illetve egy DNS kiszolgáló és egy DNS ügyfél küldhet egymásnak, rendszerint UDP protokollal, az 53-as porton keresztül. Egyes esetekben, különösen a válaszokban, az üzenet hosszúsága meghaladhatja az UDP csomag adatmezőjének nagyságát. Az üzenet csonkolását a DNS kiszolgáló egy jelzőbit átállításával tudatja. Az ügyfél ezután újra érintkezésbe lép a kiszolgálóval az 53-as porton, de most már TCP kapcsolatot épít ki, és megismétli a lekérdezést. A TCP protokollal hosszabb adatfolyamok is biztonságosan továbbíthatók. Ez a megoldás a két átviteli protokoll előnyeit egyesíti: kis adatmennyiségnél az UDP hatékonysága, míg hosszabb adatfolyamnál a TCP megbízhatósága emelkedik ki.

A DNS üzenetek felépítése

A DNS eredetileg kézzel frissített, statikus adatok dinamikus lekérdezésére szolgált, amihez kétféle üzenetet használtak: lekérdezést és választ. A dinamikus frissítéssel együtt megjelent a frissítési üzenet, amelynek szerkezete a lekérdezésekére vezethető vissza. Emiatt csak a két fő üzenettípus felépítését mutatjuk be.

A DNS lekérdezések felépítése

Az DNS lekérdezések felépítése az alábbi ábrán látható formát követi. Az üzenet fejléce mindig 12 bájt hosszú, míg a lekérdezéseket és a válaszokat tartalmazó, további rész nagysága változó.

12 bájt	párbeszéd azonosító	állapotjelzők
	leérdezés-számláló	válaszrekord-számláló
	felügyelt rekord számláló	kiegészítő rekord számláló
Változó hosszú	lekérdezések *(változó hosszú)*	
	válaszrekordok *(változó hosszú)*	
	felügyelt rekordok *(változó hosszú)*	
	kiegészítő rekordok *(változó hosszú)*	

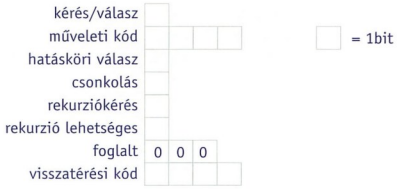
A DNS lekérdezések általános felépítése

A DNS üzenetek fejléce

A DNS üzenetek fejlécét a következő, 16 bites mezők alkotják:

- A párbeszédazonosító a DNS tranzakciók megkülönböztetésére szolgál. A kapcsolat kezdeményezője hozza létre, és a másik fél is ugyanezt az értéket helyezi el a válaszban. Több tranzakció egyidejű bonyolításakor ennek a számnak a segítségével állíthatók párba a kérések és válaszok.
- Állapotjelzők *(lásd ábra)*
- A lekérdezés-számláló megmutatja, hány bejegyzés található a DNS üzenet kérdés részében.
- A válaszrekord-számláló jelzi, hogy hány bejegyzés került a válasz válaszrekordok részébe.
- A felügyelt rekord számláló a kiszolgáló felügyelete alá tartozó rekordok számát adja meg.
- A kiegészítő rekord számláló értéke az üzenetben lévő egyéb rekordok száma.

A jelzők mezőben lévő állapotjelzőkből a kiszolgáló vagy az ügyfél fontos következtetéseket vonhat le:

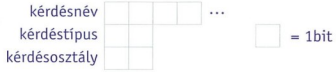


A DNS üzenetek állapotjelzői

- **Kérés/válasz:** Névszolgáltatási kérést jelez, ha értéke 0x0, és választ, ha 0x1. *(1 bit)*
- **Műveleti kód:** A névszolgáltatási művelet típusát határozza meg *(4 bit)*.
  - 0x0 lekérdezés
  - 0x1 inverz lekérdezés
  - 0x2 kiszolgálóállapot-lekérdezés
- **Hatásköri válasz:** A kiszolgáló ezzel jelzi a válaszban, hogy a kérdés részben szereplő tartománynév az ő felügyelete alá tartozik-e? *(1 bit)*
- **Csonkolás:** 0x1 értéke azt jelzi, hogy a teljes válasz nem fér bele egy UDP csomagba. Ekkor csak a válasz első 512 bájtja kerül a csomagba. *(1bit)*
- **Rekurziókérés:** Rekurzív lekérdezésnél értéke 0x1. Ha értéke 0x0, és a megszólított névkiszolgálónak nem tartozik a hatáskörébe a kérés, akkor az azoknak a DNS kiszolgálóknak a listáját küldi vissza, amelyek sikerrel válaszolhatnak. Ez a delegáció kezelésének módja a névátalakítás folyamatában. *(1bit)*
- **Rekurzió lehetséges:** A DNS kiszolgálók a 0x1 érték beállításával jelzik, hogy képesek kezelni a rekurzív kéréseket. *(1 bit)*
- **Foglalt:** Ahogy a neve is mutatja, nem használható, értéke mindig 0x0. *(3 bit)*
- **Visszatérési kód:** A 0x0 érték a sikeres válasz jele *(névlekérdezésnél ez például azt jelenti, hogy a cím megtalálható a válaszban)*. A 0x3 értéket akkor veszi fel, ha a hatásköri DNS kiszolgáló nem talál rekordot a lekérdezett névhez.

A DNS lekérdezések kérdésbejegyzései

A DNS lekérdezés tárgyát képező tartománynév a kérdésbejegyzésben található.



A kérdésmező szerkezete

- **Kérdésnév:** A lekérdezett tartománynév helye. Korábban szó esett már róla, hogy a DNS tartománynevek címkék sorozatából állnak *(a falatrax.hu például a falatrax és hu címkékből)*. A kérdésnév mezőben a címkéket azok hossza előzi meg, a pontok kimaradnak, és a karaktersorozatot 0 zárja. A falatrax.hu például így néz ki: 0x8falatrax0x2hu0x0.
- **Kérdéstípus:** Meghatározza, hogy a válaszban milyen rekordoknak kell szerepelni.
  - 0x01 címrekord *(A)*
  - 0x02 névkiszolgáló *(NS)*
  - 0x05 alias *(CNAME)*
  - 0x0C *(12)* mutató *(PTR)*
  - 0x0F *(15)* levelezés-szervező *(MX)*
  - 0x21 *(33)* szolgáltatásazonosító *(SRV)*
  - 0xFB *(251)* növekményes zónaátvitel *(IXFR)*
  - 0xFC *(252)* standard zónaátvitel *(AXFR)*
  - 0xFF *(255)* minden rekord
- **Kérdésosztály:** Értéke általában 0x00-01, ami az IN osztályt jelzi.

Erőforrásrekordok

A DNS erőforrásrekordok felépítése

A DNS kiszolgáló válaszüzenetének válasz-, hatásköri és kiegészítő részébe kerülnek a kérés feldolgozásának eredményét képező rekordok, melyek szerkezetét a 7. ábra szemlélteti.

- Erőforrásrekord-név: Változó hosszúságú mező, amely a DNS tartománynevet tartalmazza, az előző részben kifejtett kérdésnév mező formátuma szerint.
- Rekordtípus: lásd a kérdéstípust az előző részben.
- Rekordosztály: Jelenleg csak egy kódot használunk (0x00-01) az Internet osztály jelölésére.
- TTL: A rekord felhasználhatóságát határozza meg másodpercben kifejezve.
- Adatmezőhossz: Az erőforrásadatok nagyságát adja meg.
- Adatmező: A rekordtípus szerinti, változó nagyságú adat.

A DNS frissítési üzenetének felépítése

A DNS frissítési üzenet felépítése nagyon hasonlít a lekérdezésekéhez, sok mező teljesen megegyezik. A fejléc határozza meg a frissítési műveletet, a rekordmezők hordozzák a változásokat.

12 bájt	párbeszédazonosító	állapotjelzők
	zónabejegyzés-számláló	szükséges rekord számláló
	frissített rekord számláló	kiegészítő rekord számláló
Változó hosszú	zónabejegyzések *(változó hosszú)*	
	szükséges rekordok *(változó hosszú)*	
	frissített rekordok *(változó hosszú)*	
	kiegészítő rekordok *(változó hosszú)*	

A DNS frissítési üzenet általános felépítése

A DNS frissítés állapotjelzői

A DNS frissítési üzenet állapotjelzők mezőjében a lekérdezéséhez hasonló állapotjelzőket találunk.

kérés/válasz								
műveleti kód	0	1	0	1				= 1bit
foglalt	0	0	0	0	0	0	0	
visszatérési kód								

A DNS frissítés állapotjelzői

- Kérés/válasz: Frissítési kérést jelez, ha értéke 0x0, és választ, ha 0x1. (1 bit)
- Műveleti kód: 0x5 értéke jelzi a frissítést. (4 bit)
- Visszatérési kód: A frissítési kérelmekre adott válasz eredménye.
  - 0x0 A frissítés hibátlanul sikerült.
  - 0x1 Formátumhiba – a DNS kiszolgáló nem érti az üzenetet
  - 0x2 A kérelem feldolgozása közben a DNS kiszolgáló hibát észlelt.
  - 0x3 Nem létezik az a név, aminek kellene.
  - 0x4 Nem támogatott műveleti kód.
  - 0x5 Biztonsági okokból, illetve a házirend miatt a DNS kiszolgáló a műveletet nem hajthatja végre.
  - 0x6 Létezik az a név, aminek nem kellene.
  - 0x7 Létezik az a rekordcsoport, aminek nem kellene.
  - 0x8 Nem létezik az a rekordcsoport, aminek kellene.
  - 0x9 A zóna részben megadott zóna nem tartozik a kiszolgáló hatáskörébe.
  - 0xA A szükséges, illetve a frissített rekordok mezőkben lévő név nem található a zóna mezőben szereplő zónában.

Névlekérdezési üzenet

A névlekérdezési üzenet az RFC 1034-ben leírt formátumot használja, ami megegyezik a DNS lekérdezési üzenetével. A névátalakításnál bemutatott NM napló éppen a névlekérdezést mutatja be, ahol a következő mezőket találjuk:

- Lekérdezés-azonosító: Egyedi szám, ami alapján az átalakító program párba tudja rendezni a lekérdezéseket és válaszokat.
- Állapotjelzők: Többek között a standard lekérdezés és a rekurziókérés jelzésére szolgál.
- Kérdésbejegyzés-számláló: Értéke 1.
- Kérdésrész: Itt kell elhelyezni az átalakítandó tartománynevet (a példában masina.falatrax.hu) és visszaküldendő rekordokat (A címrekord).

Névlekérdezési válaszüzenet

Az átalakítók a névlekérdezésre ilyen üzenettel válaszolnak, a lekérdezési üzenettel azonos formátumban. A névátalakításnál található NM napló második felében tanulmányozhatjuk a válasz felépítését, amelyben a mezők jelentése:

- Lekérdezés-azonosító: Egyedi szám, ami alapján az átalakító program párba tudja rendezni a lekérdezéseket és válaszokat.
- Állapotjelzők: Ezekkel lehet jelezni, hogy válaszról van szó, és az átalakítás sikeresen befejeződött.
- Kérdésbejegyzés-számláló: Értéke 1.
- Válaszbejegyzés-számláló: Értéke 1.
- Kérdésrész: Megegyezik a lekérdezési üzenetben lévő kérdésrésszel.
- Válaszrész: Ide kerülnek a lekérdezésben kért rekordok *(példánkban a címrekord, amelyben megtaláljuk a kérdéses tartomány IP címét)*.

Az aliasok részben is tanulmányozhat az olvasó egy lekérdezés-válasz naplót, amely egy kicsit különbözik az előzőtől. Ebben az átalakító az ns1.falatrax.hu-hoz tartozó A rekordot próbálja keresi, de csak egy CNAME rekordot talál. A válaszba azért emellett még a kanonikus névhez tartozó A rekordot is beleteszi.

Fordított lekérdezési üzenetek

A fordított lekérdezési üzenetek szerkezete megegyezik a normál lekérdezésekével, mindössze két különbség van:

- A céltartomány neve más: az átalakító a fordított le kérdezéseknél a kérdéses IP cím alapján in-addr.arpa tartománynevet hoz létre.
- A válaszba nem A, hanem PTR rekord kerül.

A fordított lekérdezésre adott válaszüzenetek felépítése megegyezik a normál válaszüzenetekével, de az A helyett PTR rekordot tartalmaz.

Névfrissítési üzenet

A névfrissítési üzenetek formátumát az RFC 2136 határozza meg, amin már túlestünk a DNS frissítési üzenetek felépítésének tárgyalásakor. Például szolgáljon a DNS dinamikus frissítéséről szóló részben látható NM napló, melynek legfontosabb mezőit így kell beállítani:

- Lekérdezés-azonosító: A lekérdezési üzenethez hasonlóan, ennek is van azonosítója, hogy a küldő a választ az eredeti kérdéshez tudja rendelni.
- Állapotjelzők: A kérés tényét, és a dinamikus frissítést ezekkel tudjuk jelezni.
- Frissítési zóna: Értéke 1, és a zónarészben van a frissítendő zóna.
- Kötelező zóna: Értéke 2, két kötelező kell megadni.
- Frissítendő zóna: A frissítendő zóna rekordja.

Névfrissítési válaszüzenetek

A névfrissítési válaszüzenetekhez a DNS dinamikus frissítéséről szóló rész NM naplóját vesszük példának. Ebben a válasz azonos a kéréssel, eltekintve attól, hogy a válasz fejlécében lévő állapotjelzők egyike a művelet sikerességét adja tudtul. Ellenkező esetben a válasz a hibaüzenetet tartalmazza.

Összefoglalásul

A DNS korábban csak lehetőség volt a WINS-t használó Windows NT hálózatokban a legtöbb tartományművelet, illetve a nyomtató- és fájlmegosztás megvalósítására, míg a Windows 2000 hálózatokban már kulcsfontosságú szerepet tölt be, és a Windows 2000 Active Directory telepítéséhez nélkülözhetetlen.

Fóti Marcell
marcellf@netacademia.net

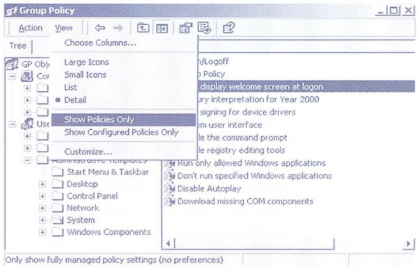
Policy sablon-fájlok szerkesztése 1.rész

A felhasználói, illetve a számítógépes beállítások központi kezelése már a Windows NT 4.0-ás környezetben is jól használható volt. A Windows 2000 csoportos házirendje *(Group Policy)* lényegesen fejlettebb az NT-ben található System Policy-hoz képest. A csoportos házirend segítségével már nem csak a regisztrációs adatbázis bejegyzéseinek értékeit tudjuk módosítani, hanem lehetőségünk van például programok telepítésére vagy logon, logoff, startup és shutdown scriptek központi kezelésére is.

Az úgynevezett sablonfájlok *(administrative templates)* határozzák meg, hogy a regisztrációs adatbázisban mely beállítások változtathatók meg a System Policy Editor, illetve a Group Policy MMC modul segítségével. A System Policy Editor és a Group Policy MMC modul alapértelmezésben az alábbi sablonfájlok használatát kínálják fel:

- **Common.adm *(System Policy):*** NT 4.0, Windows 95 és 98 operációs rendszereknél is használható. Elsősorban felhasználói felületre vonatkozó tiltásokat tartalmaz.
- **Winnt.adm *(System Policy):*** NT 4.0 operációs rendszer esetében a Shell, System és Profile kategóriákban tudjuk a beállításokat módosítani.
- **System.adm *(Group Policy):*** Ebben a sablonban a Windows 2000 operációs rendszer beállításai találhatók meg.
- **Inetres.adm *(Group Policy):*** Az Internet Explorer böngészővel kapcsolatos tiltások gyűjteménye.
- **Conf.adm *(Group Policy):*** A NetMeeting beállításait tartalmazza.

Természetesen új sablonokat is létrehozhatunk, ha elsajátítjuk a szerkesztésük fortélyait. Cikkünkkel ebben szeretnénk segítséget nyújtani. Egyszerű példákkal szemléltetve részletesen bemutatjuk az utasítások használatát. Hogy az utasítások kipróbálásakor nehogy problémák jelentkezzenek az operációs rendszer működésében, a cikkben szereplő példák alkalmazásakor a regisztrációs adatbázisban egy funkció nélküli kulcs alatt jelennek meg próbabejegyzéseink. A sablonfájlok szerkesztését az egyszerűség kedvéért a System Policy Editor felületén keresztül mutatjuk be, de a példák kipróbálhatók a csoportos házirendben is. Figyelembe kell venni azonban, hogy a Windows NT-vel szemben a Windows 2000 operációs rendszernél az új típusú csoportos házirendbeállításoknak kitüntetett helyük van a regisztrációs adatbázisban. A felhasználóra *(HKEY_CURRENT_USER hive)* és a számítógépre *(HKEY_LOCAL_MACHINE hive)* vonatkozó bejegyzések a \Software\Policies, illetve a \Software\Microsoft\Windows\CurrentVersion\Policies kulcsok alatt jelennek meg. Erre a változtatásra azért volt szükség, hogy elkerülhető legyen a régebbi System Policy egyik kellemetlen hibája. A Windows NT 4.0-ás környezetben gyakran előfordult, hogy a házirend megváltozásakor a régi bejegyzések megmaradtak a regisztrációs adatbázisban. Az új típusú házirend beállításoknál ez a probléma már nem fordulhat elő, sőt a házirendfájl letörlése esetén az általa kiváltott hatás is visszavonódik. *(Fogalmazhatunk úgy is, hogy az új típusú házirend az operációs rendszer tudtával és vezetésével végzi dolgát, így mindig konzisztens önmagával, míg a régi - megtűrt kiegészítésként - egyszerűen „alátesz" az NT4 operációs rendszernek – a hatás visszavonása egy ellentétes hatású „alátevéssel" szüntethető meg – a szerk).* Ugyanakkor az új csoportos házirendhez is készíthetünk olyan sablont, amely nem az erre fenntartott helyeken változtatja meg a regisztrációs adatbázisban a bejegyzéseket, hanem – ugyanúgy mint régen – tetszőleges helyeken módosíthatja azt. Ha ilyet használunk, figyelembe kell vennünk, hogy bizonyos esetekben a házirend megváltozásakor a korábbi beállítások is megmaradhatnak *(a regisztrációs adatbázis „tetoválására" kerül sor – a szerk.).* Éppen e kellemetlen mellékhatás miatt a régi típusú házirendbeállítások alapértelmezésben nem jelennek meg a csoportos házirend felhasználói felületén, hogy nehogy összekeverjük, és véletlenül használjuk ezeket. A View menü Show Policies Only parancsával lehet megjeleníteni, illetve eltüntetni minden régi típusú bejegyzést:



A régi típusú házirendbeállítások megjelenítése és eltüntetése

A felhasználói felületen piros ikonnal jelenik meg minden olyan beállítás, amely nem az erre fenntartott kulcsok alatt módosítja a regisztrációs adatbázist.

A System Policy Editor működésének áttekintése

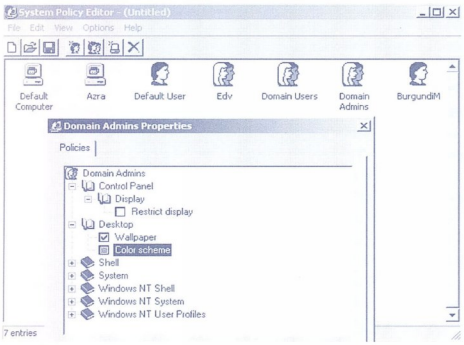
A házirendbeállítások alapját képező sablonfájlokat a Policy Editor felhasználói felületén az Options menü Policy Template parancsával tudjuk betölteni, illetve szükség esetén eltávolítani. A házirendbeállításokat tartalmazó fájl létrehozásakor alapértelmezésben csak a Default User és a Default Computer ikon jelenik meg. Az Edit menü Add User, Add Computer, Add Group parancsaival tudunk új felhasználót, számítógépet, illetve globális felasználói csoportot hozzáadni. Érdemes figyelni arra, hogy az általunk beírt nevek valódiságát sajnos nem ellenőrzi a program. A fájlt Windows NT 4.0-ás környezetben ntconfig.pol néven kell elmentenünk, mégpedig a NETLOGON megosztás alá.

A házirend szerkesztése nagyon egyszerű: ha a megfelelő ikonra kettőt kattintunk, akkor megjelennek a rendelkezésre álló beállítások. Minden beállításnak három állapota van:

- **Engedélyezett *(Enabled):*** A jelölőnégyzetben kis pipa látszik.
- **Letiltott *(Diasbled):*** A jelölőnégyzet fehér.
- **Érintetlen *(Not Configured):*** A jelölőnégyzet szürke.

nehézségi fok:

Ebben az állapotban a regisztrációs adatbázisban szereplő értékek változatlanok maradnak.



A beállítás három lehetséges állapota

A házirend beállításai az alábbi szabályszerűségek alapján érvényesülnek:

- Ha csak a Default Usert, illetve a Default Computert használjuk, akkor a beállítások a tartomány összes felhasználójára és számítógépére érvényesek lesznek.
- Abban az esetben, ha ugyanaz a beállítás eltérő értéket kap a Default User és valamelyik globális csoport esetében, akkor a globális csoport beállítása érvényesül. Ha például a Default User-nél a Logon, míg a Domain Admins csoportnál a Blank képernyővédőt állítjuk be, a rendszergazdáknál a Blank képernyővédő jelenik meg.
- Hasonlóan az előbbi esethez, az egyedi felhasználó beállításai felülírják a Default User-ét.
- Az egyedi felhasználó beállításai mindig előnyt élveznek a csoporttal és a Default User-rel szemben, függetlenül attól, hogy a beállításokat módosítjuk-e vagy sem.
- A globális csoportoknál, ha a beállítások átfedik egymást, akkor a rangsorban előrébb lévő csoportnál meghatározott értékek íródnak a regisztrációs adatbázisba. A csoportok sorrendiségét a System Policy Editor Options menü Group Priority menüpont alól elérhető párbeszédablakban állíthatjuk be.
- Ha valamelyik beállítást a csoportok szintjén nem változtatjuk meg *(a jelölőnégyzet szürke)*, a Default User esetében viszont engedélyezzük, akkor a Default User-nél megadott érték kerül a registrybe. Ez a szabályszerűség a szóban forgó csoport azon tagjára nem vonatkozik, amelyet egyedileg is felvettünk a házirend beállításait tartalmazó fájlba. Ilyenkor a csoport és a Default User beállításai nem érvényesülnek.
- Ha a csopotok szintjén megadott beállítások nem fedik át egymást, akkor a végrehajtás során összeadódnak.
- Adott csoportsorrend esetén, a rangsorban hátul lévő csoportnál engedélyezett beállítás akkor érvényesül, ha a sorban előrébb lévő csoportok esetében a szóban forgó beállítást nem módosítjuk.

A sablonfájl alapegységei

A házirenden keresztül a regisztrációs adatbázis felhasználóra *(HKEY_CURRENT_USER hive)* vagy a számítógépre vonatkozó *(HKEY_LOCAL_MACHINE hive)* részét tudjuk módosítani. A sablonfájlban a CLASS User vagy a CLASS Machine utasítással adhatjuk meg, hogy a bejegyzéseket melyik hive alatt szeretnénk megváltoztatni.

Az összetartozó beállításokat csoportokba szervezhetjük. A csoport kezdetét a CATEGORY, végét az END CATEGORY jelzi. Természetesen minden kategóriának nevet kell adnunk, amelyet többféleképpen is megtehetünk. A CATEGORY után álló karaktersort a System Policy Editor a kategória neveként azonosítja. Például: CATEGORY Kategória Neve. Abban az esetben, ha a karaktersor szóközt is tartalmaz, akkor idézőjelek közé kell tennünk. Például: CATEGORY "Kategória Neve". A név megadásának ilyen formája egyszerű és kézenfekvő, ennek ellenére nem ez az általánosan elfogadott gyakorlat. A kategórianév helyén minden esetben egy szöveges változó áll, amelynek a sablonfájl végén, az úgynevezett [strings] szekcióban adunk értéket. A szöveges változó mindig dupla felkiáltójellel kezdődik, majd ezt követi a változó neve:

```
CLASS User
CATEGORY !!CategoryName
...
END CATEGORY

[strings]
CategoryName = „Első kategória“
```

Egy szöveges változót a sablonon belül többször is használhatunk. A kategóriák tetszőleges mélységig egymásba ágyazhatók, amelyek a Policy Editor felhasználói felületén kinyitható, illetve bezárható, kék színű könyv formájában jelennek meg.

A kategóriák alatt tűnnek fel a házirendbeállítások, amelyek előtt egy jelölőnégyzet látható. Ezzel tudjuk meghatározni, hogy az adott beállítás érvényesüljön-e vagy sem. A jelölőnégyzetet a POLICY kulcsszóval hozzuk létre, amelynek nevét az általánosan elfogadott gyakorlat szerint szöveges változó segítségével határozzuk meg. A kategóriához hasonlóan a POLICY blokk végét az END POLICY-val kell jeleznünk.

A házirend használatának egyszerűsítése végett sokszor lehetőséget kell teremtenünk arra, hogy a felhasználói felületen keresztül is megadhassunk értékeket. Az ilyen jellegű adatbeviteli lehetőséget a PART kulcsszóval tudunk létrehozni. Az eddigiekhez hasonlóan a PART nevét is szöveges változóval szokás megadni. A PART-nak számos típusa van *(például EDITTEXT, LISTBOX, stb)*, amelyet az elnevezés után tüntetünk fel:

```
PART !!PartName        LISTBOX
   ...
END PART
```

Az egyes típusokról és használatukról a későbbiekben részletesen is szó lesz.
A regisztrációs adatbázis sok bejegyzésének csak két értéke lehet: nulla vagy egy. Előfordul azonban, hogy egy beállítás engedélyezése a házirendben számos érték megváltozását vonja maga után a regisztrációs adatbázisban. Máskor nem elegendő, ha a sablonfájlba közvetlenül adjuk meg az értékeket, a felhasználói felületen is kell változtatási lehetőséget biztosítanunk. Az értékadásnak különböző formái vannak tehát, amelyeket egyszerű példákon keresztül fogunk áttekinteni. Elkerülendő, hogy a példák kipróbálásakor zavar támadjon a számítógépek működésében, bejegyzések a HKEY_CURRENT_USER\AdmTeszt kulcs alá kerülnek. A házirend végrehajtásakor az AdmTeszt kulcs automatikusan létrejön. Ezt azután a Registry Editorral lehet törölni, mivel a házirenden keresztül kulcstörlés nem valósítható meg.

Az értékadás egyszerű formái

A legegyszerűbb esetben az értékadáshoz elegendő meghatároznunk a regisztrációs adatbázis megfelelő kulcsának az elérési útját, valamint a bejegyzés pontos nevét. Ilyenkor a bejegyzés típusa REGDWORD, értéke pedig 1 lesz. A beállítás letiltása során a bejegyzés törlődik a registryből.
A kulcs elérési útját a KEYNAME segítségével adjuk meg, amely a CATEGORY, POLICY és PART kulcsszavak után is szerepelhet. A CATEGORY szintjén meghatározott elérési út az egész kategóriára érvényes lesz. Ezt azonban felül tudjuk bírálni, például a POLICY kulcsszó után megadott másik útvonallal. Ez az új elérési út természetesen nem vonatkozik más beállításokra is. A KEYNAME után az elérési utat csak akkor kell idézőjelbe tenni, ha valamelyik kulcs nevében szóköz is szerepel.
A bejegyzés nevét a VALUENAME utáni karaktersor határozza meg. Bár idézőjelet itt is csak akkor kell használni, ha az elnevezés szóközt is tartalmaz, az egységes kezelés érdekében érdemes minden esetben alkalmazni.

```
CLASS User
CATEGORY !!SSamples
KEYNAME "ADMTeszt"
  POLICY !!Sample1
    VALUENAME "Sample1őValue"
  END POLICY
END CATEGORY

[strings]
SSamples = "Példák"
Sample1 = "Példa 1: Policy"
```

Ha az alapértelmezettől eltérő típusú vagy értékű bejegyzést szeretnénk létrehozni vagy megváltoztatni, akkor a VALUEON, illetve a VALUEOFF utasításokat kell alkalmaznunk:

```
POLICY !!Sample2
  VALUENAME "Sample2őValue"
    VALUEON NUMERIC 1
    VALUEOFF NUMERIC 0
END POLICY
```

Az iménti példában is REG_DWORD típusú bejegyzés jön létre. A beállítás érvénytelenítése után viszont már nem törlődik a bejegyzés, hanem nulla értéket vesz fel. Ha elhagyjuk a NUMERIC kulcsszót, akkor a bejegyzésünk REG_SZ típusú lesz. Figyeljünk arra, hogy, hogy a NUMERIC után mindig decimális formában kell beírni az értéket. Ha a házirend letiltásakor az első példához hasonlóan törölni szeretnénk a bejegyzést, akkor vagy a VALUEOFF DELETE parancsot kell használnunk vagy hagyjuk el a VALUEOFF NUMERIC 0 sort. Amennyiben a beállítás engedélyezésekor valamelyik bejegyzést szeretnénk törölni a regisztrációs adatbázisból, akkor írjuk a VALUEON után a DELETE kulcsszót.
Olyan eset is előfordulhat, hogy egy beállításhoz több bejegyzést is létre kell hoznunk a regisztrációs adatbázisban. Ezt a problémát az ACTIONLISTON és az ACTIONLISTOFF parancsokkal tudjuk megoldani.

```
POLICY !!Sample3
  KEYNAME "ADMTeszt\Actionlist"
  VALUENAME "ActionListState"
  ACTIONLISTON
    VALUENAME "ALőValue4" VALUE NUMERIC 1
    VALUENAME "ALőValue5" VALUE NUMERIC 2
    VALUENAME "ALőValue6" VALUE NUMERIC 3
  END ACTIONLISTON
  ACTIONLISTOFF
    VALUENAME "ALőValue4" VALUE NUMERIC 0
    VALUENAME "ALőValue5" VALUE NUMERIC 0
    VALUENAME "ALőValue6" VALUE NUMERIC 0
  END ACTIONLISTOFF
END POLICY
```

Az ACTIONLISTON és ACTIONLISTOFF utasítások alkalmazásakor az értéket minden esetben a VALUE kulcsszóval kell megadnunk. A bejegyzés törlésére vonatkozó szabályok megegyeznek a VALUON, VALUEOFF parancsoknál leírtakkal. A System Policy Editor sajátságosan kezeli az ACTIONLISTON, ACTIONLISTOFF utasításokat. Ha mentés után ismét megnyitjuk az ntconfig.pol fájlt, akkor tapasztalni fogjuk, hogy a jelölőnégyzet szürke lesz, függetlenül attól, hogy milyen beállítással mentettük el korábban. A beállítás eljut ugyan a felhasználókhoz, de a felhasználói felületen nem látszik, hogy kinek engedélyeztük. A problémát úgy tudjuk áthidalni, hogy a regisztrációs adatbázisban eltároljuk az Actionlist aktuális állapotát. Erre szolgál a fenti példában az „ActionListState" változó. Tapasztalataim szerint a Group Policy-nál már megszűnt ez a probléma.

Windows 2000

Értékadás a grafikus felületről

A PART különböző típusaival változatos adatbeviteli lehetőséget biztosíthatunk a rendszergazdák számára, így az értékek esetleges változásakor nem kell a sablonfájlt módosítani. A sablonfájl strukturális felépítéséből adódóan a PART minden esetben a POLICY kulcsszó után szerepel. A PART elnevezését a korábban már említett gyakorlatnak megfelelően szöveges változó formájában szokás megadni. Közvetlenül a név után kell meghatároznunk a PART típusát, amelyet az alábbiak közül választhatunk ki:

- CHECKBOX
- EDITTEXT
- DROPDOWNLIST
- COMBOBOX
- LISTBOX
- NUMERIC
- TEXT

Az értékadáshoz természetesen szükséges a bejegyzés nevének az ismerete, amelyet itt is a VALUNAME segítségével adunk meg. Mint később látni fogjuk, ez alól csak a LISTBOX és a TEXT típusok képeznek kivételt. Figyeljünk arra, hogy ha a POLICY szintjén is szerepel a VALUENAME, akkor két bejegyzés jön létre a regisztrációs adatbázisban.

```
POLICY !!Sample4
  VALUENAME "Policy_Value"
    PART !!Part1      EDITTEXT
      VALUENAME "EditTextőValue"
    END PART
END POLICY
```

A fenti beállítás engedélyezésekor PolicyValue és EditText_Value néven is létrejön egy bejegyzés a regisztrációs adatbázisban.

CheckBox

Használatával egy jelölőnégyzet jelenik meg a Policy Editor párbeszédablakának alsó részén. Alapértelmezésben REG_DWORD típusú bejegyzést hozhatunk létre, melynek értéke 1 lesz. Ha a jelölőnégyzet üres, akkor a bejegyzés törlődik a regisztrációs adatbázisból. A CHECKBOX típus esetén az alábbi parancsokat használhatjuk:

- **ACTIONLISTON, ACTIONLISTOFF:** Lehetőséget ad egyidőben több bejegyzés értékének megváltoztatására a jelölőnégyzet kiválasztásakor, illetve a kiválasztás megszüntetésekor.
- **DEFCHECKED:** A jelölőnégyzet automatikusan kijelölt állapotba kerül.
- **VALUEON, VALUEOFF:** Akkor használjuk, ha a bejegyzés alapértelmezett típusát vagy értékét szeretnénk megváltoztatni.

```
POLICY !!Sample5
KEYNAME "ADMTeszt\CheckBox"
  PART !!CheckBoxőPart        CHECKBOX
      VALUENAME "CheckboxőValue"
      VALUEON   "vőon"
      VALUEOFF "vőoff"
  END PART
END POLICY
```

A fenti példában Checkbox_Value néven hozunk létre REG_SZ típusú bejegyzést a regisztrációs adatbázisban, melynek értéke a jelölőnégyzet állapotától függően „v_on" vagy „v_off" lesz.

EditText

Szerkeszthető mezőt tudunk vele megjeleníteni. A regisztrációs adatbázisban REG_SZ vagy REG_EXPAND_SZ típusú bejegyzés jön létre. Az utóbbi típust akkor érdemes használni, ha rendszerváltozót szeretnénk megadni a mezőben (például %Systemroot%). Az EDITTEXT típus gyakran használatos parancsai a következők:

- **DEFAULT <érték>:** Meghatározhatjuk, hogy alapértelmezésben milyen érték jelenjen meg a mezőben.
- **REQUIRED:** A beállítás engedélyezésekor a mezőt ki kell tölteni.
- **EXPANDABLETEXT:** A regisztrációs adatbázisban az alapértelmezett REG_SZ helyett REG_EXPAND_SZ típusú bejegyzés jön létre.
- **MAXLEN <érték>:** A mezőbe beírható karakterek maximális számát tudjuk meghatározni.

Az alábbi példában egy olyan szerkeszthető mezőt hozunk létre, amelynek kitöltése kötelező, és maximum tíz karaktert tartalmazhat:

```
POLICY !!Sample6
KEYNAME "ADMTeszt\EditText"
  PART !!EditTextőPart  EDITTEXTREQUIRED
    VALUENAME "EditText_Value"
    MAXLEN 10
  END PART
END POLICY
```

folytatjuk...

Tomasz Balázs
balazs.tomasz@hu.hypovereinsbank.com

A hálózati forgalom elemzése — 1. rész, elméleti alapok

Ha minden Active Directory probléma megoldását a DNS-ben kell keresni, akkor ugyancsak ökölszabály alapján minden, elsőre zavaros hálózati hiba felderítésekor a Network Monitort KELL használni. Mondogatom ezt már jó ideje mindenféle fórumon, de úgy vettem észre, nem átütő sikerrel. Vannak, akik fel sem telepítik, s vannak, akik az első elkapott hálózati forgalom megtekintésekor borzadnak el, s hiszik, hogy itt a tudomány és a fantasztikum számukra soha meg nem világosodó keverékéről van szó - pedig dehogy. Kedves Olvasóim bizony megtanulták, és sikeresen le is vizsgáztak anno a hálózati architektúrák témaköréből: ez volt az OSI modell.

A sikeres vizsgához néhányan versike formájában be is magolták, hogy:

„Physical, Datalink, Network, Transport, Session, Presentation, Application"

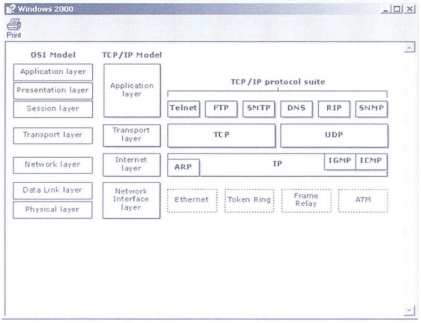
ami magyarul így hangzik:

„Minden vízbe mártott test..." ja nem, ez egy másik versike :)

Csak hát az elmúlt X évben gyakorlatilag egyetlen esetben sem vették hasznát e tudásnak, s így az szép lassan elkopott. Határozottan kijelenthetem azonban, hogy nem tekinthető komoly szakembernek az, aki nem képes olvasni az Ethernet csomagokban. Napjaink hálózati operációs rendszerei és az arra épülő komplex alkalmazások kommunikációja bizony néha olyan pofonegyszerű beállítási *(vagy tervezési)* hibán múlik, hogy utólag a fejünket verjük a falba, hogyan is nem vettük észre azonnal, hogyan is lehettünk annyira vakok!

A láthoz azonban eszköz is kell. A hálózati forgalom nyers adatainak megtekintésére úgynevezett sniffert használunk *(sniff=szimatol)*, mellyel az Ethernet *(Token Ring stb.)* hálózati forgalom bitszintű elemzésére is lehetőség nyílik. Snifferből van buta és okos, drága és olcsó, de érdekes módon az egyik legokosabb ilyen szoftver ingyenes: a Windows NT és a Windows 2000 telepítőlemezén találjuk a Network Monitor programot, melynek funkciókészlete a több százezer forintos egyedi snifferek tudásával veteksik. A teljesség jegyében mindjárt megjegyzem, hogy a NetMon két változatban áll rendelkezésünkre: a beépített ingyenesben a hackerfunkciók le vannak tiltva, ám a Systems Management Server CD-jén található NM extra a csomagkivonatolástól *(coalesce)* a csomagok átírkálásán keresztül Ethernet keretek hálózatba pumpálásáig mindent tud: kész hackerarzenál.

Mit is tud a NetMon? Elsősorban arra képes, hogy a hálózati forgalmat minden adatával egyetemben elénk tárja, hogy abból következtetéseket vonhassunk le. Ezt a rétegzett hálózati architektúra egy alacsony szintjének *(NDIS, Network Driver Interface Specification eszközmeghajtó)* megcsapolásával éri el. Mivel az NDIS meghajtó a lehető legalacsonyabb szoftverréteg a hierarchiában *(alatta már a hálókártya hardvere dolgozik)* gyakorlatilag minden olyan bitet láttatni enged a NetMon, amit a hálózati kártya felfelé, az operációs rendszer felé kibocsát magából. Itt álljunk meg egy pillanatra. Mintha az OSI modellben nem lenne NDIS réteg! Nincs bizony! Az OSI modell csak egy ajánlás arra nézvést, hogy hogyan kell *(ENE)* rétegzett hálózati architektúrákat építeni, ám ezt történelmi okokból gyakorlatilag egyetlenegy gyártó sem tartja be – nem is teheti, hisz az OSI modell helyett a TCP/IP modellt követjük, mely nem szabványos ugyan, de mindenki azt használja, mert mindenki azt használja. Az alábbi ábra a Windows 2000 Helpjéből származik, és tökéletesen mutatja az OSI és a TCP/IP össze *(nem)* függéseit.



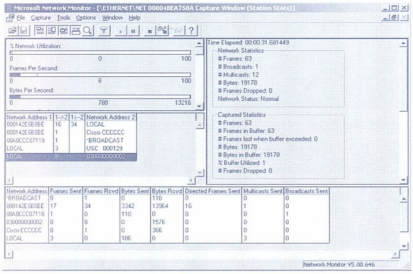
Az OSI elkésett. Addig vacakoltak vele az ISO szabványügyi hivatalban, míg mire készen lett, a világot elborította a TCP/IP, amely szintén rétegzett, csak másképp, és van vele baj bőven. Itt és most nem elemezném az OSI modell rétegeit és előnyeit, elégedjünk meg azzal, ha azt mondom: minden, ami hiányzik a TCP/IP-ből az OSI modellre épített hálózatokon elérhető lenne *(tömörítés, titkosítás, irányított csatornák)*. Ne áltassuk magunkat, a TCP/IP-ben nics titkosítás. Az IPsec, az SSL, a S/MIME, a PPTP, az L2TP mindmind toldozgatás-foldozgatás! És nincs irányított csatorna sem, a WinSock és a Named Pipes – protézis, hogy szép legyen a TCP/IP mosolya.

De térjünk vissza az NDIS-megcsapoláshoz. Ezek szerint van olyan adat, amit nem látunk a NetMonnal, mert a kártya hardverből „lekeveri"? Van bizony! Ilyen az Ethernet fejlécet megelőző, órajelszinkronizációs szerepű Preamble *(8 bájtnyi 10101010)* és a bithibák kiszűrését segítő CRC kód a keret legvégén. Ha a CRC jó, megkapjuk a csomagot *(de a CRC-t nem)*, ha pedig rossz, semmit sem kapunk! Ebből máris leszűrhető az a tanulság, hogy a NetMonnal a hálózat hardveres hibáinak megtalálására csak áttételesen, véletlenül, vagy nagyfokú ravaszsággal nyílik mód, ugyanis ha a HW rossz – nem látunk semmit. Ez az eszköz a magasszintű szoftverek botlásainak felderítésére való, kábelhibák kimérésére hívjunk szakembert.

A NetMonnal alapvetően kétféle hálózaton dolgozhatunk, pont-pont kapcsolatot megvalósító és üzenetszórásos csatornán. A pont-pont kapcsolatra a legjobb példa a kapcsolt telefonvonalak használata, ahol a tárcsázáskor kialakuló kommunikációs útvonal a későbbiekben nem változik, így az elküldött csomagok címzésére semmi szükség nincs, hisz az egyik oldalon feladott adat csakis a kábel másik végén pottyanhat ki. Ide sorolható az ATM is, bár ott virtuális csatornaazonosítókkal meg kell határozni, hogy melyik „kábe-

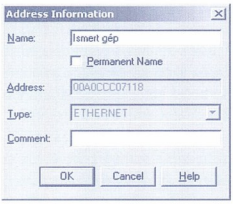
len" kommunikálunk. Üzenetszórásos csatornánál azonban egy adott alhálózat minden gépe „hallja" az összes adást, maximum nem vesz róla tudomást, és nem küldi „fel" az operációs rendszernek. Így működik az Ethernet, a Token Ring és még több más, kihalt hálózati rendszer. Ezeken minden egyes elküldött csomag tartalmazza a feladó és a címzett egyedi azonosítóját, úgynevezett MAC *(Media Access Control)* címét, hogy a hálózati eszközök eldönthessék, vajon a beérkezett csomag nekik szól-e vagy sem. Kezdeti próbálkozásunk szempontjából egyelőre közömbös, hogy HUB vagy SWITCH köt minket össze a hálózattal, mert az ingyenes NetMon úgyis csak azokat a csomagokat képes megjeleníteni, amelyeket gépünk az Ethernet címzés miatt valóban megkap. *(A „vájtszeműbbek" észrevehetik, hogy a főképernyőn található kijelzők a teljes hálózati forgalmat és terhelést mutatják, azaz a NetMon driver valójában mégiscsak promiszkusz üzemmódba kapcsolja az Ethernet kártyát, és csak a felhasználói felület van lebutítva. Ugyanez igaz a NetMon által a rendszerbe illesztett Performance Monitor számlálóról, a "% Network Segment"-ről is: minden Ethernet keretet beszámít, még azokat is, amelyek nem a mi gépünknek szólnak.)*

Indítsuk el a NetMont, és játszadozzunk vele egy kicsit! Telepítés NT4-nél Crtl Panel->Network->Services->Add, Network Monitor Tools and Agent *(vigyázat, TOOLS és Agent, nem sima Agent!)*, Windows 2000-nél Ctrl Panel->Add/Remove Software->Windows Components, Network & Monitoring Tools vagy ehhez hasonló név alatt. Szépen beköltözik a rendszergazda szokásos eszközei közé. Elindítás után így néz ki:



Alapvető használata rém egyszerű, a felhasználói felület nem nagyban különbözik a videomagnóétól, s azt – ugye – az

egész család tudja kezelni, így ezzel sem lehet gond bár a „lejátszás" gomb éppenséggel felvesz. Próbaképpen kapjunk el néhány másodpercnyi hálózati forgalmat, és nézzük, hányféleképpen elemezhetjük. Amíg a csomagok elkapása, gyűjtése folyik, addig a főképernyőn az éppen zajló forgalom statisztikáit láthatjuk. Mely gépek küldenek csomagot és kinek, milyen címzéssel *(unicast, multicast, broadcast)*. Már itt lehetőségünk van arra, hogy ha egy MAC addressről pontosan tudjuk, hogy melyik gépé, akkor a címen jobb gombbal kattintva átírhatjuk a megnevezést, ami a későbbiekben nagyban meg fogja könnyíteni az adatok értelmezését.



Ha esetleg nem tudjuk fejből a hálózat összes gépének MAC addressét, ne csüggedjünk: a NetMon majd azt is kideríti!

Promiszkusz mód

Ez a fogalom nem egy nemi eltévelyedést takar, hanem a hálózati kártyáknak egy olyan üzemmódját, amikor azokat a csomagokat is „felküldik" az operációs rendszernek, amelyeket nem is az adott gépnek címeztek. Akkor hogy kerültek oda? Az Ethernet, a Token Ring és még sok társuk úgynevezett üzenetszórásos módon közvetítik az adatokat: mindenki hallja, de csak az veszi az adást, akinek szól. Olyan, mintha egy teremben kiabálnék valakinek: mindenki hallja, aki ott van, de nem vesz róla tudomást csak az, akinek szólok. A promiszkusz üzemmódba kapcsolt hálózati kártya minden egyes „meghallott" csomagot felküld az operációs rendszernek.

MAC address és címzésmódok

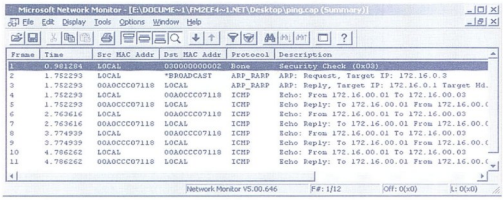
Minden egyes hálózati kártya rendelkezik egy – az esetek elsöprő többségében – gyárilag beleégetett egyedi azonosítóval, a hatbájtos MAC addressel. Ez teszi lehetővé, hogy a használt protokolltól függetlenül megvalósítható legyen a gépek egyenkénti címzése. Maga a MAC address két részre bontható: 3 bájtnyi gyártóazonosítóból és 3 bájtnyi sorozatszámból tevődik össze. *(Például a Katron kártyák 0040F6 gyártóazonosítóval rendelkeznek)*.

Az Etherneten három különböző címzésmód használható. Unicas, multicas és broadcast. A leggyakrabban használt a Unicast, melynél mindössze két gép kommunikál egymással, ilyenkor az Ethernet keret fejlécének címmezőjében a célgép MAC addressét találjuk. Vannak olyan forgalomtípusok, melyeknek minden gépre el kell jutniuk, ezt broadcast címzéssel valósítják meg. Ilyenkor a keret fejlécében nem egy adott gép MAC addresse szerepel címzettként, hanem egy speciális jelzés: 0xFFFFFFFFFFFF. Ezt mindegyik gép magának érzi – a PC-től a hálózati kártyás iratmegsemmisítőig – és a csomagot feldolgozásra továbbadja a magasabb szintű rétegeknek. A harmadik és legritkábban használt címzés a multicast. Ilyenkor egy „központi adó műsorsugárzására" kapcsolódik rá nulla, vagy több vevő olyan formán, mint ahogy a rádióadásokat hallgatjuk. Legelterjedtebb felhasználási területei közé tartozik a multimédia adatok terítése vagy egyidejű telepítések megvalósítása *(ghost)*.Kezdeti csomagelemzéseinkben a multicast nemigen fog előfordulni. Első kísérletünk legyen egy PING hálózati forgalmának elemzése. Ha tehetjük, kezdetben tiszta környezetben próbálkozzunk, ahol nincs más hálózati forgalom, mint az,

amit mérni szeretnénk, különben azonnal a filterekkel kell bajlódnunk, ami nem nehéz ugyan, de mégis jobb talán, ha csak annyi csomagunk van, mint amennyire számítottunk. Indítsuk el a Network Monitort, kattintsunk ▶ a gombon, majd pingeljünk meg egy, a belső hálózaton található gépet, ki-ki a saját hálózatán:

PING 172.16.0.2

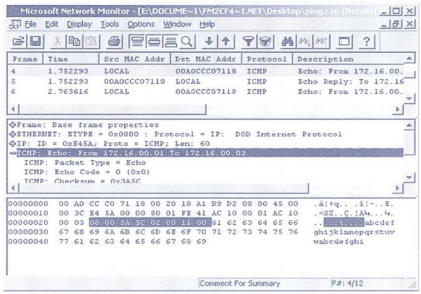
Ha megérkezett a négy válasz, állítsuk meg a felvételt, és tekintsük meg az eredményt a *(megállít+megtekint)* gombbal. Ha valóban csendes volt a hálózat, körülbelül ezt láthatjuk:



A fenti ábra mindjárt meg is válaszolja azt a lappangó kérdést, hogy a PING miért pont négy választ ad. Mert a gépünk négy kérdést küldött ki! Küldhetne többet is, kevesebbet is, sőt a –t kapcsolóval végtelen számút is, a négy egyszerűen a Windows-ok mágikus pingszáma. Már ebben a listában is rendkívül sok érdekességet láthatunk, többek között azt, hogy legalább három protokoll suhant el a hálózaton figyelő tekintetük előtt: BONE, ARP_RARP és ICMP. De lássuk a részleteket.

- A legelső oszlop az elkapott csomag sorszáma, ami még fontos lesz, ha szűréseket végzünk, mert a meg nem jelenített csomagok is megtartják sorszámukat.
- A második oszlop a felvétel megkezdésétől eltelt időt méri trilliszekundumban. Ez a kijelzés megváltoztatható, lásd később!
- A harmadik és negyedik oszlop a feladó illetve a címzett hardvercímét *(vagy boldogabb esetben nevét)* mutatja. Ha nincs meg a név, kísérletezhetünk a Display->Find All Names menüponttal, hogy megfejti-e a kisokos a többi gép nevét. Ha nem, akkor még mindig ott a jobbklatty...
- Az ötödik oszlop a legmagasabb értelmezett protoll nevét tartalmazza. Hangsúlyos, hogy LEGMAGASABB, meg hogy ÉRTELMEZETT, mert először is a NetMon alapértelmezésben nem értelmez minden protokollt (például a Kerberost sem, de felokosítható), másodszor, ha nem értelmez egy magasszintű protokollt, akkor a listában az eggyel alatta lévő réteget jelenítni meg, Kerberos esetén például annyit, hogy TCP-ről van szó – a többi számára is rejtély. A legmagasabb pedig azt jelenti, hogy ha ott ICMP-t olvasunk, akkor is gondoljunk arra, hogy az ICMP-t IP cipeli a hátán, az meg Ethernet keretben utazik.

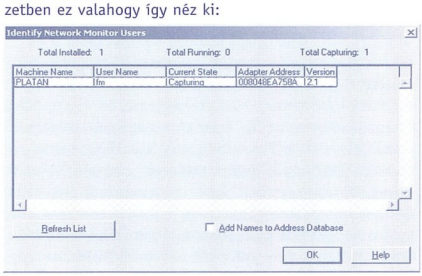
Ha kibontjuk a csomagot, ez azonnal ki is derül, duplaklatty az egyik soron:



Roppant zavaros, egyelőre ne ezen erőlködjünk, hanem nézzük a hatodik oszlopot: ez a LEGMAGASABB, ÉRTELMEZETT protokoll LÉNYEGÉT tartalmazza. Ping esetében ez nem túl informatív, de http-nél nagyon megkönnyíti az olvasást, hogy azonnal látszik a http kérés típusa *(GET, POST stb.)* Most vizsgáljuk meg a három elkapott protokoll szerepét a hálózatban!

BONE

Ezt a protokollt hiába keressük a szabványok lapjai között. Nem szabványos, hanem Microsoftos, és maga a Network Monitor bocsátja ki – tehát jelen esetben önmagunk hálózati forgalmának voltunk szemtanúi. Szerepe abban keresendő, hogy mivel a NetMon igen erőteljes eszköz még így, read only üzemmódban is *(hisz komplett Excelfájlok átküldése is elkapható vele, s abban mondjuk kollegáink fizetése olvasható)* jogos az igény, hogy **mi** megállapíthassuk amikor **más** NetMonozik. Tekintettel arra a szomorú tényre, hogy a távoli hálózati kártyákat nem lehet lekérdezni, hogy promiszkusz módban vannak-e éppen, más módszert eszeltek ki Redmondban – no ez a BONE, amit minden elindított NetMon minden tizedik másodpercben kibocsát, így jelzi a monitorozást a többieknek. Ha a NetMon bocsátja ki, vajon ki olvassa? Úgy van, a NetMon! A Tools->Identify NetMon Users menüpont azért tudja megmondani, hogy kik sniffelnek még rajtunk kívül, mert összegyűjtögeti a csontokat *(BONE=csont)* a hálózatról. Ha nincs meg a menüpont, az azért van, mert csak a NetMon főablakában jelenik meg! Váltsunk át a legelső ablakra a Window menüben, és mindjárt lesz Identify menüpont. Csendes környezetben ez valahogy így néz ki:



ARP_RARP

A következő protokoll az ARP_RARP. Ez már szabványos, sőt, létfontosságú! E nélkül nem létezne unicast címzés!
Miért?
Ha mi pingelünk egy gépet, elvárjuk, hogy az, és csak az válaszoljon a kérésünkre, annak ellenére, hogy – üzenetszórásos csatornáról lévén szó – a csomag Ethernet szinten mindenhova eljut *(a switcheket megint felejtsük el)*. Ez csak úgy lehetséges, ha a MAC address helyesen van kitöltve a PING fejlécében. Igen ám, de mi azt írtuk, hogy

PING 172.16.0.1

...és nem azt, hogy

PING 00-20-18-A1-B9-D2

Pedig ez utóbbi sokkal jobban megjegyezhető. Ki ne tudná fejből a vállalat összes hálókártyájának MAC addressét? *(Én :-)* Gondoljuk végig: mi IP címmel pingelünk, de ezt egyik gép hálózati kártyája sem fogja felismerni, mert az IP cím az operációs rendszer tudománya *(Windowséknál a registryben van, míg hálózati nyomtatók esetében az adott eszköz „operációs rendszerében")* Ebből az következne, hogy gépünk képtelen helyes MAC Adressel kibocsátani a pinget, hisz fogalma sincs a szomszéd masina hardvercíméről. Akkor? Akkor marad(na) a broadcast, hisz kezdetben kizárólag a közismert 0xFFFFFFFFFFFF cím áll rendelkezésre. De gondoljuk végig, micsoda erőforráspazarlás lenne, ha minden gép megkapná és feldolgozná pingünket, majd *(egy kivételével)* nem válaszolna, hisz egy magasabb rétegben végül is kiderül, hogy a csomag nem neki szól! A megoldás kulcsa az ARP, azaz Address Resolution Protocol. Kezdetben, rendszerindítás után valóban kizárólag a broadcast hardvercímről van tudomása minden egyes gépnek. Ha unicast hálózati forgalomra van szükség, be kell szereznie a partner MAC addressét, aminek lekéréséhez broadcast csomagot fog küldeni, valahogy így:

– Fiúklányok! Melyikőtöknek az IP címe a 172.16.0.3?

Ezt minden gép meghallja, a hálókártyák megszakítást keltenek az oprendszer felé, amely felemeli a csomagot, belenéz, és ha magára ismer – válaszol. Visszaküldi a saját MAC addressét. A kérdező pedig feljegyzi a válaszoló hardvercímét az úgynevezett ARP gyorsítótárba. Ennek kilistázása parancssorból:

ARP -a

S már jöhet is a ping, vagy bármi más unicast hálózati forgalom. Ha a történetbe belekeverjük azt a szomorú tényt is, hogy az IP címek idővel megváltozhatnak *(például DHCP címkiosztás miatt)*, akkor beláthatjuk, hogy egy masinának nem célszerű az idők végezetéig megjegyezni a többiek MAC addressét: a nem használt hardvercímek „élete" két perc, de még a használtak is csak tíz percig érvényesek, utána a szabvány szerint újra kell kérni őket. Egyszóval ARP_RARP-pal elég gyakran találkozhatunk hálózati forgalom elemzése közben.

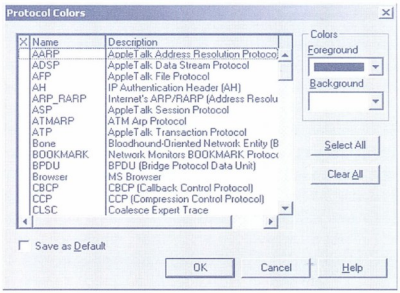
ICMP

A fantasztikus Internet Control Message Protocol rengeteg funkciót lát el, ezek közül csak egy az ICMP Echo, azaz a PING. Egy későbbi számban terveink szerint megjelentetjük az ICMP RFC fordítását mindannyiunk épülésére-szépülésére. A segédprotokoll Echo+Echo reply változata az IP kapcsolat helyességének ellenőrzésére való. Ezt most ne részletezzük, a PING parancsnak ezer kapcsolója van. Étvágygerjesztésként íme még néhány ICMP változat:

- Redirect: routerek küldik ezt megtévelyedett munkaállomásoknak, akik rossz kapun *(gateway)* akarnak kijutni az alhálóról. A Redirect-et a Windows NT 4 SP3 óta figyelembe veszik a MS operációs rendszerei, és szót fogadnak a routernek.
- Time Exceeded: az idő lejárt. Az elküldött IP csomag TTL-je nullára csökkent. Ezt az üzenetet az a router küldi vissza, akinél a csomag élete lejár, „meghal".
- Source Quench: a fuldokló router így közli a túlterhelést okozó munkaállomással, hogy lassítson, mert képtelen átvinni a megadott ütemben a csomagokat – a munkaállomás vagy figyelembe veszi, vagy nem. NT4 óta figyelembe vesszük. Hogy a Windows ME figyelembe veszi-e? Fogalmam sincs.

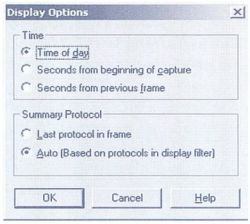
Látványosságok

S most lássuk, hogy mi mindent lehet tenni a látványosság fokozása érdekében. Nem öncélú színezgetésre gondolok, hanem arra az esetre, ha egy hosszabb „beszélgetésből" ki szeretnénk emelni mondjuk az SMTP forgalmat. Térjünk vissza a szemüveg gombbal a részletes nézetre, és színezzünk! Display->Colors



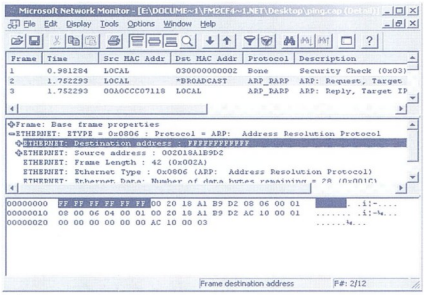
Szerintem önmagáért beszél a felhasználói felület. Keressük ki az ICMP protokollt e végtelen listából, és állítsuk át zöld alapon sárgára. Ne kíméljük az ARP_RARP-ot sem, legyen rózsaszín alapon világoskék! Ugye fáj? Kétszínű kiadvány lévén itt és most nem tudom bemutatni színhelyesen a látványt, így mindenképpen házi feladat a színezgetés.

Másik hasznos lehetőség a kijelzésmód megváltoztatása oly módon, hogy a listában ne a felvétel eleje óta eltelt másodpercek látszódjanak, hanem például a csomagok követési sebessége, vagy az elkapás ideje óra-percben. Ehhez menjünk a Display->Options menüpontra:



Hexa

Itt az ideje, hogy elmerüljünk a részletekben. Kettő katti a legelső ARP soron:



Látható, hogy Ethernet keretben utazik az ARP, s az Ethernetet kinyitva ráálltam a Destination Address sorra, aminek tartalma hexa 0xFFFFFFFFFFFF. Felismerjük? Az első ARP kérés broadcast! Vajon ki a feladó?

```
ETHERNET: Source address : 002018A1B9D2
```

Ahogy a középső ablakon kattintgatunk, az alsó, hexadecimális panelen mindig kijelölődik az a rész, aminek az értelmezett változatát éppen megsimogatjuk az egerünkkel. Így látszik, hogy a keret legelső 6 bájtja alkotja a címzett hardvercímét, a második 6 bájt a feladó címe és így tovább. Ha azonban rákattintunk a Frame Lenght mezőre, a hexa panelen nem jelölődik ki semmi! Ennek oka, hogy a NetMon néha olyanokat is mutat, ami nincs is bent a keretben: ez egy számított mező! A 13.-14. bájt az úgynevezett kerettípus *(frame type)* melynek aktuális értéke *(0x0806)* ARP-t jelent *(IP eseténe ez a mező 0x0800 lenne)*.

És végül minden rétegben megtalálható a „Number of data bytes remaining", azaz az adott réteg számára egyszerűen cipelendő, de nem értelmezendő adattömb. Ha ARP-ot cipel az Ethernet keret, akkor ez további 28 bájt. Ezzel visszaérkeztünk a rétegzett hálózati architektúrához, s jó alkalom kínálkozik az egymásba ágyazás ábrázolására. Minden magasabb réteg az alatta lévővel szabványos interfészen át kommunikál. Ahogy az adatok egyre lejjebb jutnak a protokollveremben, úgy rakódik rájuk egyre több információ, amit a NetMon meg is mutat nekünk. Az IP réteg IP fejlécet tesz hozzá, az Ethernet keretez, a TCP szekvenciaszámokkal operál stb. Erre még részletesen visszatérünk. Az alábbi ábrán egy fiktív, tehát sem TCP/IP, sem OSI modellt láthatunk, ahonnan leolvasható, hogy az egyes rétegek elküldés előtt hogyan egészítik ki az adatot saját fejléceikkel, s fogadáskor hogyan értelmezik és bontják le azt.

Ha most egy ICMP csomagot vizsgálunk meg ebből a szempontból akkor az egyes rétegekben jól látszik, hogy mennyi ott a „Number of data bytes remaining":

```
4 1.752293 LOCAL 00A0CCC07118 ICMP Echo: From
172.16.00.01 To 172.16.00.03 PLATAN 172.16.0.3
IP
Frame: Base frame properties
…
Frame: Frame data:
Number of data bytes remaining = 74 (0x004A)

ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD
Internet Protocol
…
ETHERNET: Ethernet Data:
Number of data bytes remaining = 60 (0x003C)

IP: ID = 0xE45A; Proto = ICMP; Len: 60
…
IP: Data:
Number of data bytes remaining = 40 (0x0028)

ICMP: Echo: From 172.16.00.01 To 172.16.00.03
…
ICMP: Data:
Number of data bytes remaining = 32 (0x0020)
```

Általában minden protokoll valamilyen hasznos adatot cipel, így a PING is. Nála ez 32 bájtnyi „remaining". Házi feladatként mindenki vizsgálja meg, hogy mi a PING által átvitt „hasznos" adat!

Sok sikert!

Fóti Marcell
MCSE, MCT, MCDBA
Folytatjuk...

Post Office Protocol – Version 3

(RFC 1939 – részletek)]
Network Working Group
Request for Comments: 1939
J. Myers, Carnegie Mellon, M. Rose
Dover Beach Consulting, Inc., May 1996

A Post Office Protocol 3-as verziója *(POP3)* segítségével az ügyfelek hozzáférhetnek a kiszolgálón tárolt postaládájukhoz. A POP3-at nem arra tervezték, hogy teljes körű levélkezelést nyújtson – gyakorlatilag csak a levelek letöltéséhez és törléséhez használható. Az ennél több szolgáltatást nyújtó, fejlettebb protokollt, az IMAP4-et az RFC 1730 tárgyalja.

Működés

A kiszolgáló elindítja a POP3 szolgáltatást, amely a 110-es TCP porton várja az ügyfeleket. Amikor létrejön a kapcsolat, a kiszolgáló először üdvözlő szöveget küld. Ezután a kapcsolat befejezéséig parancsokkal és válaszokkal kommunikálnak egymással.
A POP3 parancsok kulcsszavakból és azok esetleges paramétereiből állnak, ezeket kis- és nagybetűvel egyaránt írhatjuk. A parancsok végét CRLF-fel jelezzük. A kulcsszavak és a paraméterek nyomtatható ASCII karaktereket tartalmazhatnak. A kulcsszavakat és a paramétereket 1-1 szóközzel választjuk el egymástól. A kulcsszavak 3 vagy 4 karakter hosszúak. A paraméterek akár 40 karakterből is állhatnak.
A parancsokra adott válaszok egy állapotjelzőből és egy kulcsszóból állnak, melyeket további információ is követhet. A válaszok végét CRLF-fel jelezzük. A válaszok 512 karakter hosszúak lehetnek, beleértve a záró CRLF-et is. Jelenleg 2 állapotjelző van: pozitív *(„+OK")* és negatív *(„-ERR")*. A kiszolgálónak ezeket *(„+OK" és „-ERR")* nagybetűsen kell küldenie. Bizonyos parancsokra többsoros válasz adható. Ezekben az esetekben, melyeket később részletesen is tárgyalunk, a válasz minden sorát CRLF-fel zárjuk, majd utolsóként egy olyan sort küldünk, amely csak egy pontot *(ASCII kódja 46)* és CRLF-et tartalmaz. Ha a többsoros válasz valamelyik sora ponttal kezdődik, akkor a kiszolgáló a sor elejére még egy pontot tesz. A többsoros válasznak ezzel az 5 oktettel kell végződnie: „CRLF.CRLF". Többsoros válasz értelmezésekor az ügyfél először megnézi, hogy az adott sor ponttal kezdődik-e. Ha igen, és ezt a pontot nem CRLF követi, akkor ez a sor eleji pont eldobásra kerül. Ha pedig a sor eleji pontot CRLF követi, akkor az a többsoros üzenet végét jelzi, és az egész sor eldobódik.
Egy POP3 folyamat több szakaszból áll. Amikor létrejön a TCP kapcsolat, és a kiszolgáló elküldi üdvözletét, megkezdődik az AUTHORIZATION *(hitelesítési)* szakasz. Ebben a szakaszban az ügyfélnek azonosítania kell magát a kiszolgáló számára. Ha ez sikeres, a kiszolgáló megnyitja az ügyfél postaládáját és innentől kezdve a TRANSACTION *(lebonyolítási)* szakaszról beszélünk. Ebben a szakaszban az ügyfél kéréseket küld a kiszolgálónak. Amikor kiadja a QUIT parancsot, a folyamat az UPDATE szakaszba lép. Ebben a kiszolgáló felszabadítja a tranzakciós szakaszban lefoglalt erőforrásokat és elköszön, majd bezárja a TCP kapcsolatot.
Az ismeretlen, nem megvalósított vagy szintaktikailag hibás parancsokra a kiszolgálónak negatív állapotjelzővel KELL válaszolnia. Akkor is így kell tennie, ha egy parancsot nem a megfelelő szakaszban kap. Tekintve, hogy csak egyfajta „negatív" állapotjelző van, az ügyfél nem tudja eldönteni, hogy esetleges opcionális parancsát a szerver nem ismeri, vagy nem akarja/tudja végrehajtani.
A POP3 kiszolgálónak lehet időzítője, amely az inaktív ügyfeleket leválasztja. Az időzítés ne legyen rövidebb 10 percnél. Az ügyféltől érkező bármilyen parancs újraindítja az időzítőt. Ha az időzítő lejárna, a folyamat nem lép az UPDATE szakaszba – a kiszolgáló lezárja a TCP kapcsolatot az ügyfél értesítése nélkül.

Az AUTHORIZATION szakasz

A TCP kapcsolat létrejöttekor a kiszolgáló egy rövid, egysoros üdvözletet küld az ügyfélnek. Ez lehet bármilyen pozitív *(azaz „+OK" –val kezdődő)* válasz, például:
+OK POP3 server ready
A POP3 folyamat most az AUTHORIZATION szakaszban van. Az ügyfélnek azonosítania és hitelesítenie kell magát a kiszolgáló előtt. Két módon teheti ezt meg, a USER/PASS parancsokkal, vagy az APOP paranccsal. További hitelesítési módokat az 1734-as számú RFC-ben találunk. *(Az Exchange Server az APOP parancsot nem, viszont az RFC1734-ben definiált AUTH parancsot és rajta keresztül az NTLM azonosítást támogatja – a lektor)*
Az ügyfél eredményes hitelesítése után a kiszolgáló zárolja a postaládát, megakadályozandó a levelek esetleges módosítását vagy törlését, még mielőtt a folyamat az UPDATE szakaszba lépne. Ha sikerül a zárolás, pozitív állapotjelzőt küld az ügyfélnek. A folyamat ezzel a TRANSACTION szakaszba lép. Ha a postaládát nem sikerül megnyitnia *(például nem tudja zárolni, az ügyfélnek nincs jogosultsága a postaládához, vagy a leveleket nem tudja olvasni)*, akkor negatív állapotjelzőt küld. *(Amennyiben a zárolás már sikerült, de valami okból mégis el kell utasítania a parancsot, a kiszolgálónak még a negatív állapotjelző küldése előtt fel kell oldania a postaláda zárolását.)* A negatív visszajelzés után a kiszolgáló bonthatja a kapcsolatot. Ha nem teszi, az ügyfél próbálkozhat újabb hitelesítési paranccsal, vagy jelezheti a kapcsolat végét a QUIT paranccsal.
Miután a kiszolgáló megnyitotta a postaládát, sorszámot rendel a levelekhez és feljegyzi azok méretét. Az első levél az 1-es, a második a 2-es, az n-edik az n-edik sorszámot kapja. A sorszámok és a levelek méretei tízes számrendszerűek.

A TRANSACTION szakasz

Ha az ügyfél azonosította magát, a kiszolgáló pedig sikeresen zárolta és megnyitotta a postaládáját, akkor eljutottunk a lebonyolítási szakaszba. Az ügyfél ebben az alábbi parancsokat adhatja ki, akár többször is. A kiszolgáló minden parancsra válaszol. Ez a szakasz az ügyfél által kiadott QUIT paranccsal zárul, mely után a folyamat az UPDATE szakaszba lép.
A lebonyolítási szakaszban az alábbi parancsok adhatók:
STAT
A kiszolgáló válaszában pozitív állapotjelző után információkat közöl a postaládáról. Hogy könnyen értelmezhető le-

gyen, a válasz kötött formátumú: a pozitív állapotjelző *(„+OK")* után 1 szóköz, az üzenetek száma, újabb szóköz, majd a postaláda mérete oktetben megadva. Végül a sort záró CRLF jön, illetve ez előtt egyes fejlettebb megvalósítású kiszolgálók még további információkat is közölhetnek.
Vegyük figyelembe, hogy a törlésre megjelölt üzenetek nem számítanak az összegzésnél!
Példa: *(ü: ügyfélgép, k: kiszolgáló):*

```
Ü: STAT
K: +OK 2 320
```

LIST [msg]
Paraméter: egy levélsorszám *(opcionális)*, amely nem lehet törlésre megjelölt levélé.
Amennyiben van paraméter, *(és a paraméternek megfelelő sorszámú levél)* a kiszolgáló a megadott sorszámú levélről közöl információt pozitív válaszában. Nem létező sorszám megadása esetén a válasz negatív *(„-ERR")*.
Ha nincs paraméter, de a postaláda üres, a kiszolgáló pozitív állapotjelzőt, majd egy pontot és CRLF-et tartalmazó sort küld.
Ha a postaláda tartalmaz leveleket, akkor a pozitív válasz több soros lesz. A bevezető +OK után a postaládában lévő minden egyes levélről 1 sor információt küld a kiszolgáló. Hogy könnyen értelmezhető legyen, a válasz kötött formátumú: a levél sorszáma után 1 szóköz, majd a levél mérete oktetben megadva *(a levél pontos méretének kiszámítási módját később, „A levél formátuma" c. fejezetben ismertetjük)*. Végül a sort záró CRLF jön, illetve egyes fejlettebb megvalósítású kiszolgálók még további információkat is közölhetnek ez előtt.
Vegyük figyelembe, hogy a törlésre megjelölt üzenetek nem jelennek meg a listában!
Példa:

```
Ü: LIST
K: +OK 2 messages (320 octets)
K: 1 120
K: 2 200
K: .
...
Ü: LIST 2
K: +OK 2 200
...
Ü: LIST 3
K: -ERR no such message, only 2 messages
in maildrop
```

RETR msg
Paraméter: egy levélsorszám (kötelező megadni), amely nem lehet törlésre megjelölt levélé.
Ha a kiszolgáló pozitív választ küld, akkor az többsoros lesz. A kezdő „+OK" után a kiszolgáló elküldi a kért sorszámú levelet.
Példa:

```
Ü: RETR 1
K: +OK 120 octets
K:
K: .
```

DELE msg
Paraméter: egy levélsorszám (kötelező megadni), amely nem lehet törlésre megjelölt levélé.
A POP3 kiszolgáló megjelöli a levelet, mint törlendőt. További parancsokban már nem hivatkozhatunk erre a sorszámra, mert hibaüzenetet kapunk. A POP3 kiszolgáló mindaddig nem törli ténylegesen a levelet, amíg a folyamat az UPDATE szakaszba nem lép.
Példa:

```
Ü: DELE 1
K: +OK message 1 deleted ...
Ü: DELE 2
K: -ERR message 2 already deleted
```

NOOP
A POP3 kiszolgáló nem csinál semmit, csak küld egy pozitív választ. *(Általában az időtúllépés megakadályozására használják – a lektor.)*
RSET
A törlésre megjelölt levelekről eltávolítja a törlésjelzőt, majd a POP3 kiszolgáló küld egy pozitív választ.
Példa:

```
Ü: RSET
K: +OK maildrop has 2 messages (320 octets)
```

Az UPDATE szakasz
Ha az ügyfél kiadja a QUIT parancsot a lebonyolítási szakaszban, a folyamat az UPDATE szakaszba lép. *(Amennyiben a QUIT parancsot a hitelesítési szakaszban adja ki, a folyamat megszakad anélkül, hogy az UPDATE szakaszba lépne.)*
Ha a folyamat az ügyfél által kiadott QUIT parancson kívül bármilyen okból megszakadna, akkor nem lép be az UPDATE szakaszba és nem törölhet egyetlen levelet sem.
QUIT
A POP3 kiszolgáló eltávolítja a törlésre megjelölt leveleket a postaládából, majd a művelet sikerességétől függően válaszol.
Ha a törlés közben valami hiba következik be, lehetséges hogy a törlésre megjelölt levelek közül nem sikerül mindegyiket eltávolítani. Az azonban nem fordulhat elő, hogy olyan levél is letörlődjön, amely erre nem volt megjelölve.
Akár sikerült a törlésre megjelölt levelek eltávolítása, akár nem, a kiszolgáló megszünteti a postaláda zárolását és bezárja a TCP kapcsolatot.
Példa:

```
Ü: QUIT
K: +OK dewey POP3 server signing off (maildrop empty) ...

Ü: QUIT
K: +OK dewey POP3 server signing off (2 messages left) ...
```

A fenti POP3 parancsokat minden POP3 kiszolgálónak támogatnia kell.
Az alábbi opcionális parancsok több lehetőséget biztosítanak az ügyfél számára.
TOP msg n
Paraméter: egy levélsorszám *(kötelező megadni)*, amely nem lehet törlésre megjelölt levélé, valamint a kért sorok száma *(egy nem negatív szám, szintén kötelező)*.
Ha a kiszolgáló pozitív választ ad, akkor az több soros lesz. A kezdő +OK után a kiszolgáló elküldi a levél fejlécét, a fejlécet a törzstől elválasztó üres sort, majd a kért számú sort a levél törzséből. Ha az ügyfél több sort kér, mint amennyi a levél törzsében van, akkor a kiszolgáló az egész levelet elküldi neki.
Példa:

```
Ü: TOP 1 10
K: +OK
K:
K: .
...
Ü: TOP 100 3
K: -ERR no such message
```

UIDL [msg]
Paraméter: egy levélsorszám *(opcionális)*, amely nem lehet törlésre megjelölt levélé.
Amennyiben van paraméter *(és a paraméternek megfelelő sorszámú levél)*, a kiszolgáló a megadott sorszámú levélről közöl információt pozitív válaszában.
Ha nincs paraméter, és a kiszolgáló pozitív választ ad, akkor az többsoros lesz. A bevezető +OK után a postaládában lévő minden egyes levélről 1 sor információt küld a kiszolgáló.
Az egyedi azonosító a kiszolgáló által képzett karaktersorozat, amely hexadecimális 21-től 7E-ig tartalmazhat karaktereket, legalább 1, legfeljebb 70 darabot. Az azonosító nem csak egy folyamat idejéig tartozik a levélhez: ha a folyamat megszakadna még az UPDATE szakasz előtt és újat kezdenénk, a levélnek az új folyamatban is ugyanaz lesz az azonosítója. A kiszolgáló nem adhatja ki az azonosítót újra addig, amíg az eredetileg hozzárendelt levél létezik.
USER name
Paraméter: a postaláda azonosítója *(kötelező megadni)*
Korlátozás: csak a hitelesítési szakaszban, közvetlenül a POP3 üdvözlés, vagy egy sikertelen USER vagy PASS parancs után adható ki.
A USER és PASS parancsok kombinációjával történő hitelesítéshez az ügyfélnek először a USER parancsot kell kiadnia. Ha a kiszolgáló erre pozitív választ ad, akkor az ügyfél folytathatja a hitelesítést a PASS paranccsal, vagy a QUIT paranccsal megszakíthatja a POP3 folyamatot. Ha a kiszolgáló negatív választ ad a USER parancsra, az ügyfél küldhet újabb hitelesítési parancsot, vagy kiadhatja a QUIT parancsot.
A kiszolgáló akkor is adhat pozitív választ, ha a postaláda nem létezik. De létező postaláda esetén is adhat negatív választ, ha nem engedélyezi a titkosítás nélküli jelszavakkal történő hitelesítést.

Példa:

```
Ü: USER frated
K: -ERR sorry, no mailbox for frated here ...
Ü: USER mrose
K: +OK mrose is a real hoopy frood
```

PASS string
Paraméter: a kiszolgálón lévő postaládához tartozó jelszó *(kötelező megadni)*
Korlátozás: csak a hitelesítési szakaszban adható ki, közvetlenül az eredményes USER parancs után.
Amikor az ügyfél kiadja a PASS parancsot, a kiszolgáló a USER és a PASS parancsok paramétereinek alapján eldönti, hogy az ügyfél jogosult-e hozzáférni a kért postaládához. Mivel a PASS parancsnak csak egy paramétere van, használható szóköz is a jelszóban, a kiszolgáló nem fogja paraméter-elválasztóként kezelni.
Példa:

```
Ü: USER mrose
K: +OK mrose is a real hoopy frood
Ü: PASS secret
K: -ERR maildrop already locked ...
Ü: USER mrose
K: +OK mrose is a real hoopy frood
Ü: PASS secret
K: +OK mrose's maildrop has 2 messages (320 octets)
```

APOP name digest
Paraméter: egy a postaládát azonosító sztring, valamint egy MD5 digest string *(mindkettő megadása kötelező)*
Rendszerint minden POP3 folyamat USER/PASS parancsokkal indul. Mivel ilyenkor a jelszavak kódolatlanul haladnak át a hálózaton, ez súlyos biztonsági problémát jelent. Ráadásul a POP3 ügyfélprogramok nagy része új leveleket keresve igen gyakran létesít kapcsolatot a kiszolgálóval, így a jelszavak lehallgatásának esélye igen magas.
Ezért szükség van egy alternatív hitelesítési módra, amely nem küld kódolatlanul jelszavakat a hálózaton. Ezt a funkciót az APOP parancs nyújtja. *(Megj.: az Exchange Server nem támogatja az APOP parancsot, helyette használjunk teljesen titkosított (SSL) POP3 kommunikációt, vagy a cikk elején említett AUTH NTLM azonosítást /ld. RFC1734/. – a lektor.)*

Biztonságos Exchange levelezés 2. rész

A cikk előző részében a levelezés biztonsági kérdéseinek alapjaival, a javasolt telepítés módszereivel és a titkosított levelek küldésével foglalkoztunk. A mostani részben az internetes levelezés és az Outlook Web Access biztonságáról, a mentés és helyreállítás problémáiról, valamint a vírusvédelem lehetőségeiről és az Outlook 2000 SR1 biztonsági újdonságairól lesz szó.

Internetes levelezés - SMTP

Az SMTP protokoll a TCP/IP protokollcsalád internetes levelezést megvalósító, alkalmazásszintű tagja. Egyszerű szöveges parancsok segítségével működik, amelyeket akár mi magunk is lejátszhatunk az SMTP kiszolgálóval egy telnet ablakban. *(E számunkban éppen a POP3 protokollt mutatjuk be a Szabványok rovatunkban, de igérjük, rövidesen az SMTP részletes magyarázatára is sort kerítünk – a szerk.)*

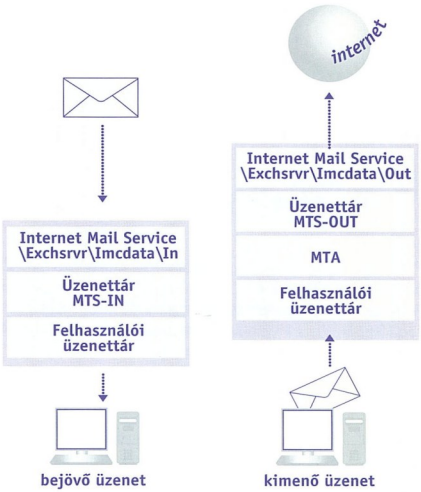
A levelezőkiszolgálókban az SMTP levelezést végző szoftverkomponens általában egy különálló modul, amely önállóan tud döntéseket hozni, természetesen a levelezőkiszolgáló címtárinformációira alapozva. Azaz el tudja dönteni, hogy egy neki küldött levelet a levelezőkiszolgálónak *(vagy más szóval üzenettárnak, ami akár lehet maga a fájlrendszer is)* kell-e átadnia, vagy pedig továbbítania kell valamilyen másik SMTP kiszolgáló felé. Ezt hívjuk SMTP továbbításnak *(relay-nek)*. A továbbítás az Internet hőskorában nem okozott senkinek sem különösebb gondot, hiszen az internetes közösségben mindenki barátként tekintett a másikra, azaz akár mások leveleit is szívesen továbbította, amennyiben azok véletlenül nem a megfelelő SMTP kiszolgálóra tévedtek. Azonban manapság, az Internet üzletiesedésével sajnos már nem engedhetjük meg magunknak, hogy bárkinek a levelét átirányítsuk, mert vannak rossz szándékú emberek, akik csak azért irányíttatják levelüket a mi kiszolgálónkkal, hogy azt leterheljék, vagy azért, mert olyan kéretlen üzleti ajánlatokkal *(spam)* akarják elárasztani a világot, amihez nem szívesen adják a saját IP címüket. Ilyenkor ugyanis az átirányított levél forrása a mi SMTP kiszolgálónk lesz és kéretlen üzleti levelek terjesztése miatt különböző „feketelistákra" tehetik a kiszolgálónkat, ami egyrészt nem túl jó reklám, másrészt nem fogják elfogadni más kiszolgálók az általunk küldött leveleket.

Az SMTP védelem egyik fő célja tehát az, hogy ne tudja bárki szabadon átirányításra felhasználni kiszolgálónkat. Másik fő célja pedig, hogy leveleinket ne tudja bárki elcsípni, elolvasni, módosítani útközben.

Az első cél érdekében azt tudjuk tenni, hogy úgy konfiguráljuk az SMTP kiszolgálónkat, hogy csak a saját levelező rendszerünkben használatos tartománynévre *(a @ utáni címrész)* küldött leveleket fogadja, vagy esetleg olyan egyéb levelező tartományokét is, amelyek számára szintén mi gyűjtjük a leveleket. A második cél érdekében használhatjuk a cikk előző részében ismertetett levéltitkosítást, vagy titkosíthatjuk magát az SMTP kommunikációt is. Sajnos mindkét fajta titkosításhoz egymás bizonyítványainak elismerésére és közös titkosítási módszerre van szükség. A levéltitkosításnál a levelező ügyfélprogramoknak, az SMTP kommunikáció titkosításánál pedig az SMTP kiszolgálóknak kell „közös nyelvet" beszélniük, hogy megértsék egymást. Jelenleg ezeket a technológiákat még nem alkalmazhatjuk általánosan, csak olyan partnerekkel, akikkel ki tudjuk alakítani a hasonló platformot. Egy több telephelyes cég például a részlegei között zajló belső levelezésre használhatja ezeket a módszereket. Az S/MIME és SSL, ASL szabványok alkalmazásának köszönhetően azonban már egyre szélesebb körben alkalmazhatjuk akár a mindennapi internetes levelezésünkben is.

Az Internet Mail Service az Exchange 5.5-ben

Az Exchange 5.5-ben az SMTP levelezést megvalósító komponens az Internet Mail Service. Ezt külön szolgáltatásként telepíthetjük az Exchange kiszolgálóra. Szorosan együttműködik az Exchange üzenettárával *(Information Store)*, egyrészt az internetről érkező és az oda küldendő levelek formátumkonverzióját az Exchange üzenettár végzi, másrészt a levelező szerverünk felhasználójának címzett levelet is az üzenettárnak adja át.



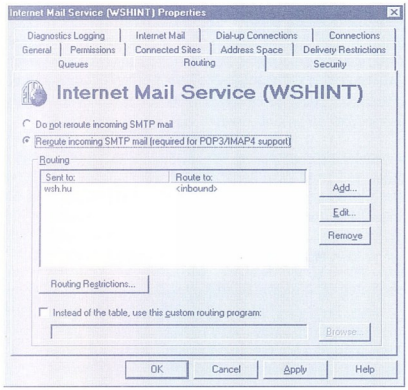
Az Internet Mail Service működése

Az Exchange 5.5 IMS komponens telepítésekor a telepítő varázsló azt kérdezi, hogy mindenki számára engedjük-e az internetes levelek továbbítását, vagy sem. Ha azt válaszoljuk, hogy „nem", akkor csak a saját levelező tartományunknak küldött leveleket fogadja el, az összes többit visszautasítja. Ez látszólag kielégíti az igényeinket, azaz így biztos senki sem fogja illegálisan továbbításra használni a kiszolgálónkat, de van ezzel egy nagyon súlyos probléma. Ugyanis ez a beállítás minden levéltovábbítást elutasít, amire viszont szükségünk lehet, ha POP3 vagy IMAP4 ügyfeleink vannak. Ezek a protokollok csak levél letöltésre alkalmasak, azaz egy kiszolgáló valamilyen levéltárából töltik le a levelet a munkaállomásra. De ezek használatakor is szeretnénk levelet küldeni, amire se a POP3, se az IMAP4 protokoll nem képes. Mi a megoldás akkor? Az, hogy SMTP-t használunk a levél küldésekor, de nem úgy, mint az SMTP kiszolgálók, azaz nem akarnak minden egyes levél küldésekor a cél-SMTP kiszolgálóhoz csatlakozni, hanem csak annyit tesznek, hogy egy továbbításra hajlandó SMTP kiszolgálónak átadják a leveleket, ami aztán a végcélhoz juttatja küldeményüket.

nehézségi fok:

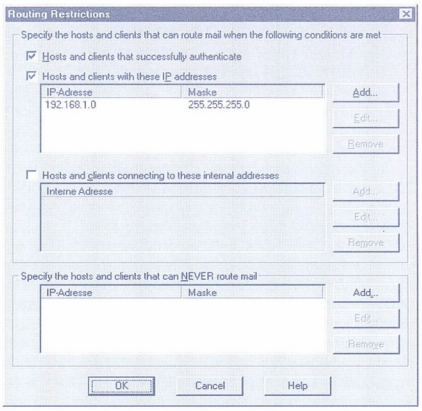
Microsoft BackOffice

A másik beállítás mellett *(Reroute incoming SMTP mail)* az itt felsorolt levelező tartományoknak szóló leveleket fogadja el az IMS, az összes többit - alapértelmezésben - után továbbküldi (relay). Ez kellemetlen, úgy tűnik távolodunk a céltól. Megfelelő beállítás esetén azonban kordában tartható, sőt már akkor hibajelzést ad, amikor a feladó még csak ott tart az IMS-sel történő SMTP párbeszédben, amikor „idegen" tartománynevet ad meg címzettként. Ekkor a feladónak nem levél fog menni, mint az előző esetben, hanem csak egy hibajelzés, ami kevésbé terheli le a kiszolgálónkat, sőt, a rossz szándékú levél adatrésze sem éri már el a szervert. Lássuk a beállításokat!



AZ Internet Mail Service útválasztó beállításai

A „Routing" tulajdonságlapon található „Restrictions" gomb megnyomására megjelenő ablakban kijelölhetjük, hogy az engedélyezett továbbítók kik legyenek: IP címük szerint, felhasználó-azonosítás alapján, vagy aszerint, hogy a több hálózati kártyával rendelkező Exchange kiszolgálónál melyik IP címre csatlakozik az ügyfélprogramot futtató munkaállomás. Ezen lehetőségek megfelelő kombinációjával elérhető, hogy továbbítás legyen is meg nem is, belső IP címekről hajrá, külsőről pedig csak akkor engedélyezze a továbbítást, ha a felhasználó érvényes név+jelszó párossal rendelkezik (successfully authenticate). Ehhez az ügyfélprogramban addig kell kattintgatnunk, amíg rá nem bukkanunk az SMTP autentikációs pipára. (Legjobb, ha az SMTP autentikációt SSL csatornán keresztül végezzük, hogy jelszavaink biztonságosan utazhassanak.)



IMS átirányítási megszorítások, engedélyek

Ezek a beállítási lehetőségek csak az 1. javítócsomagtól kezdődően találhatók meg az Exchange 5.5-ben, ebből is látszik, hogy a biztonság szempontjából is milyen fontos a legújabb javítócsomagok alkalmazása.
Az IMS „Connections" tulajdonságlapján elvileg be lehet állítani általános jellegű felhasználó-azonosítást, de ezzel a lehetőséggel mégsem élhetünk, mivel ekkor nem fog tudni bárki levelet küldeni számunkra, csak azok a kiszolgálók, akik tudják teljesíteni a követelményeket. Mi az értelme akkor ennek a beállításnak? Például az, hogy kiszolgálónkhoz tényleg nem csatlakozhat a világ bármely másik levelező kiszolgálója, csak egy bizonyos úgynevezett „smart host", amely SMTP kiszolgáló védelmi vonalként óvja a mi Exchange gépünket az esetleges támadásoktól. Ez a kialakítás azért jó és biztonságos, mert ezen a „smart host"-on nincsenek üzenettárak, azaz ha egy rossz szándékú behatoló mégis tönkreteszi vagy „feltöri" ezt, attól még a levelekhez nem fér hozzá. A belső Exchange kiszolgálónkat nem érheti el, mert az egy belső IP címen helyezkedik el, a „smart host" másik hálózati kártyájára csatlakozva.

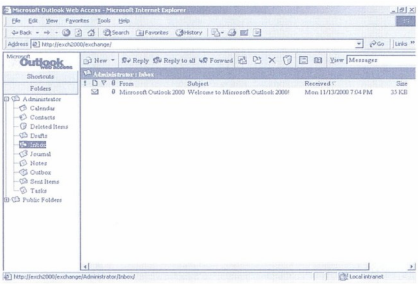
SMTP virtuális kiszolgálók az Exchange 2000-ben

Az Exchange 2000-ben a protokollok átköltöztek az üzenettárból *(POP3 és IMAP4)* és az IMS-ből *(SMTP)* az Internet Information Services komponensbe.
A másik fő változás, hogy az Exchange 2000 rendszerek közti kommunikáció is SMTP alapú lett *(szemben az Exchange 5.5 RPC kommunikációjával)*. Azaz minden Exchange 2000 már alaphelyzetben tartalmaz SMTP kiszolgálót, amire a biztonság *(SMTP átirányítás)* szempontjából oda kell figyelni! Míg az Exchange 5.5 esetében egy számítógépen csak egy SMTP kiszolgálót, azaz IMS-t telepíthettünk, addig az Exchange 2000 esetében tetszőleges számú úgynevezett SMTP virtuális kiszolgálót tudunk létrehozni. Ennek az az előnye, hogy amennyiben különböző beállításokat akarunk a küldők különböző csoportjai számára létrehozni *(például az IMAP-ot, POP3-at használó saját munkatársaink, a kívülről nekünk levelet küldők számára és egy interneten elérhető másik telephelyünkön található Exchange kiszolgáló számára)*, akkor ezt

sokkal egyszerűbben megtehetjük több, a különféle biztonsági igényeket kielégítő virtuális SMTP kiszolgálóval. Az Exchange 5.5-nél ehhez sajnos legtöbbször újabb Exchange 5.5 kiszolgálóra, vagy kiszolgálókra volt szükség.
Természetesen mindazt az SMTP átirányítás-korlátozást használhatjuk Exchange 2000 SMTP kiszolgálók esetén, amit az Exchange 5.5-nél megismertünk, de ezen kívül korlátozhatjuk a továbbítható levelek címzettjeinek számát is. A kéretlen e-mail hirdetések *(spam)* sokszor abból is felismerhetők, hogy a címzett mező rengeteg címzettet tartalmaz.
De térjünk vissza egy pillanatra ahhoz a tényhez, hogy az Exchange 2000 esetében minden kiszolgálón ott van az SMTP felület! Nem jelent-e ez vajon fokozott biztonsági veszélyt? De igen, főleg akkor, ha minden kiszolgálónknak van olyan IP címe, ami az internet felől megcímezhető. Ezért lehetőleg ne használjunk külső IP címet az összes Exchange kiszolgálónkon, hanem csak azon az egy-kettőn, amellyel ténylegesen a külső levelezésünket le akarjuk bonyolítani.
A bejövő leveleket így a külső címmel rendelkező kiszolgálókra tereltük, de hogyan tudjuk a kimenő leveleinket is erre az egy-két kiszolgálóra terelni? Erre való az SMTP Connector, ami az SMTP virtuális szerverek felett logikai réteget képez, és az útválasztási algoritmusukat tudjuk módosítani segítségükkel. Alaphelyzetben minden SMTP virtuális kiszolgáló a DNS rendszer MX bejegyzései segítségével találja meg a cél SMTP kiszolgálót, de egy SMTP Connector esetében kijelölhetünk egy úgynevezett hídfő-kiszolgálót *(bridgehead server)*, ahova az összes kiszolgálónk továbbítani fogja az internetre küldendő leveleket, és nem maguk fogják azokat közvetlenül elküldeni. Ha a bejövő leveleket is ezek a hídfő-kiszolgálók fogadják, akkor elég ezeknek aß kívülről címezhető IP címet adni, és csak ezeket kell az esetleges kívülről jövő támadásoktól fokozottan védeni. Ha a hídfő-kiszolgálónkon nem tárolunk semmilyen felhasználói postafiókot, akkor ezzel meg is valósítjuk a „smart host" szerepű kiszolgálót.

Internetes levelezés – Outlook Web Access

Ha a világban utazgatunk és éppen nem viszünk magunkkal laptopot, akkor is szeretnénk az e-mailjeinket megnézni és leveleket küldeni. Ha máshol nem, de internet kávézókban szinte bárhol hozzáférünk az Internethez és egy böngészőhöz. Levelező programhoz már nem biztos, vagy ha mégis, akkor ezeknek beállítása némi időt és hozzáértést igényel. A legkényelmesebb az lenne, ha a böngészőnkön keresztül, mint egy weboldalt tudnánk letölteni a levelesládánk tartalmát. Erre szolgál az Exchange kiszolgálók Outlook Web Access (OWA) komponense.



Outlook Web Access felület Exchange 2000-ről

Az Exchange 5.5-ben az OWA komponenst egy Internet Information Servert futtató gépre kell telepíteni, amely a böngésző felé dinamikus webtartalmat szolgáltat, ami az adatokat MAPI kapcsolaton keresztül, az Outlookhoz hasonló módszerrel éri el az Exchange 5.5 kiszolgálón.
Az Exchange 2000 esetében szinte az összes internetes protokollt az IIS-be integrálták, így természetesen a közvetlen HTTP felület is minden további nélkül elérhető.
Milyen biztonsági vonzata van a webes felületen történő levelezésnek? Mindenekelőtt az, hogy legáltalánosabban az úgynevezett egyszerű *(basic, nyílt jelszavas)* felhasználó-azonosítást használjuk a webkiszolgálók elérésekor, hiszen nem lehetünk biztosak abban, hogy a fejlettebb azonosítási módszereket minden böngésző ismeri. A nyílt jelszavas bejelentkezés során azonban a hálózati forgalom figyelésével könnyen kideríthetők a felhasználónevekhez tartozó jelszavak, ennek birtokában pedig természetesen illetéktelenek is elérhetik a leveleinket, súlyosabb esetben akár egyéb adatokat is.

A védekezési módszer itt is az, hogy az ügyfélprogram és a kiszolgáló közti kommunikációt a Secure Sockets Layer *(SSL)* réteg alkalmazásával titkosítjuk. Ekkor a kapcsolatfelvételtől kezdődően nem csak a felhasználóazonosítás, hanem a böngésző felé küldött levelek szövegei, sőt, a teljes kommunikáció titkosított, így rögtön két legyet ütünk egy csapásra.

(Meg egy harmadikat is. Nemrégiben ütköztem abba a problémába, hogy az új OWA ugyan a 80-as portról tölti le a levelek tartalmát, de WebDAV-ot használ, ami a HTTP protokoll legújabb verziója. Olyan tűzfalakon nem lehet teljeskörűen „átovázni", amelyek csak a HTTP 1.1-et engedik át. Ez ma már kevés! Ilyenkor csak az OWA kerete jelenik meg (mert az még sima HTTP-vel jön le), de a levelesláda üres marad. Az SSL ezen is segít: minden levél az SSL csatornában jut el hozzám, a tűzfal kotnyeleskedése kizárt!)

Ahhoz, hogy a webkiszolgálónk SSL rétegen keresztül tudjon kommunikálni, kulcspárral és bizonyítvánnyal kell ellátni. A cikk előző részében említettem, hogy ha az Exchange 5.5-nél alkalmazzuk a Key Management Servert az X509.3 szabványú titkosításhoz, akkor a Certificate Serverünket „el kell rontanunk" egy speciális biztonsági modullal; az IIS-ünknek ezzel a Certificate Serverrel ezután már nem fogunk tudni bizonyítványt adni. Azaz telepítenünk kell egy másikat valahova. Az Exchange 2000-nél ilyen problémánk nem lesz.
Az SSL titkosítás viszonylag processzorigényes feladat, ezért jól kell kialakítani azt az Exchange kiszolgálót, amely az üzenettár-adatbázisok kezelése mellett ilyenkor még intenzíven foglalkozik SSL titkosítással. Az Exchange 2000 esetében segítségünkre lehet ebben egy speciális lehetőség, hogy egy újabb Exchange 2000 kiszolgálót teszünk a hálózatba, amit úgynevezett „front-end" kiszolgálóvá alakítunk. Ez egyrészt csak az ügyfélprotokollokkal foglalkozik, üzenettárakkal nem, így több ideje marad az SSL titkosítások elvégzésére, másrészt pedig ez lehet a „támadási felülete" a teljes hálózatunknak, azaz SMTP szempontjából is ez lehet a front-end kiszolgálónk, a „smart host".

Vírusvédelem

A biztonság egyik legkritikusabb pontja a vírusvédelem. A vírusoknak nagyon sok fajtája van, és ezek különböző problémákat okozhatnak. Az egyik nagy csoport egyszerűen tönkreteszi az adatokat, programokat, ami talán a kisebbik baj, ab-

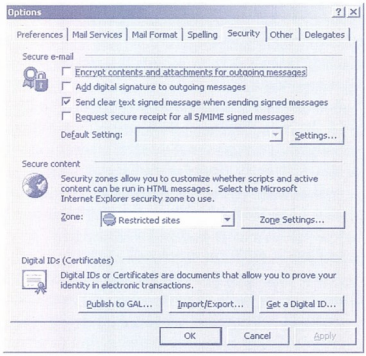
ban az esetben, ha rendszeresen végzünk adatainkról mentéseket. A nagyobb baj az, hogy egy vírus telepíthet a rendszerünkre olyan programokat, amelyek például jelszavakat küldenek a hálózatunkból valamilyen külső gépnek, vagy bármilyen egyéb módszerrel hátsó ajtót nyithat a rendszerünkben, amin keresztül a vírus küldője egyszerűen ki-be járhat, és illetéktelenül adatokhoz férhet hozzá, vagy egyéb károkat okozhat.
Az Exchange rendszerhez a telepítőkészletben nincsen vírusvédelmi lehetőség, ilyen megoldásokat egyéb gyártók kínálnak. Sajnos az Exchange 5.5-ben a hármas javítócsomag előtt nem nagyon volt olyan programozói felület, amire alapozva hatékony vírusvédelmi megoldást tudtak volna fejleszteni, ezért az ezelőtt született programok nem túl hatékonyak. Az SP3 egy olyan antivírus API-t *(VAPI)* hozott be a rendszerbe, amihez hatékony vírusvédelmi alkalmazásokat lehet fejleszteni, így érdemes vírusvédelmi megoldás kiválasztásánál ennek utánanézni, és olyan terméket választani, ami már ezt az API-t használja.

Az Outlook 2000 SR1 újdonságai

Emlékezzünk vissza a LOVE LETTER *(szerelem-)*vírusra! Ennek kapcsán megtapasztalhattuk, hogy milyen leleményesek a vírusíró emberek, és milyen jóhiszeműek az alkalmazói szoftverek készítői. A vírustámadás kapcsán a túl sok és szabadon hozzáférhető funkcióval bíró Outlook levelezőprogramhoz kiadtak több javítócsomagot, amelyeket egy SR-1a javítócsomagban összesítettek, de azóta is jelennek meg újabb javítások. Nézzük meg, hogy milyen újdonságokat hordoznak ezek a frissítések.
Az újdonságok első köre az S/MIME titkosításhoz tartozik, de ezek a funkciók rejtve maradnak, egészen addig, míg egy speciális registry kulcsot fel nem veszünk és abba a megfelelő értéket be nem írjuk:

```
HKEY_LOCAL_MACHINE\Software\Microsoft\
Office\9.0\Outlook\Security\
EnableSRFeatures = 1 (REG_DWORD)
```

Ennek hatására számos S/MIME újdonsággal fogunk találkozni. Egyrészt jóval egyszerűbb lesz a titkosított levelek küldése még Key Management Server alkalmazása nélkül is, mert a Tools/Options/Security lapon már alaphelyzetben kiválaszthatók a titkosítással kapcsolatos beállítások.



Az Outlook SR1a és a registry módosítás utáni biztonsági tulajdonságlap

A frissítés több kényelmi szolgáltatást is nyújt, például ha esetleg sérült biztonsági profilunk van, vagy a bizonyítványunk lejárt, akkor segít ezen problémák megoldásában.
Maga az S/MIME szabvány is fejlődik, így szükséges ezen újdonságok implementálása is. A legfőbb ilyen újdonság a biztonságos nyugta, ami hasonló az olvasási nyugtához, csak itt arról jön vissza értesítés, hogy ellenőrizték a küldött levélhez csatolt elektronikus aláírást. Ez a nyugta tartalmazza az eredeti hash értéket, amit az Outlook összehasonlít az elküldött üzenetek mappájában található eredeti üzenetben tárolt hash értékkel. Ezzel mi magunk is meggyőződhetünk arról, hogy az üzenet nem módosult útközben. Ehhez természetesen egy sokkal biztonságosabb küldő *(From)* mezőre van szükség, mert a hagyományos küldő mező semmilyen titkosítás alá nincsen bevonva. Ezért az S/MIME bevezette a „Signed by" mezőt, amit már bevontak az elektronikus aláírás hash algoritmusába, így azt nyomtalanul nem lehet módosítani. A biztonsági nyugta is erre a címre megy vissza.
Az SR1a és az erre épülő biztonsági javítások több újdonsága a vírusvédelemhez sorolható. Ilyen például az, hogy a végrehajtható kódot tartalmazó e-mail mellékletek vagy teljesen hozzáférhetetlenek lesznek, vagy csak lemezre mentés után nyithatók meg. Másik ilyen jellegű újdonság, hogy ha egy program a címjegyzékekhez akar hozzáférni, mint ahogyan a levélben terjedő vírusok általában teszik, akkor a felhasználó kap egy értesítést erről és neki kell engedélyezni ezt a műveletet, azaz a vírus titokban nem tud terjedni *(legalábbis a címlistán keresztül nem – a szerk.)*.

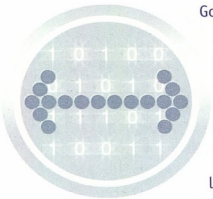
Mentés és helyreállítás

Fontos pár szót szólni az Exchange kiszolgálók mentéséről és helyreállításáról. A beépített mentés funkció egyrészt adatbázisszintű, azaz levelesládákra nem bontható le. Másrészt kifejezetten az adatbázis megsérülése esetére találták ki, és a meghibásodás pillanatáig próbálja az adatbázist visszaállítani. Ezt azért fontos hangsúlyozni, mert nem alkalmas véletlenül törölt elemek visszaállítására, azaz, ha a felhasználó, vagy valamely vírus szabályosan törölt egy levelet, akkor ez a parancs bekerül az Exchange adatbázisának tranzakciós naplójába. Ha visszaállítunk egy régebbi adatbázist, attól még a tranzakciós naplóban a törlési utasítás benne marad, így a régebbi adatbázis-állapot és az azóta született tranzakciós bejegyzések lejátszása - ami a visszaállításnál automatikusan megtörténik -eredményeként szintén nem lesz meg a visszaállítandó elem. Ha a tranzakciós naplót töröljük, akkor pedig az összes mentés óta keletkezett egyéb adat, levél veszik el. Ezért használjunk egyéb archiválási lehetőségeket, amelyek lehetővé teszik az elemek egyenkénti helyreállítását. Az Exchange 5.5-ben ilyen funkció a Message Journaling, az Exchange 2000-ben pedig az adatbázisokra beállítható archiválás. Ezek a technológiák minden adatbázist elhagyó vagy oda bekerülő üzenetről készítenek egy másolatot, ami az üzenet esetleges törlődése után is megmarad és később visszaállítható.

A téma iránt érdeklődőknek ajánljuk a WSH és a NetAcadémia Exchange Serverek biztonságáról szóló tanfolyamát.

Soós Tibor (MCSE+I, MCT)
WSH Oktatóközpont
t.soos@wsh.hu

Terminal Services Licencing



Gondoltam valamire: többen is használhatják egy időben egymástól független célokra, mi az? A címből már sejteni lehet hogy nem egy webkiszolgálóra – bár akár arra is gondolhattam volna. A Terminal Services szolgáltatás lehetővé teszi, hogy egy időben a kiszolgáló erőforrásait több felhasználó egymástól függetlenül használhassa. Ma a Terminal Services licencelésével kapcsolatos információkat tekintjük át a gyakorlatból merített példákon keresztül. Olvashatunk a Terminal Services fontos tulajdonságairól, telepítéséről és használatáról. Megnézzük közelebbről a Terminal Services licencelésének új eszközét, annak jó és rossz tulajdonságait is.

A Terminal Services

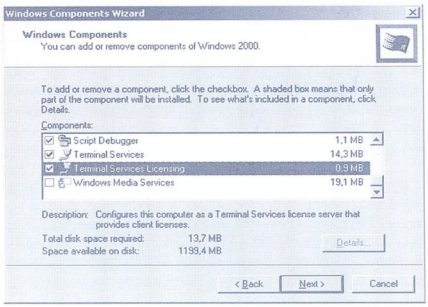
A Terminal Services különbözik minden eddig megszokott eszköztől, amivel a Windows felett az uralmat átvehettük. *(Leszámítva Cytrixet, ami a Terminal Services elődje – ezzel e cikk keretében nem foglalkozunk)*. A Terminal Services szolgáltatás segítségével távolról egy teljesen új felhasználói felületet *(terminált)* nyitunk a kiszolgálón, a konzol előtt ülő felhasználót nem zavarva. Miért fontos ez? A régebbi távfelügyeleti eszközök nagy többsége a számítógép előtt ülő felhasználó képernyőjét, egerét és billentyűzetét veszi át. A Terminal Services használata esetén a konzolt használó felhasználó nem is értesül a többi ügyfél jelenlétéről. A Terminal Services egyaránt alkalmazható alacsony költségvetésű cégeknél *(a régebbi, olcsó kliensgépek kihasználásához)* és a kiszolgálók távfelügyeletéhez is.

A Terminal Services szolgáltatáshoz szükséges feltételek

Terminal Services szolgáltatáshoz rendelkeznünk kell egy Windows 2000 vagy Windows NT terminálkiszolgálóval, egy hálózati kapcsolattal *(ami a TCP/IP mellett gyakorlatilag bármi lehet, pl.: VPN; Modem; LAN/WAN; PPTP)* egy terminál felhasználóval és a szükséges felhasználási engedélyekkel. A terminalkiszolgáló a Windows NT 4.0-hoz is elérhető, ami a Windows NT 4.0 egy speciális változata: Microsoft Windows NT 4.0 Teminal Server Edition. Ugyanazokat nyújtja, mint a Windows 2000 Terminal Services változata, kivétel ez alól a Windows 2000 néhány speciális újdonsága. *(Két különböző technológiáról van szó egyébként, a közös bennük csak a terminálszolgáltatás, mint funkció.)* A Windows 2000 kiszolgálócsalád tartalmazza a Terminal Services és a Terminal Services Licensing szolgáltatást is. Ezek komponensként telepíthetők akár a kiszolgáló telepítésekor és később, a már üzembe helyezett kiszolgálóra is *(ilyenkor a Control Panel Add/Remove Programs moduljában az Add/Remove Windows Components oldalon kitt-katt)*.

Terminal Services Licensing *(TSL)*

A Windows NT 4.0 Terminal Services Edition változatban nem volt TSL funkció, annak licencelése teljes mértékben az NT licencelésén alapult. Ez néhol jó, néhol kevésbé jó megoldásnak bizonyult.



Terminálszolgáltatás és a licencelés telepítése

A lusta rendszergazdának, akinek a TSL első és második ránézésre is csak egy újabb felesleges „kütyünek" tűnik, biztos kellemetlen élményei lesznek. Még az érdeklődő szakemberek körében is akadhat olyan, akinek egy kicsit kellemetlennek tűnhet a TSL használata. Bevallom, én is ezek közé sorolom magam – lapzártakor a véleményem egy kicsit jó irányba változott. De miért lehet kellemetlen élményünk? Egy gyermekcipőben járó technológiát látunk: ez az automatikus licencelés. Az ötlet jó, de még csiszolni kell. Remélem, hogy a következő operációs rendszerben a kényelmetlen pontok is tökéletesednek.

Ha röviden szeretném jellemezni, a TSL egy adatbázis, amiből az arra jogosult számítógépek licenceket vehetnek ki. Ezeket használhatják később a Terminal Services legális eléréséhez. Nagy előnye/hátránya *(ki-ki döntse el maga :-)* a módszernek, hogy azok és csak azok férhetnek hozzá a terminálkiszolgálóhoz, akik rendelkeznek a megfelelő licenccel, így ad-hoc „túlhajtásra" nincs lehetőség. A TSL csak a licencek kiosztásával foglalkozik, licencek beszerzésére nem használható, és nem alkalmas azok visszavonására sem. Sajnos. :-(

A TSL működése

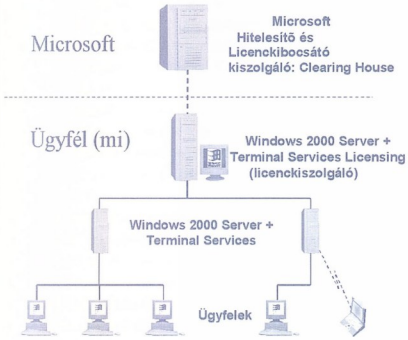
A TSL kiszolgáló telepítésére Application Mode-ban telepített terminálkiszolgáló esetén van szükség. Hogy a terminálszolgáltatás módjait megértsük, két fontos fogalmat kell megtanulnunk:

- **Administrative Mode *(rendszerfelügyeleti mód)*:** Lehetővé teszi a Windows 2000 operációs rendszer távoli felügyeletét, ezzel is csökkentve a rendszer adminisztrációs költségeit. Ekkor a Terminal kiszolgálóra csak a rendszergazdák jelentkezhetnek be. Egy időben legfeljebb 2 kapcsolat lehetséges, figyelembe véve az aktív és inaktív kapcsolatokat egyaránt. Nem igényli a TSL üzembeállítását és TS CAL-t sem kell vásárolnunk. *(CAL=Client Access Licence, azaz ügyfélhozzáférési licenc – a szerk)*.
- **Application Mode *(alkalmazásmód)*:** Nem korlátozza az egyidejű kapcsolatok számát. A szolgáltatást egyszerű felhasználók is igénybe vehetik, programokat futtathatnak a kiszolgálón, mintha az a saját számítógépükön futna. Természetesen minden, a kiszolgálón futó alkal-

nehézségi fok:

mazás csak a kiszolgáló erőforrásait veszi igénybe. Használatához a TSL szolgáltatás telepítésére és aktiválására, a felhasználók számára pedig a megfelelő TS CAL beszerzésére van szükség.

Amikor egy alkalmazásmódban működő terminálkiszolgálóhoz csatlakozunk, az egy TSL kiszolgálót keres, majd igényli a megfelelő licenceket.



A TSL kiszolgáló működésének elvi rajza

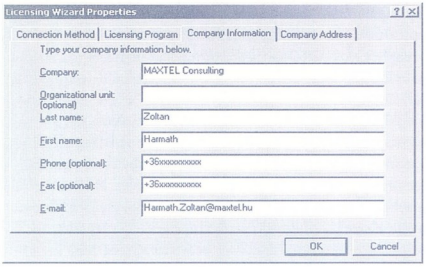
Az ügyfélgép vagy megkapja a szükséges licencet, vagy a kiszolgáló elutasítja az igénylést. Elutasítás abban az esetben lehetséges ha a TSL kiszolgáló adatbázisában már nem található kiosztható licenc. Közvetlenül elutasítást a TSL kiszolgálótól nem kapunk – ez alól kivétel az az eset, ha nincs kiosztható licenc – ami ha meggondoljuk, nem szerencsés, hisz így sajnos akár egy kóbor vándor felhasználó is „lenyúlhat" egy licencet, ami ezzel az ő gépére költözik, s onnan visszavenni nem lehet. Ez azt jelenti, hogy nem tudjuk megmondani a kiszolgálónak, kitől fogadhat el licenckérelmet és kitől nem. Ezért a terminálkiszolgálóink biztonságát a lehető legmagasabb szintre kell emelnünk: ha az ügyfelek nem érhetik el a kiszolgálót akkor nem kaphatnak arra a kiszolgálóra érvényes licencet. Miért beszélünk többes számban a terminálkiszolgálók esetében? Egy TSL kiszolgáló több terminálkiszolgálót is elláthat a megfelelő számú ügyfélhozzáférési tanúsítvánnyal *(licenc)*. Ebben az esetben –ha egy TSL több terminálkiszolgálóért felelős – figyelembe kell venni a felderítési lehetőségeket is. A terminálkiszolgáló induláskor keres egy TSL kiszolgálót. A tartományban érvényes tartományvezérlők után – ha ez szükséges – az Active Directory-ban is keres a TSL kiszolgálók után. Amennyiben ez sem jár sikerrel, broadcast üzenetszórással próbálja felderíteni a legközelebbi TSL kiszolgálót. Ezt a terminálkiszolgáló 15 percenként megismétli függetlenül attól, hogy talált TSL kiszolgálót, vagy nem, a talált TSL kiszolgálókat pedig felveszi a listájába. Ez Microsoft Windows NT 4.0-s tartomány esetén egy broadcast üzenetet, Active Directory tartomány esetén pedig a DDNS kiszolgáló adatbázisának lekérdezését jelenti. Pontosítsuk egy kicsit: Windows 2000-es tartományok esetén a TSL kiszolgálót egy tartományvezérlőre kell telepítenünk. Az alkalmazásmódban telepített terminálkiszolgáló lekérdezi a tartomány összes tartományvezérlőjét, majd ezek után megállapítja, hogy ezek közül mely kiszolgálók TSL kiszolgálók is egyben. Ezután csatlakozik a hozzá legközelebb eső kiszolgálóhoz *(a „legközelebbi" kiszolgálót az Active Directory telephelyek alapján találja meg)*. Ez az Enterprise Licensing alapja. Windows NT 4.0-s tartományok esetében a TSL kiszolgáló felderítése broadcast üzenettel történik.
A TSL kiszolgáló telepítésével feladatunk még nem ért véget, hisz most aktiválnunk kell, majd a Client Licence Key Pack-et kell telepítenünk.

TSL kiszolgáló aktiválásának lépései:

A Terminal Services Licensing kezelőprogram indításakor felderíti az elérhető TSL kiszolgálókat. Ha a hálózatunkban több TSL kiszolgáló is működik, akkor valószínűleg megtalálja mindent. Ha nem, akkor kezdődhet a hibakeresés, aminél figyelembe kell venni a TSL kiszolgálók felderíthetőségét *(DDNS; Broadcast)*. Az itt példaként használt tartományban csak egy TSL kiszolgáló van aktiválva tehát a képeken csak egy kiszolgáló lesz látható. Az indítás után szükség esetén további kiszolgálók is felvehetők, de ehhez a megfelelő jogosultsággal is rendelkeznünk kell. Miután elindítottuk a TSL kezelőprogramot, láthatjuk a kiszolgálónkat, melynek bal alsó sarkában egy kis piros jel mutatja számunkra: a kiszolgálónk nincs aktiválva. Ahhoz, hogy a kiszolgálót aktiválhassuk, el kell döntenünk, hogy az aktiváláshoz szükséges információkhoz milyen úton szeretnénk hozzájutni:

- **Internet:** Közvetlen, titkosított hálózati kapcsolat
- **World Wide Web:** titkosított kapcsolódás a Microsoft Terminal Services webhelyhez
- **Fax**
- **Telefon**

Az aktiválás varázslót a kiszolgáló nevére jobb gombbal kattintva csalhatjuk elő *(Activate Server)*. Itt az első fülön lehet a kapcsolat módját kiválasztani. Fax és telefon választása esetén további információkat is kér a rendszer, például az országot ahol vagyunk. Magyarországon nincs ilyen ügyfélszolgálat, ezért nemzetközi hívást kell kezdeményeznünk, aminek a minősége sok esetben rossz, sőt nagyon rossz. :-(
A választásunknak megfelelő módon tehát közölnünk kell a Microsoft-tal a szükséges adatokat. *(A kép csak tájékoztató jellegű, egy aktivált kiszolgáló adatlapját ábrázolja)* Meg kell adnunk a cég adatait, egy kapcsolattartó személy nevét és e-mail címét. Ez utóbbi adat nagyon fontos nemcsak nekünk, de úgy tűnik a kibocsátó szervnek is, mert egy felugró ablakban az e-mail címünket ismét kéri a rendszer.



Terminálkiszolgáló adatlapja

Az adatlap hiánytalan kitöltése után elkezdődhet a kiszolgáló aktiválásának folyamata, ami nem vesz igénybe sok időt és nagy sávszélességet. Akár egy 33.6-s modem sebessége is megfelel. *(Tapasztaltam már olyat, hogy a modem nem tudta felvenni a kapcsolatot a Microsoft-tal, de abban az esetben a rendelkezésre álló sávszélesség közelített a nullához).* Az aktiválás természetesen más módszerekkel is történhet. A cél a TSL ID megszerzése! Alapvető különbség csak a kapcsolat létrehozásában és a használt felületben keresendő. Mi történik az aktiválási folyamat során? Nagyon fontos lépés ez a TSL kiszolgálónk életében. Adataink a Microsoft Clearing House számítógépéhez kerülnek és ott kapunk egy tanúsítványt. Ezzel lesz majd képes a kiszolgálónk a Microsoft-tal történő titkosított adatforgalom lebonyolítására. Egy X509-es tanúsítványt *(Certificate)* kapunk vissza, amit a kiszolgálónk eltárol. Önmagában ez még nem lenne elég, de az X509-es tanúsítványunk csak a mi gépünkhöz jó és a telepített Windows 2000 kiszolgálónk sorozatszámával együtt érvényes csak. *(Miért jó ez? ld. 2000. November: Hálózatunk biztonsága)* Kicsit elébe vágtunk a dolgoknak. A TSL kiszolgálónak szánt gépünk felveszi a kapcsolatot a Microsoft erre hitelesített gépével. Ezek után 3 lehetőségünk van:

- A megfelelő kód birtokában befejezzük az aktiválási folyamatot.
- A megfelelő kódra várva az aktiválási folyamatot később fejezzük be.
- Az aktiválási folyamatot visszavonjuk.

Az első esetet akkor válasszuk, ha a Microsoft-tól a megfelelő visszajelzést megkaptuk. Esetünkben a visszajelző levélre várunk. A harmadik lehetőség pedig bármely hiba esetén megfelelő visszalépést biztosít számunkra. Rövid időn belül meg kell kapnunk a visszaigazoló levelet a megadott email címre. Az elektronikus levél tartalmaz egy aktiválási kódot. Az imént elvégzett módszerrel az aktiválási folyamatot el kell indítanunk, majd ott a második lehetőséget kiválasztani. Ekkor a 35 karakteres alfanumerikus karaktersorozatot kell begépelnünk *(ezt kaptuk meg elektronikus levél formában valószínűleg anyanyelvünkön)*. A rendszer burkoltan figyelmeztet arra, hogy ez a karaktersorozat csak a meghatározott sorszámú telepített Windows 2000 kiszolgálóval érvényes. Miután ezt begépeltük, a rendszer ismét felveszi a kapcsolatot a Microsoft erre hitelesített kiszolgálójával egy titkosított csatornán keresztül. Ezzel számítógépünket felkészítettük arra, hogy a Microsoft gépével közös biztonsági csatornán kommunikáljon a jövőben. Mit vehettünk észre? A csomag *(e-mail)* ami tartalmazza a megfelelő jelszót, átment a hálózaton. Van egy közös adat *(a kiszolgáló sorozatszáma)*, amelyet a mi gépünk és a Microsoft kiszolgálója is ismer, és amit a megfelelő HASH használatával átalakítva azonos eredményt kell kapniuk. Klasszikus autentikációról beszélhetünk, ami sokban hasonlít a Windows NT hálózatokban már megszokott NTLM és NTLMv2-höz. A különbség csak a magasabb titkosítás, mint pl. az NT esetében az NTLMv2 lehet. Ennyit a TSL kapcsolatainak biztonságosságáról, most inkább nézzük meg, hogy mi történt akkor, amikor aktiváltuk a kiszolgálónkat. Az egyik és legfontosabb, hogy a kiszolgálónk ezentúl bármikor képes felvenni a kapcsolatot a Microsoft-tal a megbízotti kapcsolat létrehozásának köszönhetően. A másik nagyon fontos funkció, hogy ezután az alkalmazásmódban telepített kiszolgálónk is működni fog, mert létezik a hálózaton egy kiszolgáló aki a tanúsítványok kiadását elvégzi. Most a figyelmes olvasó egy kicsit elcsodálkozik: hogy lehetséges ez? Nem is kell CAL? De hisz az imént azt mondtam, hogy két dolog szükséges a boldogsághoz: egy aktivált TSL kiszolgáló *(megvan)* és sok-sok telepített CAL *(Client Licence Key Pack)* – nos ez utóbbit eddig még nem telepítettük. Ennek ellenére a TSL kiszolgáló aktiválása után minden kérést teljesít – ideiglenes, 90 napos tanúsítványt oszt ki a kérelmezők részére! A CAL-ok telepítése nélkül az alkalmazásmódban telepített terminálkiszolgálót legfeljebb 90 napon keresztül érhetjük el *(bővebben később)*.

Client Licence Key Pack telepítése:

A Client Licence Key telepítésével kiszolgálónk alkalmassá válik a szükséges és rendelkezésre álló tanúsítványok kiosztására. A telepítés során szükségünk lesz a CAL ID-re amit a CAL beszerzésével azonos időben az eladótól meg kellett kapnunk. Ez egy 35 karakteres alfanumerikus kód. A kód megadása után a CAL-ok elérhetők, kioszthatók. Amennyiben ideiglenes CAL-ok is kiosztásra kerültek, azok tulajdonosai a következő csatlakozáskor a most telepített CAL-ból kapnak egyet. MCSP cégek most ne kezdjék el keresni a CAL ID-t. Ez egy kicsit komplikáltabb, később szólok erről is.

CAL-ok telepítése:

A TS CAL-ok a megszokott csatornákon keresztül szerezhetők be. A TSL kiszolgáló alapvetően két különböző licencelési formátumot támogat:

- **TS CAL:** Ezt telepítve meghatározott mennyiségű licenc áll rendelkezésünkre. A licencek kiosztásával az új felhasználók részére biztosítható licencek száma csökken. Gyakorlatilag mindenki választhatja ezt a módszert.
- **Terminal Service Internet Connector:** lehetővé teszi, hogy a terminálkiszolgáló akár 200 egyidejű kapcsolatot is kiszolgáljon anélkül, hogy TS CAL-okat kelljen megvásárolni. Ennek használatára csak a Select vásárlók jogosultak.

A licencek beszerzése után azokat a Licensing varázsló segítségével telepíthetjük. Ennek során a már ismertetett módok valamelyikéből *(telefon; fax; web; Internet)* kell választanunk. Fontos kérdés lehet az, hogy milyen beszerzésből származik a licenc, amit telepíteni szeretnénk. *(Tehát tudnunk kell azt, hogy Open Licence, Select, vagy Other kategóriába tartozunk.)* A választásunknak megfelelően a rendszer adatokat kér a szerződéssel kapcsolatosan. A szükséges adatok megadása után a telepítés végére érünk, ezzel a licencek kioszthatóvá válnak.

Solution Providerek esetén ez kicsit nehezebb feladat. A szerződés megkötésének pillanatában elérhetővé válik 20 darab TS CAL, amit fel lehet használni. Ilyenkor a licencelésnél csak és kizárólag a telefonos változatot használhatjuk *(ez csak a CAL telepítésére vonatkozik)*, ahol majd a megfelelő ID-t megkapva a telepítés befejezhető. Ilyenkor szükségünk lesz az MCSP ID-re is.

CAL-ok élete, azok „védelme":

Egy újonnan telepített Terminal kiszolgáló *(Pear Seat)* alkalmazásmódban használva megfelelően működik TSL és CAL-ok nélkül is. Ez az átmeneti idő 90 napot jelent. A 90-ik napon a Terminal kiszolgáló minden hozzá érkező kérést a 'Server too busy' hibaüzenettel elutasít. A felhasználók hozzáférési tanúsítványa nélkül ezután nem működik. Ebben az esetben csak egy lehetőségünk van *(jogi és logikai szempontokat vizsgálva)*: egy TSL kiszolgálót kell üzembe helyeznünk, ami CAL-okat fog osztani a felhasználóink számára *(vásárolt vagy ideiglenes CAL-t)*. A TSL kiszolgálót a fent olvasott módon telepíthetjük és aktiválhatjuk. A kiszolgálónk ettől a lépéstől kezdve képes a Terminál kiszolgáló kérését megfelelően ellátni. Innentől minden kérésre ideiglenes, legfeljebb 90 napig érvényes tanúsítványt bocsát ki, ez érvényes a Windows 2000 operációs rendszercsaládot futtató felhasználói gépekre is. Ismét nyertünk 90 napot :), de ekkor már égető lehet a probléma, ezért nem marad más, mint a TSL CAL-ok beszerzése, azok telepítése és megfelelő védelme. A CAL-ok telepítése után a kiszolgáló minden kérésre egy CAL kiosztásával válaszol, ha annak nem látja akadályát. Ha a kérés egy szabályosan telepített terminálkiszolgálótól érkezett, azt akkor és csak akkor utasítja el, ha nincs kiosztható CAL. Ez aggasztó lehet, mert fűnek-fának elosztogatja a CAL-okat! Nagyon meg kell gondolnunk tetteinket! Mit lehet tenni a CAL-ok fogyása ellen? Logikus lenne, ha a TSL kiszolgálónkat kellene védeni, de nem. A terminálkiszolgálót magát kell védeni, mert a TS CAL igénylését indítja el. Ezért a következők valamelyikével védhetjük a CAL-okat:

- A Logon locally *(Helyi bejelentkezés)* jog erős, nagyon erős szabályozása a terminálkiszolgálón.
- A terminálelérés korlátozása a terminálkiszolgálón.
- IPSec.
- Access this computer from network *(Számítógép elérése hálózaton keresztül)* jogosultságok korlátozása.

Ezek közül valóban fontos és hasznos beállítás a Logon locally jogosultsággal rendelkező felhasználók számának csökkentése. Mivel a CAL igénylését egy sikeres terminálbejelentkezés indítja el, ezért ezzel megfelelő módon védhetjük a kiszolgálónkat *(Mi a helyzet a roaming userekkel? A szerk.)*. További előnye az, hogy nem csökkenti a kiszolgáló funkcionalitását, mint az az IPSec és az Access this computer from network jog megvonása esetén tapasztalható. A CAL-ok fogyásának ez volt az első példája, de nagyon fontos, hogyha egy felhasználói gépet újratelepítünk, az a kiosztott CAL-t elveszíti. Mi ennek az oka? A TSL kiszolgálótól kapott ügyfél-hozzáférési engedélyt az ügyfél számítógépe a helyi rendszerleíró adatbázisban tárolja. Újratelepítés esetén a rendszerleíró adatbázis bejegyzései eltűnnek, tehát a licenc is! Ekkor fordulhat elő az az eset, hogy két azonos nevű számítógép két darab ügyfél-hozzáférési tanúsítványt kap. Mit tehetünk ilyenkor? Fel kell vennünk a kapcsolatot a Microsoft ügyfélszolgálatával, amihez ismét elő kell keresnünk a CAL ID-t *(MCSP cég esetén az MCSP ID-t)* és persze egy telefont, amiről Írországgal is tudunk beszélni. *(Sajnos hazánkban egyelőre nem vásárolható közvetlen írországi forródrót – a szerk.)*

És végül egy hasznos eszköz:

Windows 2000 Resource Kit-ben található egy eszköz, amellyel elemezhetjük a TSL kiszolgálónk adatbázisát. A parancssori eszköz neve Lsreport.exe. A program az adatbázisból tabulátor karakterekkel elválasztott szövegfájlba írja a kért adatokat. A következő táblázatban az eszköz parancssori paraméterei láthatók:

/F fájlnévA program kimenetét a „fájlnév"-nevű fájlba irányítja. *(Az alapértelmezett fájlnév az „lsreport.txt")*

/D kezdődátum [végdátum] Csak azokat a licenceket írja ki, melyek a kezdődátum és a végdátum között kerültek kibocsátásra. *(A végdátum alapértelmezése az aktuális dátum)*

/T Csak az ideiglenes licenceket listázza.

Kiszolgálólista Azoknak a kiszolgálóknak a listája, ahonnan a program lekérdezi az információt. Ha nem adjuk meg, a forrás a tartományvezérlő

/? A program használatáról ad útmutatót.

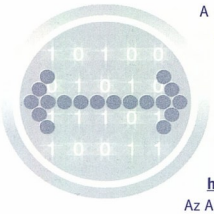
Harmath Zoltán
MCSE, MCP+I
Zolee@geniusgroup.hu

Zárszó

Töprengtünk a szerkesztőségben, hogy vajon miért kellett a Terminal Services használatát ennyire megnehezítő licenceléssel piacra dobni. A magyarázat szerintem abban keresendő, hogy a TS önmagában olyan „csábos", hogy emiatt az egy szolgáltatás miatt esetleg az emberek hajlamosak lennének mondjuk Windows 2000 Professionalt VENNI, de Advanced Servert HASZNÁLNI. Korábban – valljuk be – ez a lehetőség nem volt annyira csábító, mert az NT4 esetén egy üres Server nem sok csábos eltérő szolgáltatással rendelkezett egy Workstationhoz képest.

fm

Component Load Balancing



A Microsoft Application Center 2000 *(továbbiakban Application Center)* a Microsoft .NET kezdeményezés alapját képező Enterprise Server termékcsalád része. A termékcsaládot ezen kívül többek között a Commerce Server 2000, a BizTalk Server 2000 és az SQL Server 2000 képezik – további információk a Microsoft .NET weboldalán találhatók:

http://www.microsoft.com/net

Az Application Center célja, hogy megkönnyítse a Windows 2000- és Internet Information Service 5.0-alapú webkiszolgálók telepítését és kezelését. Az Application Center segítségével felügyelt webhelyek méretezhetők, robusztusak, biztonságosak és könnyen kezelhetők. A webkiszolgálók fürtbe rendezésének fő célja az, hogy azok az ügyfél számára továbbra is csak egy kiszolgálónak látsszanak. A látszat fenntartásához az kell, hogy a webhely teljes tartalma a fürt minden tagkiszolgálóján egyaránt és egyformán jelen legyen; a tartalom alatt értjük a webhelyet képező statikus információkat, fájlokat, regisztrációs adatbázis-kulcsokat, COM+ komponenseket, stb. A fürt „egészsége", működési paraméterei a teljesítmény-számlálók, felügyeleti eszközök és programozható események segítségével könnyen felügyelhető.

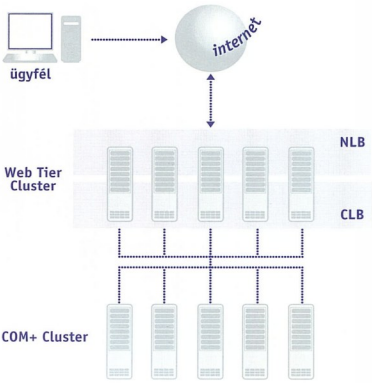
Application Center fürtök

Az Application Center felhasználásával többrétegű fürtöket telepíthetünk, ahol a különböző rétegek különféle terheléselosztási megoldásokon alapulnak. A terheléselosztásnak köszönhető az Application Center méretezhetősége és robusztussága. Kétfajta terheléselosztásról van tehát szó, ezek:

- hálózati terheléselosztás *(Network Load Balancing, NLB)*, és
- komponens-terheléselosztás *(Component Load Balancing, CLB)*

Az Application Center fürt felépítése meghatározza, milyen terheléselosztást alkalmazhatunk az adott helyzetben. Általában többrétegű *(n-tier)* környezetben dolgozunk, ahol a terheléselosztás a webkiszolgálásánál jelentkezik. Az ilyenkor használt hálózati terheléselosztás *(NLB)* feladata a beérkező IP kérések gyors és folyamatos kiszolgálása.

A webrétegen működő szoftver *(például ASP oldalak)* COM+ komponenseket használhatnak; a komponensek az adott számítógépen, vagy a Distributed COM-nak köszönhetően egy távoli gépen futnak. Komponens-terheléselosztás (CLB) nélkül ezek a távolból létrehozott objektumok statikusak, nem „foglalkoznak" az azokat futtató gép pillanatnyi terhelésével. A CLB célja, hogy a hálózati terheléselosztáshoz hasonlóan a COM+ objektumok futtatásának terhét is megossza a fürtben működő tagkiszolgálók között. *(ez a fürt nem azonos az NLB fürttel, ld. az ábrát – a szerk.)*



Tipikus Application Center fürt-felépítés

Hálózati terheléselosztás

A hálózati terheléselosztás *(NLB)* a Windows 2000 Advanced Server és a Windows 2000 Datacenter Server szolgáltatása. Az NLB a beérkező TCP, UDP és GRE hálózati terhelést elosztja a fürt tagkiszolgálói között. Az elosztás alapja egy, a kiszolgálók előzetes terhelési beállításain alapuló, statisztikai algoritmus. Az NLB fürt dinamikusan méretezhető, automatikusan alkalmazkodik a tagkiszolgálók kieséséhez és új tagok munkába állításához *(általában anélkül, hogy ebből az ügyfelek bármit is észrevennének)*; a rendszer képes észlelni a tagok működési hibáit, és automatikusan kizárni őket a szolgáltatásból.

Az Application Center leegyszerűsíti az NLB fürtök kezelését. A Cluster Creation Wizard automatikusan elvégzi az NLB beállítását, lényegesen egyszerűbben, mintha azt a Windows 2000-ben nekünk saját kézzel kellene megtennünk. Az Application Center segítségével könnyen új tagokat adhatunk a fürthöz, távolíthatunk el onnan, illetve helyezhetünk üzemen kívül. A hálózati terheléselosztással kapcsolatos további hasznos információk a

http://www.microsoft.com/windows2000/library/howitworks/cluster/nlb.asp címen találhatók.

Komponens-terheléselosztás

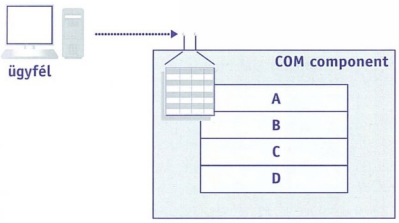
A komponens-terheléselosztás *(CLB)* feladata a COM+ komponensek által okozott terhelés elosztása. A COM+ komponensek kész szoftverelemek, programozható, különféle programozási nyelvekből *(például VBScript, ASP, Visual Basic, C++)* egyaránt elérhető objektumok. Ezek a komponensek képezik a szoftverek újrafelhasználható építőkockáit. CLB esetén a komponensek egy különálló COM+ kiszolgálófürtön futnak. Ha az alkalmazás egy COM+ objektumot szeretne használni, a CLB azt a COM+ fürt valamelyik tagkiszolgálóján hozza létre. Amint az első ábrán látható, az objektumot felhasználó üzleti logika a webrétegen működik, a CLB csak a létrehozott COM+ objektumokat „szórja szét" a CLB tagkiszolgálók között.

nehézségi fok:

Component Object Model

A CLB alapja a Component Object Model, más néven COM. A COM-kompatíbilis objektumok egy általános felületen keresztül képesek az általuk megvalósított szolgáltatások „közzétételére". A megírt COM objektum különböző COM-kompatíbilis programozási nyelveken és operációs rendszerekben módosítás nélkül használható. A rugalmasság kulcsa a COM programozói felület *(COM interfész)*.

A COM komponensek szolgáltatásai az objektum egy vagy több interfészén keresztül érhetők el (ezen interfészek egyike az, amely a többi interfész lekérdezésére való). A komponens használatához olyan programozási nyelvre és környezetre van szükség, amely képes kezelni ezeket az interfészeket *(például az előbb már említett Visual Basic, ASP, VBScript, JavaScript vagy éppen a Visual C++)*. Az interfész nem más, mint a támogatott funkciók *(metódusok)* kötött formájú táblázata.



A COM komponens interfészei

A COM komponenseket általában .dll-ekben, vagy futtatható állományokban *(.exe)* helyezik el, amely azután szükség esetén telepíthető a távoli számítógépre is. A távoli objektumok elérését a Distributed COM *(DCOM)* szolgáltatás végzi, távoli eljáráshívások *(Remote Procedure Calls, RPC)* segítségével.

COM+ Services

A COM+ Services a Windows 2000 operációs rendszer alapszolgáltatása, ez tartalmazza a COM-alapú szolgáltatásokat és a Microsoft Transaction Server-t *(MTS)*. A COM+ Services komoly szolgáltatásai közé tartozik a tranzakciókezelés, az objektum életpályájának kezelése, a biztonsági szolgáltatások, események és várakozási sort kezelő *(vagy azt használó)*, úgynevezett queued komponensek kiszolgálása.

COM+ komponensek

A COM+ komponensek olyan COM komponensek, amelyek képesek kihasználni a COM+ Services szolgáltatásait. A COM+ komponensekkel szemben támasztott egyik alapkövetelmény, hogy információkat hordozzanak önmaguktól. A COM architektúra ezekből az alapinformációkból tudja megállapítani, hogy az adott komponens milyen COM+ szolgáltatást használ vagy támogat *(például tranzakciókezelést, vagy éppen terheléselosztást)*.

A COM+ komponensek alkalmazásokba *(Applications)* tömörülnek. Ezek az alkalmazások nem azonosak az Application Center alkalmazásokkal. A COM+ alkalmazás COM+ objektumok gyűjteménye, az Application Center alkalmazás pedig az adott üzleti környezetben használt erőforrások *(webhelyek, fájlok, COM+ komponensek, regisztrációs adatbázis-kulcsok, stb.)* listája.

A komponens-terheléselosztás működése

A CLB két fő részből áll:

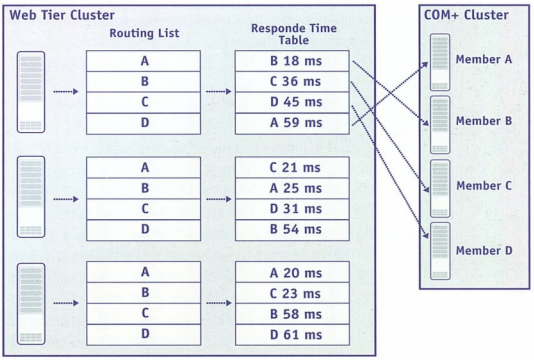
- A COM+ fürt terheléselosztását felügyelő CLB szoftver
- A COM+ fürt *(egy Application Center által felügyelt ki szolgálófürt, amely COM+ komponensek futtatására való)*

A CLB szoftver

A CLB szoftver dönti el, hogy egy adott COM+ komponens az adott körülmények között melyik tagkiszolgálón jöjjön létre. A COM+ komponenst létrehozó üzleti logika a webrétegen működik. Ez a logika általában egy ASP script, amely a CreateObject hívást használja a szükséges COM+ komponens létrehozásához *(ez a kérés mélyebb szinten természetesen CoCreateInstance híváshoz vezet)*. CLB esetén a komponens nem a helyi gépen, hanem a COM+ fürt *(az útválasztási lista (routing list) és a válaszidő-táblázat alapján meghatározott)* tagkiszolgálóján jön létre. A komponens elkészítése után a tagkiszolgáló visszaküldi a megfelelő interfészt. A CLB az objektum létrehozása után már nem vesz részt a rendszer működtetésében *(az a DCOM feladata – a szerk.)*.

Útválasztási lista *(Routing List)*

Az útválasztási lista általában a webrétegen található fürt tagkiszolgálóin található, és a CLB-ben használható COM+ fürt tagkiszolgálóinak listáját tartalmazza *(ld. az ábrán)*. Más esetben ezt a listát az úgynevezett COM+ útválasztó-fürt is tartalmazhatja *(a COM+ útválasztó-fürt feladata csak az útválasztás, nem tartalmaz webréteghez tartozó üzleti logikát)*. Cikkünk keretein belül csak a webrétegű megoldással foglalkozunk.



Útválasztási listák és válaszidő-táblázatok

Az útválasztási listát a rendszergazda hozza létre a webrétegen működő fürt vezérgépén, ami azután automatikusan eljut a fürt többi tagkiszolgálójához. Így a tagkiszolgálók nem tartalmazhatnak különböző COM+ tagkiszolgálókból álló útválasz-

tási táblázatokat. Az útválasztási táblázat webrétegen történő elhelyezésének előnye, hogy így kiküszöböljük az egypontos hibalehetőséget: ha bármelyik tagkiszolgáló leáll, a többi saját táblázatát használva tovább végezheti a munkáját.

Válaszidő-táblázat *(Response Time Table)*

Másodpercenként öt alkalommal *(200 ms-onként)* a webréteg tagkiszolgálóin futó CLB szoftver felveszi a kapcsolatot az útválasztási listában található COM+ fürt tagkiszolgálóival. A válaszok megérkezésének pillanatában a CLB szoftver válaszidő-táblázatot készít. Minél gyorsabban válaszolt egy COM+ tagkiszolgáló, a táblázatban annál előkelőbb helyen szerepel. A webréteg tagkiszolgálója ezután e táblázat alapján dönti el, hogy a kért COM+ komponenst melyik COM+ tagkiszolgálón hozza létre *(úgynevezett round-robin módon a kiválasztott ezután a táblázat végére kerül, és az eddigi második helyezett lép előre)*. A beérkezett kérés tehát a leggyorsabban válaszoló, legkevésbé terhelt COM+ tagkiszolgálóra kerül, a következő kérések pedig már máshoz, egészen addig, míg a táblázat „átfordul". A legközelebbi időmérés alkalmával a táblázat frissül, és minden elölről kezdődik. Minden CLB szoftvert futtató tagkiszolgáló saját válaszidő-táblázatot tart fenn, ez az információ az útválasztási listával ellentétben nem szinkronizált a tagok között.

A COM+ fürt

A webréteg fürtjéhez hasonlóan a COM+ fürtöt is az Application Center varázslójával hozhatjuk létre. A fürt minden tagkiszolgálóján azonos COM+ komponenseknek kell szerepelniük, ezek telepítését külön varázsló könnyíti meg. A komponens létrehozása után annak fel kell ismernie, hogy egy CLB fürtben működik.

Fürt-tűrő COM+ komponensek
(Cluster Aware COM+ Components)

A CLB használatához a COM+ komponensnek fel kell tudni ismernie, hogy milyen helyzetben működik. A legfontosabb a komponens állapotkezelése. A COM+ komponensek nem tárolhatnak önmagukra vonatkozó állapot-információkat, mert ez rossz hatással van a méretezhetőségre és a tranzakció-kezelésre. A méretezhetőséggel azért lehet probléma, mert a rosszul megírt objektumokat nem használhatjuk fel újra azok újbóli létrehozása vagy inicializálása nélkül. A tranzakciókezelés bonyolódik, mert a komponensek állapotinformációi nem léphetik át a tranzakciók határait. A CLB használatához további megkötések szükségesek. A lényeg, hogy a komponens nem használhatja ki „helyzeti előnyeit" *(bárhol, bármelyik tagkiszolgálón működnie kell)*, hiszen a CLB működése közben a komponensek bármelyik tagon létrehozhatók. COM+-ban például processz-szintű tárhelyet használhatunk ahhoz, hogy egy komponens több példánya között megosztott adatokat tároljunk ott. COM+ fürtben ezzel óvatosan kell bánni, hiszen nincs garancia arra, hogy a komponens példányai ugyanazon a tagkiszolgálón jönnek létre. Ha az újabb példány másik tagkiszolgálóra kerül, a processz-szintű információ elveszik. A komponensek állapotinformációit közös, állandó, a fürt tagkiszolgálói által egyaránt elérhető helyen *(pl. DBMS-ben)*, vagy az ügyfél számítógépén *(pl. cookie-k segítségével)* kell tárolni.

A komponens-terheléselosztás telepítése

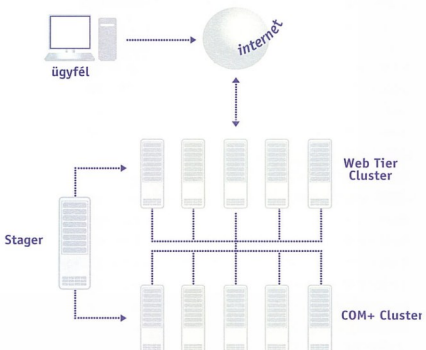
A COM+ fürt telepítése az alábbi lépésekből áll *(a leírás az Application Center Beta 2 verzióján alapul)*:

1. **A fürtök telepítése**

   A CLB-hez két fürtre van szükség: az egyik fürt az útválasztási listát tartalmazza. Ez a fürt (ahogyan azt az előbbiekben is említettük) általában ugyanaz, mint ami az ügyfeleket kiszolgálja (webrétegen futó NLB fürt). A másik maga a COM+ fürt, ennek telepítéséről az online súgóban olvashatunk.

2. **COM+ komponensek telepítése a COM+ fürt tagkiszolgálóira**

   Az Application Center alkalmazásokat a telepítő varázsló segítségével más fürtökre is telepíthetjük. Ez főleg fejlesztői célra használatos fürtöknél hasznos, amikor a folyamatosan változó alkalmazásokat csak időnként frissítjük az éles fürtön (ld. ábra). A telepítő varázslót a fejlesztői gépen futtatjuk (Stager), ami tartalmazza az adott Application Center alkalmazáshoz tartozó COM+ komponenseket is. A témáról bővebben a súgó „Synchronizing and Deploying Content" fejezete ír.



Fejlesztői gép az Application Center-ben

   A telepítő varázsló a COM+ komponenseket alapértelmezésben nem telepíti, mert ahhoz szükség lehet az IIS *(és esetleg a tagkiszolgáló)* újraindítására. A komponensek telepítése a varázslóban természetesen bekapcsolható.
   Fontos még, hogy a komponensek telepítése során a teljes fürtöt nem állíthatjuk le. Folyamatos telepítésre van szükség, a tagkiszolgálók átmeneti leállítására, majd újraindítására. A Microsoft Application Center 2000 Resource Kit további információkat tartalmaz a témáról.

3. **A COM+ komponensek telepítése a webréteg fürtjére**

   A CLB használatához természetesen a COM+ komponenseket a webréteg fürtjének tagkiszolgálóira is telepíteni kell. A telepítő varázsló ebben is segít, de ne feledjük el, hogy a COM+ komponensek telepítése az IIS, esetleg a számítógép újraindításával járhat. Itt is fontos a folyamatos rendszerfrissítés. Másik fontos tényе-

ző, hogy a webréteg tagkiszolgálóin található COM+ komponenseknek szinkronban kell lenniük a COM+ fürt tagkiszolgálóira telepített példányokkal *(ld. még: Microsoft Application Center 2000 Resource Kit)*

4. **CLB-támogatás beállítása a webrétegre telepített COM+ komponensekben**
   A webréteg tagkiszolgálóira telepített komponenseket be kell állítani a CLB támogatásához. Ehhez a Component Services Explorer-t használhatjuk, ami az Application Center MMC modulján keresztül érhető el. Az adott komponens kiválasztása után a tulajdonságlap Activation oldalán jelöljük be a Component supports dynamic load balancing jelölőnégyzetet.
5. **Az útválasztási lista felépítése**
   Az útválasztási listát a webréteg fürtjének vezérkiszolgálóján hozzuk létre, majd az automatikusan eljut a többi tagkiszolgálóra. A COM+ fürt tagkiszolgálóinak listája az Application Center felügyeleti moduljában, a webréteg fürtjének tulajdonságlapján, a Services oldalon állítható be.

Ennyi. Miután mindent telepítettünk, a webrétegen futó alkalmazás a COM+ fürt tagkiszolgálóin futó komponenseket használja majd.

A Component Load Balancing működés közben

Lássuk, hogyan bizonyosodhatunk meg gyorsan a CLB működéséről. Feltesszük, hogy egy fejlesztői gép segítségével telepítjük majd a szükséges elemeket a webréteg és a COM+ fürt tagkiszolgálóira.

1. A fejlesztői gépen Visual Basic segítségével hozzunk létre egy COM+ komponenst, ami a következő metódust tartalmazza:

```
Public Function GetName() As String
  Set WS = CreateObject
("wscript.network")
  GetName = WS.Computername
  Set WS=nothing
End Function
```

2. A COM+ Services Explorer segítségével a komponenst rendeljük hozzá egy COM+ alkalmazáshoz.
3. A fejlesztői gépen hozzunk létre egy Application Center alkalmazást, ami tartalmazza a COM+ komponenst.
4. Telepítsük a komponenst a COM+ fürtre. Ne felejtsük el a Deploy COM+ applications opciót kiválasztani, különben a komponens nem települ fel.
5. Készítsük el a default.asp-t az alábbi tartalommal *(természetesen a YourComponent.YourClass helyére az általunk létrehozott új komponens ProgID-ját írjuk – a szerk.)*:

```
<script language=vbscript
runat="server">
for n=1 to 50
  set x=createobject
("YourComponent.YourClass")
  Response.Write "Component created on: "
  Response.Write x.GetName
  Response.write "<br>"
  set x=nothing
  next
</script>
```

6. A fejlesztői gépen hozzunk létre egy újabb Application Center alkalmazást, ami a default.asp-t és a COM+ komponensünket tartalmazza.
7. Telepítsük az alkalmazás a webrétegre.
8. Ellenőrizzük *(készítsük el)* a webréteg útválasztási listáit és állítsuk be a COM+ komponensek CLB-támogatását.
9. Egy ügyfélgépen nyissuk meg a default.asp oldalt. Ha elsőre nem működik, előfordulhat, hogy az IIS éppen újraindul (a komponens-telepítés miatt).

Ha minden jól ment, a képernyőn megjelenik a komponens példányait futtató COM+ fürt tagkiszolgálóinak neve.

Mikor használjuk a CLB-t?

A CLB nagyszerű megoldás elosztott környezetek építésére. Néha azonban a CLB használata nem ajánlott, például, ha a teljesítmény, a méretezhetőség és a biztonság fontos szempont. Lássuk, mi indokolja a fenti állítást:

Teljesítmény

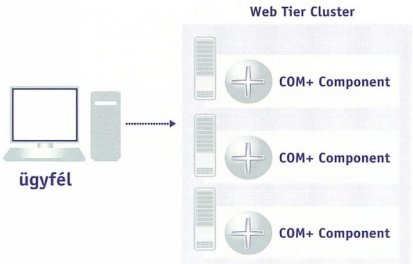
Bármilyen jó is egy webhely, nem lesz sikeres, amíg nem produkál megfelelő teljesítményt. Két fontos szempont az adatátvitel és a válaszidő, a CLB használata mindkét tényezőre hatással van.

Átvitt adatmennyiség

Az átviteli teljesítmény meghatározza, hogy mikor használhatunk hálózaton keresztüli eljáráshívást. A CLB természetesen rontja ezt a teljesítményt, ezért a fürtök tervezésénél ezt figyelembe kell venni. A táblázatban látható, milyen teljesítménnyel működik egy egyszerű „Hello, world" szöveget visszaadó COM+ komponens különféle környezetben:

Környezet	Hívás / mp.	Relatív sebesség
COM+ kiszolgálóalkalmazás (.exe) 10BaseT hálózatban	625	1.0x
COM+ kiszolgálóalkalmazás (.exe), helyi gép, out-of-process	1923	3.08x
COM+ library alkalmazás (.dll), helyi gép, in-process	3333	5.33x

Világos, hogy a hálózaton keresztüli eljáráshívás jóval lassabb, mint a helyi gépen futó objektumok használata. Ez minden szoftverre igaz, legyen az Microsoft, vagy más. Ezért, ha az átviteli teljesítmény nagyon fontos, a CLB nem biztos, hogy jó választás. Ilyenkor a komponenseket érdemes a webréteg tagkiszolgálóira telepíteni. A CLB támogatás elveszik, de a terheléselosztás az NLB-nek köszönhetően továbbra is elérhető.



COM+ komponensek a webréteg fürtjében

Válaszidő

A webhelyek válaszideje nagyon fontos, és a webhely felépítése a válaszidőt nagyban befolyásolja. A webrétegen futó COM+ komponensek ronthatják a webkiszolgáló teljesítményét. Ha a nem COM+-alapú működés válaszideje fontos, a komponenseket érdemes külön COM+ kiszolgálófürtre telepíteni. Ez csökkenti a webkiszolgálók terhelését, és csökkenti a válaszidőt azon ügyfelek felé, akik nem használnak COM+-alapú szolgáltatást *(például statikus weblapokat töltenek le)*. Mások számára a hálózati működés miatt a teljesítmény valószínűleg csökkenni fog. Másik lehetőség, hogy az útválasztási listákat is külön fürtre telepítjük, így még több teljesítményt szabadítunk fel a webkiszolgálókon.

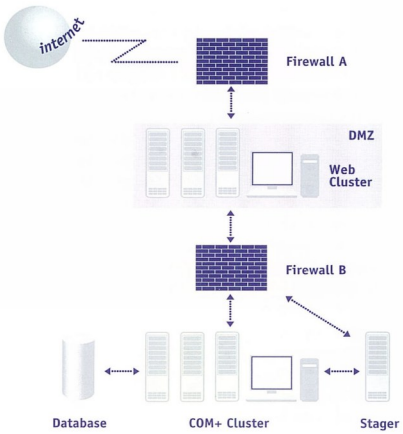
Világos, hogy a nagy teljesítmény és a CLB nem minden esetben fér össze, ezt a rendszer tervezésekor figyelembe kell venni.

Kezelhetőség

Külön COM+ fürtök használata megkönnyíti a rendszer kezelését. Segítségével elkülöníthető a COM+ komponensek és a hálózati kérések kiszolgálása és kezelése. Lehetőség van arra is, hogy egy COM+ fürt több webrétegen futó, független fürtöt szolgáljon ki, és viszont.

Biztonság

A CLB-t sokszor a webkiszolgáló biztonságának növelésére használjuk. Ha a COM+ komponensek feladata az adatok kezelése, azok biztonságosan érik ez azokat. Ha a komponens a webkiszolgálón fut, könnyebben elérhető (és támadható), mintha az külön tagkiszolgálón működik. A COM+ fürtöt általában tűzfal védi *(az ábrán ez a Firewall B)*, ami megakadályozhatja például azt, hogy egy komponenst az ügyfél önmaga hozzon létre, és azzal visszaéljen *(pl. COM+ kéréseket csak a webréteg fürtjétől fogadunk el)*. Az ábrán látható demilitarizált zóna *(DMZ)* a webréteg fürtjét védi két oldalról, két tűzfal segítségével.



COM+ fürt tűzfalak mögött

Összefoglalás

A CLB nagyszerű megoldás elosztott alkalmazások fejlesztéséhez, de óvatosan kell használni. A CLB használata ronthatja a teljesítményt, de ezt ellensúlyozzák előnyei: a méretezhetőség, biztonság, a könnyű telepítés és a terheléselosztás. Széleskörű használata megkönnyíti majd a .net-alapú megoldások fejlesztését.

További információk

http://www.Microsoft.com/ApplicationCenter – Legfrissebb információk a Microsoft Application Center 2000-ről

Microsoft Application Center 2000 Resource Kit – Az Application Center-rel kapcsolatos mélyebb ismeretek tárháza. Nemsokára megjelenik.

http://www.microsoft.com/com – COM és COM+ információk.

http://www.microsoft.com/net – További információk a .NET-ről.

http://www.microsoft.com/windows2000/library/howitworks/cluster/nlb.asp – Hálózati terheléselosztás

Domainregisztráció

Március elseje óta az ISZT *(Internet Szolgáltatók tanácsa)* liberalizálta a domainnév regisztrációt. Sokan azonban félreértik a szabadságot, és úgy vélik, hogy eljöve az ő idejük, hatalmas anyagi előnyre tehetnek szert. A következőkben azt szeretném körüljárni, hogy hol van a határa a szabadságnak, mi jelent korlátot és miért.

A március elsejei változás, a liberalizálás azt jelenti, hogy amíg korábban csak a cégjegyzékbe bejegyzett cég a cégnevével azonos, vagy lapengedély alapján a lap címével azonos domainnév regisztrációjára formálhatott jogot, úgy e határnaptól kezdve bárki, tehát magánszemély is jogosult arra, hogy akár fantázianéven is domaint igényeljen.

A szabadság azonban csak addig terjedhet, amíg az más jogát, jogos érdekét nem sérti. A domainnév igénylésekor az igénylőlapon az aláíró külön nyilatkozik arról is, hogy a domain kiválasztásában a megfelelő gondosságot tanúsította, és meggyőződött arról a cégjegyzék, illetve a Magyar Szabadalmi Hivatal által a web-en közzétett védjegy lista tanulmányozását követően, hogy a választott domain nem jogsértő. Természetesen a gyakorlatban az igénylők egy része úgy írja alá az igénylőlapot, hogy e gondossági követelménynek nem tesz eleget, illetve a domainbrókersége törekvők éppen annak tudatában csapnának le az adott jól csengő kereskedelmi névre, hogy azt majd jó pénzért adják át az arra jogosultnak.

Domain igénylése esetén prioritással rendelkeznek azok, akiknek a *(rövidített)* cégneve vagy bejegyzett védjegye azonos a domainként igényelt karakterekkel. A prioritással nem rendelkezők igényét két hétre „várólistára" helyezik, és ez alatt a két hét alatt bárki *(tehát nem csak a jogosult!)* kifogást terjeszthet elő az igényléssel szemben. Ez esetben az igénylésről és a kifogásról egy tanácsadó testület állásfoglalásának kikérése után határoz a regisztrátor. Abban az esetben, ha kifogás nem érkezik be, a domain automatikusan regisztrálásra kerül. A regisztrációt követően ébredő védjegy-jogosult, vagy akinek cégneve, illetve kereskedelmi neve az adott domain, a bíróságnál keresheti igazát. Bírói fórumok között azonban lehet választani, és a hagyományos, köztudottan a nagy ügyhátralék miatt lassú rendes bíróság helyett a kérdésben járatosabb, tapasztalatokkal is rendelkező jogászokból és informatikai szakemberekből összeállított, gyorsan *(három hónapon belül)* döntő választott bíróságot is igénybe lehet venni.

Mivel magam is jártam már el választott bíróként domainigényléssel kapcsolatos ügyekben, így tapasztalatból állítom, hogy a három tagú bírói testület bizony nincs könnyű helyzetben. Milyen döntést hozzon a bíróság például abban az esetben, ha a madonna.com domainnevet egy magánszemély nyilvánvalóan eladási céllal igényelte? Ha érdekli, mi történik az informatika és a jog határvidékén *(BSA, digitális aláírás, szerzői jog, tartalomfelelősség)* iratkozzon fel a Páneurópa Jogász Unió levelezési listájára! Csatlakozás: **legalnet-join@lyrs.pju.h**u *(üres levéllel)*. Levelezés: **legalnet@lyrs.pju.hu** A madonna szó igen elterjedt szerte a világban, és az énekesnő mellett számos egyház, alapítvány, stb. nevében is szerepel. Számít-e ilyen esetben, hogy ugyanez a szó védjegyoltalom alatt áll az ismert énekesnő javára? Az ügyet eldöntő választott bíróság végül az énekesnőnek adott igazat, annak ellenére, hogy az igénylő ingyenesen tovább akarta ajándékozni a domaint a Madonna nevet viselő karitatív alapítványnak. Hasonló eset történt ismert likőrgyárosunk nevével megegyező domain esetében, és a közkedvelt italt, valamint a tulajdonos nevét is jelentő domain átadására kötelezték az igénylőt. Kicsit nehezebb a helyzet akkor, ha a piacon csak egy meghatározott réteg által ismert és kedvelt termék márkaneve, amely egyben védjegyoltalom alatt is áll kerül domainként regisztrálásra. Történetesen e márkanévnek egyidejűleg más jelentése is ismert. Vajon ez esetben is megilleti-e a védelem a védjegy jogosultját a piac más területén tevékenykedő igénylővel szemben? Automatikusan, csak a védjegyoltalomból következtetve, ilyen védelemre nem tarthat igényt. A védjegy ugyanis nem általános védelmet jelent, hanem meghatározott árucsoport, áruosztály védjegye. Más területen ugyanakkor ugyanazon szóösszetétel használata a piac más áruira nézve nem tilos. Gondoljunk például a sport csokoládéra. Ilyen néven nyilván nem lehet más édességet forgalomba hozni, és ha a domain igénylője a sport.hu-t más édesség eladásra kívánja használni, úgy nyilván sérti a sport csokoládé gyártójának jogait. De ha ugyanilyen domain alatt sporthírreket tesz közzé, akkor a csokoládé gyártója alaptalanul próbálna fellépni. Ez utóbbi esetben azonban már a tartalom is döntő lehet a vita megítélésében. A tartalomról, annak jogsértő jellegéről azonban majd egy következő cikkben írok részletesebben.

E néhány gondolattal arra kívántam rámutatni, hogy a domainnevekkel kapcsolatos problémák korántsem egyértelműek, és megítélésük, biztos álláspontból való eldöntésük bizony még a bíró gyakorlat további fejlődését feltételezik. A jog azonban konzervatív és természeténél fogva lassan reagál a világ új és új kihívásaira, így sem a jogalkotók, sem a jogalkalmazók szemére nem vethető, hogy az Internet robbanásszerű terjedésével újonnan felvetődő kérdésekre még csak nehezen találnak választ.

Dr. Mayer Erika
Ügyvéd
a Páneurópa Jogász Unió elnöke
mayer@nadas-mayer.hu

A domainregisztrációval kapcsolatos tudnivalók, az általános szabályozás, a regisztrációt végző szervezetek *(regisztrátorok)* és a cikkben is említett kéthetes várólista a **http://www.nic.hu** címen érhető el. Ezen a címen ellenőrizhetjük azt is, hogy az általunk igényelni tervezett domainnév szabad-e még, vagy az már valakinek a tulajdonába került. ***(http://www.nic.hu/domainsearch/)***

[mICK]

Összetett lekérdezések

Az előző cikkünkben belemélyedtünk az illesztések lelkivilágába. Megnéztük, hogy egymásba ágyazott lekérdezésekkel milyen egyszerűen meg lehet oldani bonyolult problémákat is. Most további igen hasznos nyelvi elemekkel ismerkedünk meg, amelyek segítségével csoportosíthatjuk adatainkat, műveleteket végezhetünk a csoportokkal, és nagyon látványos riportokat tudunk készíteni. Ehhez kapcsolódóan megnézzük, hogy a szerverbe beépített függvények segítségével mennyire át lehet alakítani az adatok jelentését.

Csoportosítsunk!

Bemelegítésül nézzük meg az alábbi lekérdezést:

```
SELECT
  od.OrderID, p.ProductName,
  od.UnitPrice * od.Quantity AS Amount
FROM
  [Order Details] od
INNER JOIN
  Products p
ON
  od.ProductID = p.ProductID
ORDER BY
  OrderID
```

OrderID	ProductName	Amount
10248	Queso Cabrales	168.00
10248	Singaporean	98.00
10248	Mozzarella	174.00
10249	Tofu	167.40

Egyszerűen kilisztáztuk a megrendeléseket, kiszámolva az adott tétel értékét *(od.UnitPrice * od.Quantity)*. Szép ez a lista, csak túl részletes. Például a 10248-as megrendeléshez három sort listázott ki, mert a megrendelés három altételből állt. Egy riportban nem érdekesek az ilyen részletek, általában csak arra van szükség, hogy egy megrendelés összesen mekkora értékű volt. Első felindulásunkban a már ismertetett SUM függvényt használnánk, ami képes arra, hogy összegezze a tételeinket:

```
SUM(od.UnitPrice * od.Quantity) AS Amount
```

Persze ez nem azt tenné, amit várnánk tőle, hanem az összes megrendelés értékét összeadná, és eredményül egy számot kapnánk, amelyben minden megrendelés együttes értéke lenne. Mi lenne, ha lenne olyan utasításunk, amivel megmondhatnánk, hogy csoportosítsa a sorokat az OrderID mező alapján, és a csoportokra végezze el az összegzést? Természetesen van ilyenünk, a GROUP BY az. Segítségével a GROUP BY mögé írt oszlopok szerint történik az eredményhalmaz csoportosítása, azaz az azonos OrderID-jú sorokból egyet készít, és a csoportokra kiszámítja az aggregált eredményeket.

Alakítsuk át a példánkat úgy, hogy megrendelésenként összegzett listát készítsen a GROUP BY és a SUM segítségével:

```
SELECT
  od.OrderID,
  SUM(od.UnitPrice * od.Quantity)
  AS Amount
FROM
  [Order Details] od
INNER JOIN
  Products p
ON
  od.ProductID = p.ProductID
GROUP BY
  od.OrderID
ORDER BY
  OrderID
```

OrderID	Amount
10248	440.00
10249	1863.40
10250	1813.00

Nagyszerűen működik! Miért vettem ki a p.ProductName oszlopot? Azért, mert semmi értelme, hisz pont az volt a célunk, hogy termékektől és megrendelés tételektől független listát kapjunk. Ha ezt elfelejtenénk, figyelmeztetni fog a szerver:

```
Column 'p.ProductName' is invalid in the se-
lect list because it is not contained in ei-
ther an aggregate function or the GROUP BY
clause.
```

Azaz a fordító a p.ProductName-et csak akkor fogadja el SELECT mögött, ha azt vagy felsoroljuk a GROUP BY-ban, vagy egy aggregáló függvénybe foglaljuk bele. Az előbbinek az lenne a következménye, hogy a megrendelés tételeken belül termékenként tovább lenne bontva a részösszeg. Sokszor ez is cél lehet. A második javaslattal kapcsolatban: nehéz lenne olyan beépített aggregáló függvényt keresni, ami a terméknéven valami hasznosat tudna végezni. Úgyhogy ezt felejtsük el.

Mi van, ha csak azokat a megrendeléseket akarjuk kilistázni, amelyek összmegrendelés értéke nagyobb, mint 1000$? A WHERE használata sajnos nem vezet eredményre, mert az csak az egyes Order Details sorokban található értékekre tud szűrni, és nem pedig azok összegére. Másképpen fogalmazva valami ilyesmit szeretnénk látni a WHERE-ben:

```
WHERE
   SUM(od.UnitPrice * od.Quantity) > 1000
```

vagy

```
WHERE Amount > 1000
```

Csakhogy a WHERE-ben nem lehet használni aggregáló függvényeket, így a SUM-ot sem. Hogyan juthatunk túl ezen a dilemmán? Úgy, hogy van egy olyan speciális záradék *(clause)*, amelyet arra találtak ki, hogy a GROUP BY által definiált csoporton lehessen vele feltételeket érvényesíteni. Ez a záradék a HAVING. A HAVING nagyon hasonló a WHERE-hez, a különbség abban rejlik, hogy a záradékok után álló kifejezés mikor kerül

Microsoft BackOffice

kiértékelésre. A WHERE után álló kifejezést a csoportosítás előtt értékeli ki a végrehajtó egység, azaz a WHERE segítségével előre kiválogatjuk azokat a sorokat, amelyeket csoportosítani szeretnénk. Ezután jön maga a GROUP BY-ban előírt csoportosító művelet. Létrejönnek a csoportok, valamint kiszámítódnak a csoportokra kijelölt aggregáló kifejezések. Ekkor jön a képbe a HAVING. A parancsvégrehajtó kidobálja azokat a csoportokat, amelyekre nem teljesül a HAVING után álló feltétel. A szenzációs az a dologban, hogy a HAVING után használhatunk aggregáló függvényeket, ellentétben a WHERE-el!
Mivel a WHERE segítségével drasztikusan le lehet csökkenteni a csoportosítandó sorok számát, ezért érdemes minden olyan feltételt, ami nem a csoportokra vonatkozik a WHERE-be rakni a HAVING helyett. Ha ellenkezően cselekszünk, a fordító nem fog figyelmeztetni minket. ő szolgai módon végrehajtja a lekérdezést az általunk előírt módon. A felhasználók viszont szólnak majd az alkalmazásunk lassúsága miatt...
Szerencsére azonban a Query Optimizer ennél okosabb. Általában, hangsúlyozom, általában észreveszi, hogy nem optimálisan írtuk meg a lekérdezést, és a nem megfelelő helyre írt kifejezéseket a végrehajtás idejére átrakja a megfelelő helyre. Még sokszor tapasztaluk majd, ahogy az SQL Server fejlesztők intelligenciája igyekszik pótolni a buta alkalmazásfejlesztőét. Hogy érthetőbb legyen a HAVING és a WHERE közötti különbség, egy táblázatban összefoglaltam, hogy milyen helyzetben melyik záradékot használhatjuk.

	WHERE	HAVING
Mikor szűr?	A csoportosítás előtt	A csoportosítás után
Mit szűr?	Sorokat	Csoportokat
Tartalmazhat-e aggregáló függvényeket?	Nem	Igen

Nézzünk egy példát, amelynek segítségével közelebb kerülhetünk a GROUP BY és a HAVING szelleméhez. Lássunk egy elég bonyolult lekérdezést, amiben minden benne van, amit eddig tanultunk:

```
SELECT
  p.ProductID, p.ProductName,
  'Amount' = SUM(od.UnitPrice *
   od.Quantity)
FROM
  [Order Details] od
INNER JOIN
  Orders o
ON
  o.OrderID = od.OrderID
INNER JOIN
  Products p
ON
  p.ProductID = od.ProductID
WHERE
  o.OrderDate >= '1998.05.05' AND
  o.OrderDate <= '1998.05.07'
GROUP BY
  p.ProductID, ProductName
HAVING
  SUM(od.UnitPrice * od.Quantity) > 800
ORDER BY
  Amount DESC
```

Az SQL kód magyarra fordítása: készítsünk egy olyan listát, amely az 1998. május ötödike és hetedike közötti megrendelések összértékét listázza ki, termékenkénti bontásban. Csak azokra a termékekre vagyunk kíváncsiak, amelyek megrendeléseinek összege nagyobb, mint 800 dollár. A lista legyen rendezve az eladási érték alapján, csökkenő sorrendben. Huh, magyarul nehezebb megfogalmazni, mint SQL-ül! Nézzük meg a kimenetét:

```
ProductID ProductName     Amount
64        Wimmers gute    4389
 2        Chang           1178
16        Pavlova          802
```

Néhány szó a szintaktikával kapcsolatban. Az

```
'Amount' = SUM(od.UnitPrice *
   od.Quantity)
```

kifejezés a

```
SUM(od.UnitPrice *
   od.Quantity) AS Amount
```

kifejezéssel egyenértékű. Lehet így is írni, meg úgy is írni. Használjuk az, amelyik olvashatóbb számunkra. A leglustábbak a második formát használják, úgy, hogy még az AS-t is elhagyják *(én is ilyen vagyok)*.
Az ORDER BY-ban az Amount álnevet használhattuk a bonyolult SUM(od.UnitPrice * od.Quantity) helyett. Ezt a szintaktikai könnyítést sajnos csak az ORDER BY-ban használhatjuk ki, a HAVING-ben már nem. Kár.
Az

```
o.OrderDate >= '1998.05.05' AND
o.OrderDate <= '1998.05.07'
```

szűrőfeltételt elegánsabban is megfogalmazhatjuk a BETWEEN operátor segítségével:

```
OrderDate BETWEEN
 '1998.05.05' AND '1998.05.07'
```

A két kifejezés logikai értéke azonos.
Gyakori kérés a marketing vagy a pénzügy részéről, hogy olyan összesített statisztikát kérnek, amelyben az eladási adatok napi bontásban láthatók. Mivel a generálódó listát emberek fogják kiértékelni, ezért őket elsősorban az irányvonalak érdeklik, nem pedig az összes részeredmény az utolsó bitig. Így például feltételként szabják, hogy a napi listában csak azok a termékek szerepeljenek, amelyekből több mint

egyet rendeltek meg egy adott napon *(a nap slágere)*:

```
SELECT
   OrderDate,
   p.ProductName,
   'Amount' =
   SUM(od.UnitPrice * od.Quantity),
   COUNT(*) AS OrderedProducts
FROM
... --ugyanaz, mint az előző lekérdezésben
WHERE
   OrderDate BETWEEN
   '1998.05.01' AND '1998.05.07'
GROUP BY
   OrderDate, p.ProductID, ProductName
HAVING
   COUNT(*) > 1
ORDER BY
   OrderDate, Amount DESC
```

```
OrderDate   ProductName Amount  OProd
1998-05-05  Chang         532     2
1998-05-06  Chang         646     2
1998-05-06  Grandma's     525     2
1998-05-06  Tofu          488     2
```

Ebből olyan okosságokra lehet következtetni, hogy a Chang nagyon finom lehet, mert ötödikén és hatodikán is megrendeltek belőle kettőt is! Ennél tovább azonban nem megyünk, ez nem a mi szakmánk.

Szakmai szempontból az utolsó oszlop érdekes a számunkra: `COUNT(*) AS OrderedProducts`. A `COUNT(*)` a többi aggregáló függvényhez hasonlóan másként viselkedik, ha GROUP BY van a közelben: nem az egyedi sorokat számolja meg, hanem a csoportokon belüli sorok számát. Másképpen: nem a teljes eredményhalmazra ad egy eredményt, hanem minden egyes csoportra külön-külön. Pont ez az, ami nekünk kellett, és a `HAVING` volt olyan szíves aggregáló függvényt beengedni a feltételek közé. Hurrá!

Beépített skaláris függvények

Mi is az a skaláris függvény? A függvény állatfaj azon alfaja, amely egy értékből egy értékre képez le. Azaz nem olyan, mint a SUM volt, ami sok sor tartalmát összegezve adott vissza egyetlen számot, mert ő az aggregáló típusú függvények képviselője, azaz, amelyek több értékből állítanak elő egyet. Inkább gondoljunk az abszolút értéket képző ABS függvényre *(a blokkolásgátló függvény :)*. `ABS(-4) = 4`. Azaz a mínusz négyet leképezte plusz négyre. Nagy csodák vannak ebben a Serverben!

Az SQL Server nagyon sok beépített, skaláris függvénnyel rendelkezik. Vannak matematikai célúak: abszolút érték *(ABS)*, trigonometrikus függvények *(SIN, COS, TAN, COT, valamint ezek arcus megfelelői)*, logaritmus függvények *(LOG, LOG10)*, exponenciális függvény *(EXP)*, kerekítések *(ROUND, FLOOR, CEILING)*, véletlen szám generáló függvény *(RAND)* stb. Majdnem olyan gazdag a kínálat, mint más „polgári" nyelvekben, mint pl. a Visual Basic-ben.

Van nagyon sok szövegkezelő függvény: különböző darabolások *(LEFT, RIGHT, SUBSTRING)*, szövegdarabok keresése *(PATINDEX, CHARINDEX)*, kis-nagybetű konvertálók *(UPPER, LOWER)*, számformátumról szövegre átalakító *(STR)*, kezdő és záró szóközt levágó *(LTRIM, RTRIM)*. Vannak egzotikusabbak is: REVERSE, ami megfordít egy szöveget: „cirmos" ‡ „somric". Van olyan, ami nagyon hasznos lenne, ha magyarul is működne, csak hát a magyar nyelv elég ellenálló a formalizálással szemben: a DIFFERENCE és a SOUNDEX. A DIFFERENCE egy 0-4-es skálán képes megmondani két szövegről, hogy kimondva, hangzásban *(!)* mennyire hasonlítanak. Nem a karakterláncok írt, hanem kimondott formája. Ez a szolgáltatás nagyon jól jönne például egy telefonkönyv alkalmazásnál, ahol nem lehet tudni, hogy pontosan hogyan írtak egy nevet, de például valahogy úgy hangzott, hogy „socó". A „Smith" és a „Smythe"-re a például DIFFERENCE azt mondja, hogy a távolságuk 4, azaz nagyon hasonlítanak. Az „other" és a „brother" szavak 2 távolságra vannak egymásra, azaz még hasonlítanak, de azért nem annyira. Nagyon jó lenne ez magyarul is! Így talán a Gizike és a gőzeke is kaphatna egy 1-est.

Tovább a függvények útján. Nagyon hasznosak, bár ritkábban használatosak az úgynevezett metaadat-függvények. Ezek a rendszertáblákból kérdeznek le adatokat *(manuálisan tilos, mert bármelyik szervizcsomag megváltoztathatja)*, melynek segítségével belső információkat lehet megtudni az adatbázisunk tulajdonságairól. Ha például az alkalmazásunk kíváncsi, hogy a Employee nevű tábla FirstName nevű oszlopa hány byte-ot foglalhat el az adatbázisban:

```
COL_LENGTH ('Employee', 'FirstName')
```

Ez egy nvarchar*(50)*-es oszlopra 100-at adna eredményül *(az nvarchar Unicode formátumú, azaz minden karakter 2 bájtot foglal el)*.

További függvénykategóriákat is találunk még a Serverben, de ezeket terjedelmi okok miatt most nem közöljük. A Books Online-ban részletesen dokumentálva vannak mindannyian.

Végül, de nem utolsó sorban beszéljünk az egyik leghasznosabb függvénycsaládról: a dátumkezelő függvényekről. Ezek megérnek a többinél kicsit több figyelmet.

Dátumzsonglőrködés

Eddig elég egyszerű volt bevetni a GROUP BY-t, mivel midig volt egy olyan oszlop, amire természetesen lehetett csoportosítani. Azonban ilyen nem mindig létezik. Például az előző példában az OrderDate napra kerekített érték volt, így könnyű volt napra csoportosítani, mivel csak be kellett írni a GROUP BY-ba. De mit teszünk, ha heti bontásban várják a kimenetet? Ha nem ismerjük a DATEPART függvényt, akkor bajban leszünk. De ha igen, akkor:

```
SELECT
   DATEPART(wk, OrderDate) AS WeekNum,
   'Amount' = SUM(od.UnitPrice *
   od.Quantity),
   COUNT(*) AS OrderedProducts
FROM
...
WHERE
```

```
  OrderDate BETWEEN
  '1998.01.01' AND '1998.02.01'
GROUP BY
  DATEPART(wk, OrderDate)
HAVING
  COUNT(*) > 1
ORDER BY
  WeekNum, Amount DESC
```

WeekNum	Amount	OrderedProducts
1	4691.0000	12
2	30894.6600	30
3	18822.6700	38
4	24330.5400	38
5	22115.8500	34

A DATEPART függvény az egyik leggyakrabban használt beépített függvény. Visszatér egy egész számmal, ami a date paraméterben megadott dátum egy bizonyos darabjának felel meg. A használata nagyon egyszerű:

```
DATEPART(datepart, date)
```

ahol a datepart a következő kifejezések valamelyike lehet:

Jelentés	Teljes név	Rövidítés
Év	year	yy, yyyy
Negyedév	quarter	qq, q
Hónap	month	mm, m
Az év n. napja	dayofyear	dy, y
Nap	day	dd, d
Hét	week	wk, ww
A hét n. napja	weekday	dw
Óra	hour	hh
Perc	minute	mi, n
Másodperc	second	ss, s
Ezredmásodperc	millisecond	ms

A függvényben használhatjuk mind a teljes nevet, mind a rövidítést.
A példánk bizony sánta. Mivel minden évben van 1. hét, 2. hét, satöbbi, ezért a lekérdezésünk össze fogja vonni az összes év ugyanazon hetébe eső eladásokat. Ez természetesen nem helyes, és ezt a problémát úgy fogjuk orvosolni az utolsó, szinte tökéletes lekérdezésünkben, hogy a hét sorszáma mellé felsoroljuk az évet is mind a SELECT utáni listában, mind a GROUP BY-ban, így egyértelműen azonosítva lesz a hét.
Az előző példánkban arra voltunk kíváncsiak, hogy az adott dátum az év hányadik hetébe esik. Azonban az SQL Servert Amerikában írták, ahol a hét első napja a vasárnap, nem pedig a hétfő. Ők tudják, mi a jó nekik, azonban a DATEPART is ennek megfelelően működik, aminek mi nem örülünk. Mivel azonban a Microsoft nem csak Amerikában akarja eladni az SQL Servert, ezért beépítette annak lehetőségét, hogy megváltoztassuk a hét első napját:

```
SET DATEFIRST 1
```

Ennek hatására a hét első napja ismét a hétfő lesz, így az összes dátumkezelő függvény helyesen fog működni. Az SQL Serverben sok, a fentihez hasonló beállítás létezik, amelyekkel a szerver alapértelmezett viselkedését változtathatjuk meg. Ezekkel egy későbbi cikkben még részletesen foglalkozunk.
A dátumkezelő függvények további igen hasznos képviselője a DATEADD függvény. Ez, mint a neve is sugallja, arra való, hogy egy dátumhoz hozzáad valamilyen időintervallumot. Mi határoz meg egy időintervallumot? A hossza és a mértékegysége. Ennek megfelelően a függvény formátuma a következő:
DATEADD *(datepart , number, date)*
A datepart az előző táblázatban közölt értéket veheti fel, a weekday kivételével, mert annak nincs semmi értelme ebben az összefüggésben. Szerintem a dayofyear-nek sincs, de az úgy működik, mintha day-t írtunk volna. A number egy egész szám, ami az intervallumot írja le. A date pedig a kiinduló dátum. Nézzük meg működés közben:

```
SELECT DATEADD(day, 1, '2000/01/05 18:12')
2000-01-06 18:12:00.000
SELECT DATEADD(mi, 5, '2000/01/05 18:12')
2000-01-05 18:17:00.000
SELECT DATEADD(hh, -3, '2000/01/05 18:12')
2000-01-05 15:12:00.000
DECLARE @d DATETIME
SET @d = '2000/01/05 18:12'
SELECT @d + 3
2000-01-08 18:12:00.000
```

Az első három példa morális tanulsága: nincs DATESUB, a DATEADD-ot kell negatív számmal meghívni.
Az utolsó példa ravasz. Egy dátum típusú mezőhöz hozzáadunk 3-at, egy egész számot, aminek az a jelentése, hogy a dátumot megnöveli 3 nappal. Érdekes, de ha valaki nem tudja explicit a + operátor e polimorf tulajdonságát, az meglepődhet a kódunkon.
A harmadik hasznos dátumkezelő függvény a DATEDIFF. Formátuma:

```
DATEDIFF (datepart, startdate, enddate)
```

Azaz a startdate és enddate dátumok különbségét adja vissza a datepart-ben definiált egységben:

```
--Hány másodperc is egy nap?
SELECT    DATEDIFF(second,    '2000.01.01',
'2000.01.02')
86400
```

Dátum agytorna

Szép lett az előző példánk listája, de nekem például nem sokat mond az, hogy most a 38. hétben járunk. A menzán és a hivatalokban gyakran láthatjuk ezt a fajta időmeghatározást, de nekem sokkal szimpatikusabb lenne a lista, ha a hetet jelző szám mellett ott láthatnám azt is, hogy mely dátumhatárok zárják az adott hetet. Próbáljuk meg kitalálni, hogyan le-

hetne ezt összerakni Transact SQL-ben. Nem lesz triviális, kéretik egy dupla KV-t inni a következők előtt!
Adott a dátumunk, tároljuk ezt a @d változóban. Azt, hogy ez a dátum az év hányadik hetébe esik, a DATEPART(week, @d) függvénnyel könnyedén megtudhatjuk. Hogyan lesz ebből meg a keresett hét kezdő dátuma? Úgy, hogy valahogyan meg kellene találni az adott év első hétfőjét, és ahhoz hozzá kellene adni annyiszor 7 napot, ahányadik héten járunk az első hétfőhöz képest. Az év eleji tört hét is hétnek számít! Nézzük meg mindezt Transact SQL-ben!

```
--A hét első napja a hétfő legyen
SET DATEFIRST 1
--Ehhez a dátumhoz keressük a hetet és a
--hetet záró határokat
DECLARE @d DATETIME
--Ebben lesz az év első napjának dátuma
--(nem első hétfő, hanem január elseje!)
DECLARE @dFirstDayOfYear DATETIME
--Teszt dátum. Ez egy keddi nap a 2. héten
SET @d = '2000/01/04'
--A dátumhoz tartozó hét tesztelése
SELECT DATEPART(week, @d)
2
--Az év első napjának megkeresése
SET @dFirstMondayOfYear = CONVERT(CHAR(4), @d,
112)
SELECT @dFirstDayOfYear
2000-01-01 00:00:00.000
```

Itt álljunk meg egy pillanatra. Mi az a CONVERT függvény, és mit jelent a 112-es paraméter? A CONVERT a különböző adattípusok közötti konverzióra való. Különösen akkor hasznos, ha dátum formátumot kell szöveggé konvertálni. Az első paramétere mondja meg, hogy milyen típussá szeretnénk konvertálni. Itt char(4)-et adtunk meg, ami 4 karaktert képes tárolni. A második paraméter a konvertálandó kifejezés, a harmadik pedig a konvertált eredmény formátumát szabályozza. Dátum bemenet és szöveg kimenet esetén a 112 azt jelenti, hogy a dátumot yyyymmdd formátumra konvertálja át. De hisz az eredmény 8 karakter, mi meg char(4)-et adtunk meg! Ez benne a trükk. 2000. 01. 04.-ből 20000104 lenne, de mivel a char(4) csak az első 4 karaktert képes eltárolni, a maradék négy egyszerűen elveszik. Azaz mi lesz a konverzió eredménye? "2000". De akkor miért kaptunk a SELECT @dFirstDayOfYear eredményeként 2000-01-01-et? Azért, mert a @dFirstDayOfYear dátum típusú, és a szerver a „2000" sztringet 2000. január 1-é konvertálta. Implicit módon, azaz anélkül, hogy erre külön megkértük volna. Ez azért egy kicsit piszkos munka volt. Inkább segítsünk neki:

```
SET @dFirstDayOfYear = CONVERT(CHAR(4), @d,
112) + '.01.01'
2000-01-01 00:00:00.000
```

Na, ez így már szép. De menjünk tovább. Hogyan kapjuk meg ebből az első hétfőt? Ha tudjuk, hogy január elseje a hét hányadik napja, akkor ebből már könnyű kiszámolni, hogy az első hétfő hányadikára esik: menjünk vissza az év első napjától annyi napot, ahányadik napra esik az a hétben, és adjunk hozzá 8-at. Például 2000. január elseje szombat volt, ami a hét 6. napja.

```
SELECT DATEPART(weekday, @dFirstDayOfYear)
6
```

Ha visszamegyünk 6 napot, az 1999. december 26-a, ami a 2000. év első hétfőjét megelőző hétfő előtti nap *(vasárnap)*.

```
SELECT    DATEADD(day,    -DATEPART(weekday,
@dFirstDayOfYear), @dFirstDayOfYear)
1999-12-26 00:00:00.000
```

Ehhez már csak hozzá kell adni 8 napot, és meglesz a 2000. év első hétfője.

```
SELECT    DATEADD(day,    -DATEPART(weekday,
@dFirstDayOfYear)+8, @dFirstDayOfYear)
2000-01-03 00:00:00.000
```

Eljutottunk a tárgy év első hétfőjéig. Most már nincs más dolgunk, mint ehhez hozzáadni annyiszor 7 napot, ahányadik héthez keressük a hetet kezdő hétfőt.

```
SELECT    DATEADD(day,    (-DATEPART(weekday,
@dFirstDayOfYear)+8)  +  (DATEPART(week,  @d)-
2)*7, @dFirstDayOfYear)
2000-01-03 00:00:00.000
```

Azért kellett kettőt kivonni a hét számából, mert 1-től kezdődik a hetek számozása és nem 0-ától, valamint, mert az aktuális napot megelőző hétfőre vagyunk kíváncsiak, nem pedig a következőre. A záró napot innentől kezdve gyerekjáték meghatározni, csak nem kettőt, hanem egyet kell kivonni a hetek számából.

```
SELECT    DATEADD(day,    (-DATEPART(weekday,
@dFirstDayOfYear)+8)  +  (DATEPART(week,  @d)-
1)*7, @dFirstDayOfYear)
2000-01-10 00:00:00.000
```

Alakítsuk át a korábbi GROUP BY-os példánkat úgy, hogy a hetek száma mellé legyen kiírva azok kezdete és vége is. Ehhez az előbb kiagyalt kifejezéseket össze kell vonni, és be kell írni a megfelelő helyre a kiinduló lekérdezésben, valamint rakjuk bele az évet is, ahogy korábban ígértük:

```
SET DATEFIRST 1
SELECT
  DATEPART(year, OrderDate) AS YearNum,
  DATEPART(wk, OrderDate) AS WeekNum,
  DATEADD(day, (-DATEPART(weekday,
  CONVERT(CHAR(4), OrderDate, 112) +
  '.01.01')+8) + (DATEPART(week,
  OrderDate)-2)*7, CONVERT(CHAR(4),
```

Microsoft BackOffice

```
    OrderDate, 112) + '.01.01') AS StartDay,
    DATEADD(day, (-DATEPART(weekday,
    CONVERT(CHAR(4), OrderDate, 112) +
    '.01.01')+8) + (DATEPART(week,
    OrderDate)-1)*7, CONVERT(CHAR(4),
    OrderDate, 112) + '.01.01') AS EndDay,
    'Amount' = SUM(od.UnitPrice *
    od.Quantity),
    COUNT(*) AS OrderedProducts
FROM
...
WHERE
    OrderDate BETWEEN
    '1997.12.22' AND '1998.01.18'
GROUP BY
    DATEPART(year, OrderDate),
    DATEPART(wk, OrderDate),
    DATEADD(day, (-DATEPART(weekday,
    CONVERT(CHAR(4), OrderDate, 112) +
    '.01.01')+8) + (DATEPART(week,
    OrderDate)-2)*7, CONVERT(CHAR(4),
    OrderDate, 112) + '.01.01'),
    DATEADD(day, (-DATEPART(weekday,
    CONVERT(CHAR(4), OrderDate, 112) +
    '.01.01')+8) + (DATEPART(week,
    OrderDate)-1)*7, CONVERT(CHAR(4),
    OrderDate, 112) + '.01.01')
ORDER BY
    WeekNum, Amount DESC
```

YN	WN	StartDay	EndDay	Amount	OP
1997	52	1997-12-22	1997-12-29	17678	32
1997	53	1997-12-29	1998-01-05	14871	18!
1998	1	1997-12-29	1998-01-05	4691	12!
1998	2	1998-01-05	1998-01-12	30894	30
1998	3	1998-01-12	1998-01-19	18822	38

Konklúzió

Látható, hogy az évváltásnál nem jól működik a dátumkezelő algoritmusunk. Ezt még tökéletesíteni fogjuk a következő számban. És miért lett ennek az egyszerű feladatnak a megoldása ilyen bonyolult, annak ellenére, hogy nem is tökéletes? Azért, mert majdnem ugyanazt a hosszú kódrészletet négyszer egymás után le kellett írnunk, szinte változatlanul. De hát nem azt tanultuk az iskolában, hogy az ismétlődő kódrészeket függvényekbe kell rakni? De! És nem arról regéltek nekünk, hogy a függvények paraméterezésével még a nem teljesen azonos kódrészeket is össze lehet vonni? De, de, de! Akkor miért nem élünk ezzel a lehetőséggel? Microsoft SQL Server 7-ig azért nem, mert nem volt meg a módunk erre, mert nem voltak User Defined Function-ök, felhasználói függvények. De SQL 2000-ben végre vannak, és pont az ilyen problémákra adnak igen elegáns megoldást. De erről majd a következő részben.

Soczó Zsolt MCSE, MCSD, MCDBA
Protomix Rt.

BILL GATES MONDJA...

A kiszolgálók lesznek a számítástechnika új központjai?

A személyi számítógépek gyorsan fejlődtek az elmúlt két évtized folyamán, és olyan számítási teljesítményt nyújtanak, melyet régen elképzelni sem tudtunk volna – ráadásul több százmillió embernek. Most, hogy a kommunikációs hálózatok is robbanásszerű fejlődésen mennek át, sokunk kíváncsi rá, hogy vajon ismét központosítva lesz-e a számítástechnika világa.

Óhatatlanul felmerül bennünk a kérdés: nem lehetséges-e az, hogy egy „izmos" asztali gép, laptop vagy akár PDA *(Personal Digital Assistant)* helyett ezeknél sokkal egyszerűbb eszközöket használjunk, melyek nagysebességű hálózati kapcsolatokon keresztül érnek el hatalmas teljesítményű központi számítógépeket?

Ez igen csábító elgondolás. Az információk központilag tárolódnak, és bárhonnan elérhetjük őket. Ha a zene, melynek meghallgatási jogát megvásároltuk központilag lenne tárolva, nem lenne szükséges otthonunkban CD lemezeket vagy audio kazettákat *(vagy bármely más hanghordozót)* tartani, bár a Hi-Fi minőség eléréséhez még mindig ragaszkodnunk kellene ezekhez.

Való igaz, néhány tárolóeszköz központi helyre kerül. Az elektronikus levelezést szolgáló programok például vagy az asztali gép merevlemezén, vagy a központi vállalati kiszolgálón *(esetleg mindkettőn)* tárolják az üzeneteket. A Hotmail.com elképesztő népszerűsége – mely annak köszönhető, hogy felhasználói bármikor, bárhonnan elérhetik ezt az ingyenes e-mail alkalmazást, melynek csak központi tárolója van – bizonyítja legjobban a központilag tárolt információ szükségességét. A Hotmail-nek ugyanis több mint 25 millió felhasználója van.

Az információ kétségkívül egyre központosítottabbá válik, a kiszolgálók pedig egyre nagyobb teljesítményűek lesznek. Ezek a kiszolgálók biztosítják a webhelyek elérhetőségét, összevonják az adatokat, és kereshetővé teszik azokat. A nagy központi számítógépek egyre több adatunkat fogják tárolni, és automatikusan elérhetővé teszik azokat számunkra, függetlenül attól, hogy mi, a végfelhasználók milyen számítógépet is használunk. De ez semmiképp sem jelenti azt, hogy a PC-k és más információ-elérést szolgáló eszközök, melyekkel az emberek a kiszolgálókhoz kapcsolódhatnak, kevésbé intelligensek lesznek. Épp ellenkezőleg, ezek teljesítménye is növekedni fog.

Az elmúlt években kiegyensúlyozatlanság volt megfigyelhető. A számítási teljesítmény és a tárolókapacitás legnagyobb növekedése a PC-knél jelent meg, melyeket az emberek közvetlenül használnak, és amelyeket ezért általában „ügyfeleknek" hívunk. Természetesen a kiszolgálók is fejlődtek, de nem igazán tartottak lépést a kapcsolat ügyféloldalán végbemenő gyors fejlődéssel.

Az elkövetkezendő években a kiszolgálók képességei várhatóan gyorsabban fognak fejlődni, mint az ügyfeleké. A teljesítmény mindkét oldalon nőni fog, ám most a kiszolgálók javára billen a mérleg.

Az egyik oka annak, hogy a kiszolgálók önmagukban nem képesek kielégíteni az összes jövőbeni igényt, az, hogy a kommunikációs infrastruktúra még nem elég jó, és egy jó darabig még nem is lesz az.

Az irodákban dolgozók általában meglehetősen gyors hálózatokat használnak, de a legtöbb Internetre kapcsolódó otthoni felhasználó betárcsázásos, telefonos hálózatot használ. A kábelmodemek és a telefonos DSL kapcsolatok néhány helyen megvalósulnak, de a nagysebességű otthoni kapcsolatok a legtöbb ember számára még sokáig elérhetetlenek maradnak, még az Egyesült Államokban is. Az USA-ban sok PC nem kapcsolódik az Internetre, és ez az arány Európában még roszszabb, mert itt sokkal magasabbak a kommunikációs költségek *(az USA-ban a helyi hívások ingyenesek – a ford.)*.

Ráadásul még a kábelmodemek és a DSL kapcsolatok sem elég gyorsak ahhoz, hogy például a hagyományos televíziónak megfelelő minőségű képátvitelt biztosítsanak. Egy igazán jó Internet-kapcsolatnak 1 megabitet kellene átvinnie másodpercenként – manapság egy CD-ROM ennél 50-szer, egy merevlemez pedig százszor nagyobb teljesítményre képes.

Mivel időbe telik, amíg az adatok átmennek a hálózaton, fontos az ügyfél és a kiszolgáló között átviendő adatmennyiséget minimálisra csökkenteni. Ennek pedig egyik *(és legkézenfekvőbb)* módja az, hogy bőséges számítási teljesítményt, memóriát és tárolókapacitást biztosítunk ügyféloldalon, így nem kell szükségtelenül a kiszolgáló erőforrásaira támaszkodnunk. Az ár/teljesítmény viszonyszám gyors csökkenésével egyre inkább lehetséges ennek megvalósítása. Az alacsony árak miatt az emberek „kezébe adhatjuk" ezt a teljesítményt, és ha az ügyfélgépek teljesítménye elegendő, csak kevés idő vész el a kiszolgáló és az ügyfél között zajló adatkommunikáció miatt. Mindegy, hogy egy hálózati kapcsolatnak mekkora a sávszélessége, mindenképpen lesznek lassulások a „késleltetés" miatt. Az egyik ilyen lassulást okozó tényező a fény sebessége, amely határt szab az információáramlás gyorsaságának.

Egy másik ok arra, hogy a számítási teljesítmény miért nem lesz központosítva az, hogy ebből rengetegre van szükség az új eszközök, adattípusok és a multimédiás folyamatos adatátvitel *(például videolejátszás)* kezeléséhez. Szükség van a számítási teljesítményre az új számítógépes játékoknál is, melyek háromdimenziós, gyorsan mozgó alakzatokat jelenítenek meg. Egy központi kiszolgálóhoz kapcsolt terminál pedig erre egész egyszerűen nem képes.

Az embereknek olyan eszközökre lesz szükségük, mely alkalmazkodik minden olyan új dologhoz, amit csak hozzá akarnak kapcsolni. Ezeknek a perifériáknak a számbeli növekedése még sokáig folytatódni fog, a számítási teljesítmény pedig elosztik.

Végül, bár úgy tűnik, hogy az információ központilag tárolódik, jelentős része helyben is megtalálható. Ily módon akkor is elérhetjük információinkat, ha éppen nincs hálózati kapcsolatunk, a kiszolgáló nem működik, vagy a hálózat leterheltsége miatt csak lassan, vagy egyáltalán nem tudnánk azt elérni. Az adatok tükrözése az összes olyan eszközre, melynek el kell azokat érnie, a két „világ" jó tulajdonságait egyesíti. Ha az ügyfélgép véletlenül meghibásodna, az információk egy központi helyen el vannak mentve, egyébként pedig gyorsan, helyben elérhetők. Az adatok a felhasználók számára észrevétlenül másolódnak át a különböző gépekre, így maradnak szinkronban.

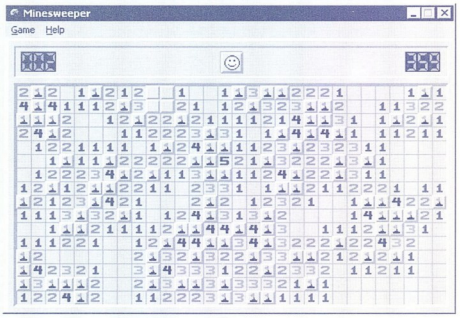
A tárolókapacitás tehát központosul? Általában igen. A számítási teljesítmény viszont nem.

Microsoft

Dupla pezsgő

K: Kezdeti sikerek után rendre elakadok a Minesweeperrel végzett munkámban. Noha a Beginner általában sikerül, az Expert szinte soha, mert a legutolsó néhány bomba sajnos nem egyértelműen rejtőzik a mezők alatt. Ilyenkor becsukott szemmel, imát mormolva kattintok egyet, de ez a módszer gyakran nem jön be. Csalódott és kielégítetlen vagyok.

V: Bizony sokan nehezen jutnak túl az informatikussá válás eme nehéz fázisán. Valóban az a legbosszantóbb eset, amikor már csak négy mező van hátra az Expert kirakásához, mint az alábbi ábra esetében:



Ha kiemeljük a problémás részt, rövid gondolkodás után rájöhetünk, hogy itt patthelyzet alakult ki. Két akna van hátra, ám azok egyforma eséllyel helyezkedhetnek el a bal alsó+jobb felső átlón, valamint a jobb alsó+bal felső vonalban. Ennyire kényes helyzetekben csak a NetAcademia, mint kiemelt megoldásszállító segíthet! Ha valami nem megy, az csak átmeneti állapot lehet, esetleg saját tudatlanságunk mocsarában vergődünk, mint jelen esetben is. A NetAcademia rendszermérnökei a legkorszerűbb módszerek igénybevételével *(Internet böngészése stb.)* jutnak hozzá létfontosságú információkhoz, amelyeket azután rendszeresen megosztanak a magyar IT társadalommal. Sajnos Redmond egyelőre nem ismeri el erőfeszítéseinket, pedig a Winmine-probléma megoldása fontosságban vetekszik holmi primitív kék halálok elhárítási trükkjeivel; szerintünk e tudásnak egyértelműen a Knowledge Base adatbázisban a helye!

A megoldás egyébként rém egyszerű. Szorult helyzetünkben gépeljük a Minesweeper arcába a következő karaktersorozatot:

```
XYZZY<Enter><Bal Shift>
```

(Természetesen a kacsacsőrök nélkül!)

E billentyűkombináció ismerete sokkal fontosabb, mint a Ctrl+Alt+Del-tudás, hisz ez magát a mission critical Winmine.exe-t kapcsolja át debug üzemmódba! Ezután a képernyő bal felső sarkában lévő képpont *(magyarul pixel)* kigyullad, és annak megfelelően villog, hogy az egérkurzorral vajon akna felett járuk-e. Ha igen, elfeketedik! :-O

Így már gyerekjáték a fenti feladványt megfejteni:

A hála, kegyelet és megemlékezés csokrait a szerkesztőségbe kérjük...

Troubleshooting

Sajnos hiba nélkül nincs haladás, a Winmine-paradoxon sem fog mindenki számára azonnal megoldódni. Ha a pixel nem gyullad ki, annak két oka lehet

1. Elgépelődött a varázsbillentyűkód. Try again.
2. A bal felső pixel kilóg a képernyőről. Megoldhatatlan probléma, ilyenkor feltétlenül forduljunk szakemberhez. *(Például megkérhetjük a takarítónénit, hogy ismételje meg azt a techno-portörlést, amitől kiment a kép a monitor kávájáról, s közben próbáljuk meg ellesni, és feljegyezni, hogyan csinálja!)*

BUÉK!

Microsoft® SQL Server

Microsoft Back Office® Family Member

1994-ben ismerkedtem meg a Microsoft SQL Server-rel, 4.0 a volt a verziószáma. Mai szemmel nézve persze döbbenetesen primitív volt, ám már akkor is rendelkezett azokkal a képességekkel, amelyek miatt az ember örömmel dobta el a Clipper-t: tranzakciókezelés, ügyfél-kiszolgáló felépítés, többszálú végrehajtás stb. Félelmetesen összetett banki rendszereket készítettünk akkoriban 486DX4-100 processzoros, 16 megabájt memóriával felvértezett „szervereken". Az elmúlt évek során az SQL Server megfiatalodott: fürge lett és erős. S ahogy verzióról verzióra fiatalodott, úgy vált egyre okosabbá is. Zseniális lekérdezés optimalizálója segítségével szakavatott kezekben szárnyakat kap. A TPC-C teljesítményteszt világbajnoka rendszerfelügyeleti szempontból is kiváló termék. A 7.0 már lehetővé tette az adatbázis által összegyűjtött adatok okos elemzését, hisz megjelent benne az OLAP kiszolgáló. S ami napjainkban lázba hoz, az a mesterséges intelligencia határmezsgyéjén található adatbányászat. Az adatok rejtett összefüggéseinek feltárása minden adatbázis tartalmát életre kelti. Januárban a NetAcademia Kft, a BST Kft. és a Microsoft Magyarország közös szervezésében megrendezésre kerülő

SQL Server Napon

megpróbálok valamennyit átadni azokból a tapasztalatokból, melyek az évek során „rám rakódtak". Természetesen egyetlen nap nem lehet elegendő a teljes információhalmaz átadására. Mindenkit szeretettel várok SQL Server 2000 tanfolyamaimon, melynek helyszíne a Business Software Training Hivatalos Microsoft Oktatóközpont (CTEC).

Jelentkezés a konferenciára és részletes tudnivalók a
http://www.netacademia.net/SQLnap
vagy
http://www.bst.hu/SQLnap
címen!

Az SQL Server nap rövid tematikája a következő:

- ***Adatbázisok a rendszergazda hozzáértő kezében.*** *Azaz mi mindent tehetünk a teljesítmény fokozása érdekében még akkor is, ha készen vett alkalmazással van dolgunk?*
- ***Az SQL 2000 újdonságai programozók számára.*** *Ismerkedjünk meg az SQL 2000 fejlesztőknek szóló fantasztikus újdonságaival: XML támogatás, felhasználói függvények, indexelhető nézetek, instead-of triggerek, Kaszkád referenciális integritás*
- ***Adatbányászat*** *A mesterséges intelligencia határmezsgyéjén húzódó technológia az adatok között megbúvó rejtett összefüggések feltárására.*
- ***Online Analitical Processing.*** *Adatelemzés mesterfokon*

BST Business Software Training

Hivatalos SQL 2000 Server vizsgaelőkészítő tanfolyamok a NetAcademia Kft. és a Business Software Training Hivatalos Microsoft Oktatóközpont (CTEC) közös szervezésében:

2072 Administering a Microsoft SQL Server Database, 5 nap. Időpont: 2001 január 22-26.

2073 Programming a Microsoft SQL Server Database, 5 nap Időpont: 2001 február 5-9.

Jelentkezés:
www.netacademia.net, illetve **www.bst.hu**

www.netacademia.net
A legjobbakat tanítjuk.
1105 Budapest, Ihász utca 13. • Tel.:263-2732

NETACADEMIA

tech.net

ISSN 1586-5185
12
9 771586 518005