Microsoft® TechNet

100% technológia ▪ 0% marketing

Hálózati terheléselosztás

ISA Server 2004 Enterprise

Jogosultságok — SharePoint Portal Server 2003

Az Active Directory Service Interfaces és a .NET

VI./2. szám
2005. április

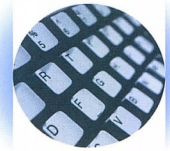
ISSN 15865185 02
9 771586 518005

Ne álljon meg a me.NET!

Microsoft GOLD CERTIFIED Partner | Learning Solutions

Rendelkezik már alapvető .NET programozási ismeretekkel, de úgy érzi ez kevés? Lelkes amatőr, de többre vágyik?

A SZÁMALK Továbbképzés áprilistól számos Microsoft .NET képzést kínál a fejlesztői ismereteiket bővíteni vágyóknak.



Minél több tanfolyamon vesz részt, annál kedvezőbb áron juthat hozzá! Akár több százezer forintnyi kedvezmény!

SQL 2000 programozás (2073): . . . . . 2005. április 11–15.
Adatkezelés, ADO.NET (2389): . . . . . 2005. április 25–27.
Windows alkalmazások fejlesztése (2555/2565): 2005. június 6–10.
Webes alkalmazások fejlesztése (2310): . . . . . 2005. július 11–15.
XML webszolgáltatások fejlesztése (2524): . . . . . 2005. szeptember 5–7.

Nálunk a tanfolyami járatok biztosan indulnak!

Tovább információk:
SZÁMALK Oktatási Rt. Továbbképzés, 1115 Budapest, Etele út 68.
Simon Ferenc, 203-0304/3050 mellék, simonf@szamalk.hu
www.szamalk.hu/tisza

SZÁMALK

Partner az oktatási megoldásokban

Microsoft TechNet

TechNet Magazin
VI. évfolyam, 2. szám
2005. április

Szerkesztőség és kiadó:
Microsoft Magyarország Kft.
1031 Budapest, Graphisoft park 3.

Felelős kiadó:
Székely Tamás marketingigazgató

Szerkesztő:
Takács Gitta (Epszilon Press)

Szaklektor:
Fóti Marcell (Netacademia)

Lapterv és nyomdai előkészítés:
Dobák Ildikó
Pataki Bernadett
(Ars Luna Bt.)

Borítófotó:
Archív

Nyomda:
AduPrint Kiadó és Nyomda Kft.
1033 Budapest, Csikós utca 8.
Felelős vezető: Tóth Béláné

Webcím:
www.microsoft.com/hun/technet/
E-mail:
technetmagazin@microsoft.hu

ISSN 1586-5185

Az integrált kommunikáció jövője

Egy nemzetközi webkonferencián Bill Gates felvázolta, hogy milyennek képzeli az integrált kommunikáció jövőjét. A Microsoft többféle technológia alkalmazása révén igyekszik megvalósítani az integrált kommunikációt: minden termékébe beépíti a jelenlét-észlelési funkciót, úgy integrálja egymással a különböző kommunikációs csatornákat (az e-mailt, a telefont, az azonnali üzenetet, az SMS-t, a videokonferenciát és a webkonferenciát), hogy problémamentesen lehessen váltani a különböző csatornák között, továbbá olyan intelligens szoftvert biztosít, amely a felhasználó elérhetőségének és egyéni beállításainak megfelelően képes kezelni a kommunikációt.
Az új integrált kommunikációs megoldások legfontosabb előnyei:

- **Sokoldalú jelenlét-észlelés** – A Microsoft szoftverekbe épített jelenlét-észlelésnek köszönhetően az infómunkások ellenőrizhetik az egyes személyek elérhetőségét, mielőtt kommunikációt kezdeményeznének velük.
- **Egységes kezelőfelület** – A valós idejű kommunikáció különböző csatornáinak (e-mail, telefon, azonnali üzenetek, SMS, videokonferencia és webkonferencia) egyesítése révén az infómunkások mindig az adott helyzetben optimális kommunikációs módot választhatják, problémamentesen válthatnak a különböző csatornák között, de akár egyszerre is használhatják őket.
- **Beépített intelligencia** – A kommunikációs eszközökkel integrált szoftverek automatikusan kezelni tudják a kommunikációs funkciókat, ismervén a felhasználók beállításait, fizikai helyét, szervezeti viszonyát és közös témáit. A szoftveres intelligencia segítségével a felhasználók saját környezetüktől és a hívó féltől függően szabályozni tudják, hogy hogyan irányítsa hozzájuk a rendszer a bejövő üzeneteket.

A Microsoft Office Rendszer új valós idejű együttműködési termékei és szolgáltatásai: Microsoft Office Communicator 2005 (korábbi kódnevén „Istanbul") – új integrált kommunikációs alkalmazás, a Microsoft Office Live Communications Server 2005 legújabb ügyfélprogramja. A Communicator sokoldalú jelenlét-észlelési funkciókat nyújt, egyetlen alkalmazásban kombinálja a valós idejű kommunikációs csatornákat, emellett módot ad a PC és a telefon integrálására is

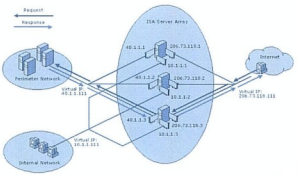
- **Microsoft Office Live Communications Server 2005 Service Pack 1** – a Microsoft valós idejű kommunikációs platformjának frissítése. A szervizcsomag tartalmazza a Microsoft Office Communicator 2005 támogatását, tökéletesebb védelmet nyújt a „spim" (kéretlen azonnali üzenetek - spam over IM) ellen, és a követelményeknek megfelelő kapcsolatteremtési lehetőségeket biztosít a Live Communications Servert használó vállalat, és az MSN, az AOL és a Yahoo! nyilvános azonnali üzenet-hálózata között. A Live Communications Server vállalati szintű azonnali üzenet-kezelő és jelenlét-észlelési szolgáltatást nyújt.
- **Microsoft Office Live Meeting 2005** – a Microsoft népszerű webkonferencia-szolgáltatásának jelentős továbbfejlesztése, amelynek segítségével a felhasználók online értekezleteket tarthatnak, és folyamatos együttműködést valósíthatnak meg anélkül, hogy egyszerre ugyanazon a helyen kellene lenniük. A legfontosabb fejlesztések közé tartozik az iparág első integrált konferenciahívás-vezérlője, amely az összes vezető audiokonferencia-szolgáltató számára biztosítja a Live Meeting összeköttetés Office alkalmazásokból való indítását, valamint a hét újabb nyelvi verzió elérhetőségét.

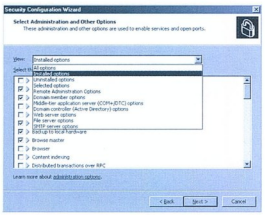
Tartalom

ISA Server 2004 Enterprise Edition

Hálózati terheléselosztás

A Microsoft ISA Server 2004 Enterprise Edition új megvilágításba helyezi a tűzfalainkon alkalmazott hálózati terheléselosztást (NLB). Ez a technológia nagy rendelkezésre állást és skálázhatóságot tesz lehetővé az akár 31 tagúvá bővíthető kiszolgálófarmok számára.





Windows Server 2003SP1

Mi várható?

Jelen cikk írásakor már megérkezett a publikus Windows Server 2003 SP1 RC2, túl vagyunk egy közel másfél éves tesztelési intervallumon és talán a cikk megjelenésekor már letölthető is lesz az új szervizcsomag végleges változata, amely az XP SP2 mintájára szintén több lesz, mint egy szimpla javításhalmaz.

Active Directory hozzáférés .NET komponensek segítségével

Az ADSI és a .NET

A .NET Framework igen könnyen használható formában biztosítja a címtár elérését (nemcsak az Active Directoryét), listázhatjuk a kiválasztott objektumokat, tetszőleges objektumot hozhatunk létre, és lekérdezhetjük, módosíthatjuk tulajdonságaikat is.

Kalandozás az Internet Explorer körül

Miért jó, illetve miért lehetne még jobb?

A kérdés talán költőinek hat, bár nem annak szántam. Úgy érzem, érdemes foglalkozni a Microsoft böngészőjével, hiszen egy sor olyan előnnyel rendelkezik, amelyeket nem szívesen áldozunk fel, csak azért, mert van pár olyan hiányossága, ami azért kezelhető.

ASP.NET 2.0 (Whidbey)

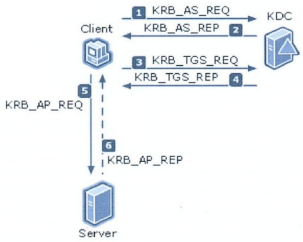
Mi várható a 2005. évi ASP.NET-ben? III. rész

Életszagú teljesítményfokozás az ASP.NET Cache szolgáltatásaival

Windows szolgáltatások 6. rész

A LocalSystem

Haladva a kirakós játékban, ebben a cikkben 15 újabb szolgáltatás ismertetése következik, persze továbbra is kizárólag a Windows Server 2003 Standard verziójának alapértelmezett telepítése során felkerülő szervizek közül. Már csak két rész van hátra ahhoz, hogy a közel 100 alapszolgáltatás mindegyikét megismerjük



Ami a hivatalos Microsoft tanfolyamokból kimaradt...

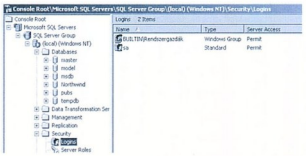
SharePoint Portal Server 2003 – jogosultságok

Rovatunkban újra a „csinálj magad intranetet" témát járjuk körbe. A Windows SharePoint Services és a SharePoint Portal Server 2003 jogosultságait futjuk át röviden, megnézzük milyen lehetőségeink vannak felhasználóink jogosultságainak kezelésére, meghatározására. Hogyan tudunk csoportokat kezelni és ezeket hozzáférési, műveleti jogokkal felruházni? Mindez hogyan hat a csoportmunka-helyekre?

Dr. Watson

Napi feladatok az SQL Server 2000-rel

Sok cikk jelent már meg SQL Server témakörben, de eddig csupán fejlesztői szemszögből vizsgáltuk az adatbáziskezelést. A többség számára ez a megközelítés teljesen haszontalan. Ők azok az adatbázis-rendszergazdák, akik egy módosíthatatlan készterméket üzemeltetéséért felelősek, abból kell kihozniuk a maximális teljesítményt, azt kell karbantartaniuk stb. Nekik szól ez a cikk.



ISA Server 2004 Enterprise Edition

Hálózati terheléselosztás

A Microsoft ISA Server 2004 Enterprise Edition új megvilágításba helyezi a tűzfalainkon alkalmazott hálózati terheléselosztást (NLB). Ez a technológia nagy rendelkezésre állást és skálázhatóságot tesz lehetővé az akár 31 tagúvá bővíthető kiszolgálófarmok számára.

Az NLB rendkívül jól használható minden olyan kiszolgáló-alkalmazásban, amelynek működése nem függ az alkalmazást futtató kiszolgáló hálózati identitásától. A Windows NLB-farmhoz kapcsolódó kliensek, és ugyanígy a szerverfarm tagjain futó alkalmazások sem szereznek tudomást a normális hálózati kommunikáció során arról, hogy a kiszolgálófarm tagjaiként együttműködő számítógépek önálló identitással rendelkeznek. A terheléselosztó farm használata nem új keletű megoldás a tűzfalakkal foglalkozók számára, mondhatnánk, hogy ez már egy kicsit „lerágott csont". Hogy miért nincs ez teljesen így, azt a Microsoft ISA Server 2004 Enterprise Edition-ban alkalmazott, teljesen újszerű megközelítés világítja meg számunkra. Az ISA új verziójában a tűzfalszoftverrel integrált, a tűzfal által menedzselt NLB megoldást kapunk.

Alapvetően kétféle lehetőség kínálkozik az NLB kialakítására:

- Az egyik a „hagyományos" módszer, amelyben az ISA tulajdonképpen nem is tud arról, hogy egy kiszolgálófarm tagjaként működő Windows-on fut. Ebben az esetben az NLB-t a már unásig ismert felületen konfiguráljuk, nagy küzdelmet víva több hálózati kártyás megfejtésekkel. Ennek a megoldásnak határozott előnye, hogy független az ISA verziójától (megcsinálhatjuk Standard Edition-nel is), konfigurációjától. Hátránya viszont, hogy az ISA szabálykészletek szinkronizálása a farm tagjai között nincs megoldva.
- Az új verzióban azonban új megoldás is van az NLB kialakítására, valamint a konfiguráció kezelésére is, más szóval az ISA 2004 integrált NLB támogatással rendelkezik. Ez a megoldás nagymértékben egyszerűsíti a több kártyával rendelkező gépekből épülő kiszolgálófarm kialakításával járó bonyolult beállítások áttekintését, hátránya viszont, hogy csak az ISA Enterprise verziójával működőképes.

A hálózati terheléselosztás lehetővé teszi a kiszolgálófarm összes tagja számára egy adott alhálózaton a fürt közös (Cluster IP) IP címére érkező forgalom feldolgozását. Minden egyes tagon az NLB meghajtó filterként funkcionál a hálózati csatoló illesztőprogramja és a TCP/IP réteg között. Az ISA 2004 is ezen a ponton illeszkedik a forgalomba lehetővé téve a többhálózatos komplex környezetekben is a terheléselosztás használatát. Ha a megvalósításban az NLB-ISA integrált megoldást választjuk, a terheléselosztást per-hálózat alapon szabályozhatjuk, azonban ebben az esetben hálózatonként nem használhatunk több hálózati csatolót.

Kiszolgálófarm-tagok közti (Intra-Array) kommunikáció

Ha ISA integrált NLB megoldást alkalmazunk, minden ISA számítógépünkbe további egy hálózati kártyát kell szerelnünk, melyet a farm tagjai az egymás közti kommunikációra fogják használni. Erősen javasolt ezeket a csatolókat szeparált hálózaton, dedikált aktív eszközön keresztül összekapcsolni; ez a megoldás növeli a rendszer teljesítményét, valamint fokozza a biztonságot. A tagok közötti belső kommunikációt ezután az új interfész használatára kell állítani.

Az NLB-integráció alapértelmezésben nincs engedélyezve a telepítés után. A funkció bekapcsolásának módszerére később még visszatérünk. Miután engedélyeztük az integrációt egy ISA tömbre, minden olyan tömb-szintű hálózatra különkülön beállíthatjuk az NLB működését, amelyik fizikailag is kapcsolódik a tömb tagjaihoz. Semmiképpen ne engedélyezzük olyan hálózatokon, amelyekhez nem tartozik fizikai kapcsolat, mert az helytelen működéshez vezet. Az ajánlás ezzel kapcsolatban egyértelmű és egyszerű: ha bekapcsoljuk a terheléselosztást, tegyük azt meg minden fizikai kapcsolattal rendelkező hálózaton, kivéve a tömb tagjai által belső kommunikációra használt (cluster network) hálózatot. Az NLB bekapcsolása után minden hálózaton meg kell adnunk a kiszolgálófarm tagjai által közösen használt Virtuális IP címet (cluster IP). Később részletezzük, hogyan is kell ezt csinálni.

Az ISA csomagszűrő rétegébe épített analizáló-motor vizsgálatot hajt végre a beérkező csomagokon, ezért az ISA együttműködik az NLB-vel annak érdekében, hogy a bejövő és a kimenő forgalom minden egyes kapcsolati szál (session) esetében az ISA tömb ugyanazon elemén haladjon keresztül. Ez egy fontos része az integrációnak, mivel e nélkül nem lehetne alapos csomagvizsgálatot végrehajtani.

Terheléselosztás és virtuális magánhálózatok

Amikor egy távoli kliens VPN kapcsolatot kezdeményez egy ISA tömbbel, a tömb tagjai közül csak az egyik építi fel a kapcsolatot, és IP címet allokál a kliens számára. Ettől a ponttól kezdve a távoli kliens minden forgalma ezen a tagon keresztül halad. Ha VPN-t és NLB-t is használni szeretnénk egy ISA tömbön, fontos a tűzfal-házirend készítésénél odafigyelni néhány részletre. Az ilyen szituációkban nem alkalmazhatunk felhasználó-azonosításon alapuló szabályokat a vándorló VPN klienseink számára, egész pontosan olyat nem, amelyik egy felhasználó számára engedélyez valamilyen forgalmat. Ennek az oka egyszerű: elképzelhető, hogy másik ISA szerver kezeli a VPN kapcsolatot a klienssel, mint amelyik a kliens kérésének kiszolgálását hajtja végre. Ebben az esetben, ha a kérés teljesítéséhez azonosításra van szükség, a VPN felhasználó azonosítói nem továbbítódnak és a kérés teljesítése megtagadásra kerül. Egy kivétel van ez alól, a HTTP protokollra vonatkozó szabályok csoportja, mivel a felhasználói azonosítás ott ebben az esetben is tökéletesen működik (mert a web-proxy bezavar a képbe).

Telephelyek közti kapcsolatok

Ha az ISA tömbünkön engedélyeztük az NLB-t egy távoli telephely-kapcsolatban, a tömb egyik tagja automatikusan hozzárendelődik az adott VPN csatornához, ezért a telephelyek között nem alakulnak ki párhuzamos csatornák. Abban az esetben azonban, ha a csatornát üzemeltető tag kiesik a műszakból, a VPN csatorna automatikusan felépül egy másik tagon.

Két tömb is összekapcsolható telephelyek közti VPN segítségével, ebben az esetben azonban mindkét oldalon ismerni kell a túloldali dedikált IP címeket is. Az ISA szerver NLB-funkcionalitását használhatjuk a Windows Server 2003 NLB beállítások menedzselésére az ISA tömb tagjain. Az integrált NLB megoldás további előnye, hogy a kliensek (a virtuális IP-re érkező) kérése automatikusan továbbítódik a VPN-t üzemeltető ISA taghoz, valamint ennek hibája esetén a kapcsolat (és ezzel együtt a kliens forgalom is) automatikusan áttelepül egy másik működő gépre. Van azonban néhány megfontolandó kérdés az NLB és a telephelyek közti kapcsolatok kialakításával kapcsolatban:

- Több ISA-ból álló tömb esetén, ha NLB-t szeretnénk használni, muszáj az integrált megoldást választani, ellenkező esetben a telephelyek közti VPN nem fog működni.
- Ha ISA Integrált NLB-t használunk (mást pedig, ugye, nem lehet) VPN környezetben, a külső interfészen (vagy azon, amelyik fogadja a VPN kapcsolatokat) kötelezően engedélyezni kell a terheléselosztást. Ugyanígy kell eljárnunk minden olyan hálózat esetén, ahonnan vagy ahová a VPN kliensek forgalmát engedélyezni kívánjuk.
- Ha több szerveres környezetben NLB-t engedélyezünk és telephelyek közti VPN-t is használunk, az ISA Configuration Storage Server (CSS) funkciót át kell helyezni egy olyan számítógépre, amelyik nem tagja semmilyen ISA tömbnek, vagy egyáltalán nem futtat ISA szolgáltatásokat (lehetőleg valamelyik védett hálózatba). Ha a CSS az ISA tömb valamelyik tagján lenne, és ez a tag történetesen nem lenne azonos a VPN csatornát üzemeltető géppel, a túloldali VPN kiszolgáló elvesztené a kapcsolatot a CSS tárolóval, mivel olyan helyen keresné, ahol nincs is.

Virtuális IP címek

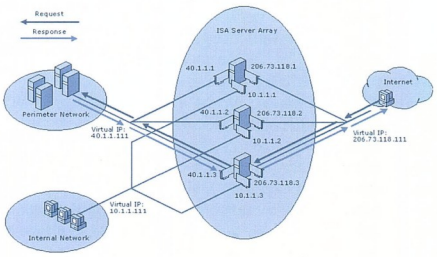
Amikor a terheléselosztást engedélyezzük, további IP címet kell hozzárendelnünk az adott hálózathoz. Amint ezt megtesszük, az ISA szerver önállóan automatikusan módosítja a hálózati objektum tulajdonságát, csakúgy, mint a TCP/IP beállításokat a Windows hálózati beállításokban. Minden hálózati csatolón tehát, amelyen az NLB-t bekapcsoltuk, feltalálható két IP cím: egy az újonnan hozzárendelt „virtuális" IP cím és az eredeti „dedikált" IP cím is. ISA integrált NLB módban tehát minden csatolónak rendelkeznie kell egy saját dedikált címmel. A dedikált és a virtuális címnek ugyanabban az IP hálózatban kell lennie, valamint az alhálózati maszkoknak is egyformának kell lenniük.

Egy csatoló több IP cím

Néhány esetben olyan megoldásra lehet szükségünk, amikor az ISA farm egy hálózati kapcsolatához több IP címet kell rendelnünk. Ilyen eset lehet például, ha a külső interfészen több szolgáltatást is meg szeretnénk hirdetni, de ezeket szolgáltatásonként különböző IP címen (pl.: egy mail- és egy webszerver külön IP-n). A több IP címmel rendelkező hálózati kapcsolatokat az ISA integrált NLB megoldás nem tudja igazán kezelni. Maga az NLB ugyan kifogástalanul működik több címen is, azonban az ISA menedzsment felület nincs felkészítve az ilyen helyzetekre. Kétségbe esni nem kell, mindent pontosan úgy kell csinálni (integrált NLB módban), ahogy a szimpla – egy IP címes – esetben is, majd amikor elkészültünk a beállításokkal, előkapjuk a Windows hálózati beállításokat, és a már meglévők mellé felvesszük a további szükséges címeket is. A lényeg: az összes további IP címet csak akkor szabad felvennünk, ha az ISA integrált NLB konfigurálással végeztünk. Ehhez hasonlóan kell eljárnunk visszafelé menet is, ha nem akarjuk felborítani a tömbünk konfiguráció-tárát, azaz, ha el szeretnénk távolítani az NLB-t egy hálózatról, először a pluszban hozzárendelt címeket kell leszednünk. Tehát jegyezzük meg, hogy a hálózati kártyán elsőnek beállított IP cím automatikusan dedikált IP-vé, a többi virtuális IP-vé válik, de ezek közül csak az első (szummában a második) kerül automatikusan az ISA menedzsment konzolból a Windows hálózati beállításokba.

Kétirányú affinitás

Az ISA szerver NLB módban mindig automatikusan kétirányú affinitást állít be a kapcsolatok kezelésére. A legtöbb esetben a szimpla affinitás (ezt tudja ugye a Windows NLB alapból) nem hatékony megoldás. A legegyszerűbb példa erre, ha megnézünk egy egyszerű szerverpublikálást: Egy belső védett hálózatban működő szolgáltatást meghirdetünk a külső interfészen.



1. ábra Kétirányú affinitás

Ebben az esetben az NLB-t érdemes beállítani mind a külső – az internet felé néző –, mind a belső – a szerver felé néző lábon beállítani.
Mivel a meghirdetett szolgáltatást futtató szerverünk SecureNat kliens, alap értelmezett átjáróként a kiszolgálófarm belső, virtuális címét kell megadnunk. A helyes működéshez azonban szükséges, hogy a szerver válaszai mindig a kiszolgálófarm azon tagján keresztül jussanak el a klienshez, amelyen a kérés érkezett, mivel ez az egyetlen ISA szerver a farmban, amely tárolja az adott kapcsolati szálhoz tartozó állapot-, authentikációs és egyéb információkat. Ennek a problémának a megoldására vetjük be a kétirányú affinitást, melynek működését az 1. ábra szemlélteti.

Egyedi azonosítók a kiszolgálófarmban (Host ID)

A telepítés során – előlünk rejtve – automatikusan generálódik minden ISA kiszolgálófarm-tag számára egy egyedi azonosító (2-től 32-ig), ez azonosítja az egyes gépeket a farmon belüli kommunikáció során. Ezt az értéket általában nem szabad megváltoztatnunk, bár előfordulhat olyan szituáció, amikor probléma lép fel, ilyen esetben az ISA szerver beleírja „bánatát" az eseménynaplóba, és megkér, hogy hárítsuk el a problémát. Az egyedi azonosítók átírása viszonylag egyszerű feladat:

- Az ISA menedzsment konzolban bontsuk ki a tömb elemeit, menjünk a (Configuration → Servers) batyura
- A jobboldali ablakban válasszuk ki a szerverünket, majd jobb klikk → Properties
- A „Host ID" fülön válasszunk a listából egy még nem használt azonosítót

Ne felejtsük el, hogy most hibaelhárítottunk, és az eredeti hiba miatt valószínűleg nem tudott elindulni a Microsoft Firewall Service, ezt ilyenkor manuálisan újra kell indítani.

Az NLB leállítása/elindítása

Az integrált NLB módban az ISA szerver felelős azért, hogy megállapítsa, hogy a farm minden tagja aktív, működőképes és a forgalom megfelelően áthaladhat minden hálózat között. Ha az NLB egy szerver legalább egy hálózatán nem működik, az ISA kikapcsolja azt az adott szerver minden hálózatán. Ugyanígy az ISA találja ki azt is, ha a tag megfelelően működik és visszakapcsolódhat a farmhoz. A következő ellenőrzéseket hajtja végre a döntéshez:

- A számítógép elérhető
- A „Microsoft Firewall Service" fut
- Az NLB fut minden hálózati csatolón
- Minden hálózathoz hozzárendelhető egy fizikai csatoló
- Az NLB úgy van beállítva a gépen, hogy csak az ISA szolgáltatások után induljon.

A terheléselosztás engedélyezése

A továbbiakban tehát nézzük át az integrált NLB kialakításának egyes lépéseit.
Az integrált terheléselosztás engedélyezéséhez a következő lépéseket kell végrehajtanunk:

- Az ISA menedzsment felületen bontsuk ki a tömb elemei batyut, válasszuk a (Configuration → Networks) helyet
- A részletek ablakban válasszuk ki a „Networks" fület, ezen belül a megfelelő hálózatot.
- A feladatpult részben kattintsunk a „Tasks" fülre, majd engedélyezzük az NLB-t az „Enable Network Load Balancing Integration" varázsló elindításával
- A „Virtual IP" részben adjuk meg a megfelelő IP címet (ez a kiszolgálófarm közös címe lesz az adott hálózaton)
- A „Mask" részbe a megfelelő alhálózati maszkot írjuk (ez kötelezően ugyanaz, mint ami a szerver dedikált IP-jéhez tartozik)

Apró kellemetlenség, hogy miután az összes hálózaton beállítottuk a terheléselosztást (kivéve természetesen a dedikált Cluster Network-öt) az ISA tömb összes tagját újra kell indítani.
Ha el kívánjuk távolítani az Integrált NLB-t szervereinkről, ez az előzőekben ismertetett helyen hajtható végre.
Először le kell tiltani az NLB-t az összes hálózaton, majd teljesen el kell távolítani az ISA tömbről (Array → Configuration → Networks → Tasks → Disable Network Load Balancing Integration)
Ha az ISA szerver szoftvert távolítjuk el a gépekről, az NLB beállítások nem törlődnek, és az NLBS meghajtó sem kerül eltávolításra a gépről.

Az integrált terheléselosztás vezérlése

Néhány fontos utasítást az ISA konzolról is kiadhatunk az NLB meghajtó számára, nem kell a parancssoros felületet használnunk. Sajnos nem minden NLB parancsot használhatunk, mert a feladatpult ide vonatkozó menüje csak akkor aktív, ha az NLB fut (tehát pl. elindítani nem tudjuk). A vezérlőparancs kiadásához menjünk el a (Monitoring → Services → NLB) batyuba, majd a feladatpulton a „Tasks" fület bökve kiválaszthatjuk a „Drain and Stop Selected Service" vagy a „Stop Selected Service" parancsot.

Kiszolgálófarm-tagok közti (Intra-Array) kommunikáció beállítása

A telepítési procedúra során egy cím automatikusan kiválasztódik a belső kommunikáció számára, ez a cím általában a belső hálózatba tartozó hálózati kártya címe. Ahogy az előzőekben is láthattuk, több szerveres környezetben a tagok közti kommunikációra dedikált adaptert használunk, ezért ezt a beállítást a telepítés után meg kell változtassuk. A lépések a következők:

- Állítsuk be az új adapter IP címét a belső kommunikációhoz: (Configuration → Servers → Aktuális tag → Tasks → Configure Selected Server → Communication → Use this IP address for communication between array members)
- Új hálózati objektum létrehozása: (Configuration → Networks → Tasks → Create New Network)
- Engedélyezzük a Web Proxy klienseket a hálózaton: (Configuration → Networks →Cluster Net → Tasks → Edit Selected Network → Web Proxy → Enable Web Proxy clients)
- Tiltsuk le a Firewall klienseket a hálózaton: (Configuration → Networks → Cluster Net → Tasks → Edit Selected Network → Firewall client → Enable Firewall client support for this network)

Tiba L. Tamás
tamas.tiba@trainerc.hu
MCSE+Security, MCT

Windows Server 2003SP1

Mi várható?

Jelen cikk írásakor már megérkezett a publikus Windows Server 2003 SP1 RC2, túl vagyunk egy közel másfél éves tesztelési intervallumon és talán a cikk megjelenésekor már letölthető is lesz az új szervizcsomag végleges változata, amely az XP SP2 mintájára szintén több lesz, mint egy szimpla javításhalmaz.

Ebben az évben számos újdonság kerül(het) be a Windows kiszolgálókat üzemeltetők látóterébe. Az itt **[1]** található vége-hossza nincs listában szó van sok új vagy megújított termékről, csomagról, kiegészítőről, de kiszolgálói vonalon kiemelkedik belőle három nagyobb súlyú termék. Az egyik az – előreláthatólag – 2005 második felében debütáló, jelenleg R2 **[2]** fantázianevű kiegészítés, amely nem javítócsomag, hanem valódi bővítmény, egy csokor „Feature Pack", olyan szolgáltatásokkal mint az ADFS, ADAM, Windows SharePoint Services, az új DFS, vagy a Print Management Consol, stb. Sajnos a NAP (Network Access Protection **[3]**) és az új TS funkciók kikerültek belőle, de várhatóan így is teli leszünk izgalmas, új lehetőségekkel. Az év végén viszont eljön a Longhorn Server Beta1 ideje is (és a kliensé is), valószínűleg még több változással és újdonsággal, de erről még korai nyilatkozni.

A harmadik fontos elem várható legkorábban, ez pedig a Windows Server 2003 első szervizcsomagja. A felsoroltak közül valószínűleg ez a legszürkébb, de így is be tudok számolni jópár érdekes és hasznos változásról, újdonságról. Szeretném az elején leszögezni, hogy a cikk a publikus és a nem publikus bétaváltozatok használata során szerzett tapasztalatokból származik, ám azt garantálni, hogy minden ugyanígy lesz a végső verzióban is, nyilván nem lehetséges.

Egyszóval: a biztonság

Ekörül forog még szinte minden, az SP1 legnagyobb számban megjelenő újításai (és persze a szokásos integrált frissítések is) erről szólnak. Jelentős mennyiségű az XP SP2-ben már megismert restrikció került bele ebbe a szervizcsomagba is, pl. a Windows Tűzfallal, az Internet Explorerrel (add-ons tiltás, ActiveX „BindToObject", information bar, pop-up blokkoló, zone elevation blocking, stb.) vagy a Outlook Expresszel kapcsolatban (pl. AES), de az RPC és DCOM szigorítások, a DEP illetve a TCP/IP kapcsolatok terén is ugyanazok a szigorúbb elvek érvényesülnek, ám ezekről most nem lesz szó, viszont nézzünk egy felsorolást azokról az újdonságokról, amelyeket megemlítünk:

- Telepítés utáni automatizmusok (Windows Firewall + Post-Setup Security Updates)
- Állomány- és mappa hozzáférés változások (Access-based Directory Enumeration)
- Security Configuration Wizard
- Wireless Provisioning Services
- Ömlesztett újdonságok (Network Access Quarantine Control, DCdiag, RIS, RsoP, stb.)

A telepítés utáni automatizmusok apropóján

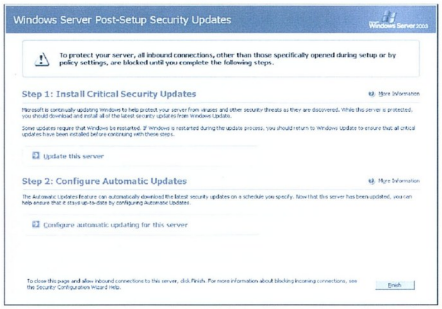
Egy sommás megállapítással kezdeném: hogy egy szűzen (tűzfal/vírusirtó/antispyware nélkül) az internetre kötött Windows operációs rendszer mennyi ideig képes biztonságban működni, az vita tárgya szerteszét a világban, viszont a sebezhetőség realitása nem az. Ezért fontos teendő volt, hogy a kliensek (XP SP2) után a kiszolgálók sebezhetősége is erőteljesen javuljon egy ilyen szélsőséges szituációban. A megoldás két részből áll: egyrészt a Windows tűzfal automatikus és az indítási folyamat egész alacsony szintjén történő bekapcsolásából másrészt az ún. PSSU (Post-Setup Security Updates) technika alkalmazásából.

A Windows tűzfalat jól ismerhetjük már az XP SP2-ből, ezzel kapcsolatban gyakorlatilag nem látszik változás. Az viszont, hogy mennyiben szolgálja a korai védelmet, rögtön kiderül, ha megnézzük, hogy hogyan és mikor indul el. Ugyanis korábban (akármilyen csúcs védelemmel rendelkeztünk) mindig létezett egy lyuk a hálózati szolgáltatások és az ICF vagy más tűzfal bekapcsolása között (kivéve pl. ISA 2004, amely bebújik a TCP/IP réteg alá és így már az indítás összes fázisában el tudja hessegetni az illetékteleneket). Ennek a holtidőnek a léte nem túl szerencsés, de muszáj volt, hiszen a tűzfal-alkalmazásnak be kellett várnia a többi rendszerszolgáltatást a függőségi viszonyok miatt. Az SP1 telepítése után viszont rögtön életbe lép egy statikus szabály alapú állapotszűrés (ún. rendszerindítási házirend), ami a DNS/DHCP protokollok mellett átengedi a csoportházirend érvényre jutásához szükséges forgalmat is, de semmi mást nem. Ez a házirend csak addig

él, amíg a Windows Tűzfal nem kezd el működni, és van még egy fontos tulajdonsága: nem lehetséges módosítani.
A másik lényeges elem a kötelező felhívás az azonnali frissítésre, amely az üzemeltető első bejelentkezése alkalmával elindul. Ezzel egy időben a tűzfal az összes bejövő kapcsolatot ignorálja, mindaddig amíg be nem zárjuk ezt az ablakot, ergo biztonságos körülmények között letölthetjük és telepíthetjük a frissítéseket, majd mehet minden tovább. Azonban van egy kivétel, pontosabban kettő: ha korábban a csoportházirend segítségével kivételeket állítottunk be a tűzfallal kapcsolatban, vagy a telepítés során engedélyeztük a Remote Desktop-ot, akkor ezeket a kapcsolatokat a lezárás alatt is engedélyezi a rendszer.
A PSSU nem érhető el átlagos módon pl. a Start menüből. Csak a Windows Server 2003 olyan teljes telepítésére vonatkozik, amely szervizcsomagot tartalmaz (pl. az SP1-gyel integrált verzió). Akkor sem jelenik meg a PSSU, ha a következő operációs rendszerekről frissítünk:

- Windows 2000-ről a Windows Server 2003 SP1-re
- Windows Server 2003-ról Windows Server 2003 SP1-re

Viszont ha egy Windows NT 4.0 rendszert frissítünk, akkor használható lesz ez a szolgáltatás. További tudnivaló: ha felügyelet nélküli telepítési szkripttel telepítjük Windows Server 2003 rendszert, vagy ha esetleg a csoportházirend segítségével engedélyezzük vagy tiltjuk a Windows Tűzfalat, akkor sem fog megjelenni az SP1 telepítése után.



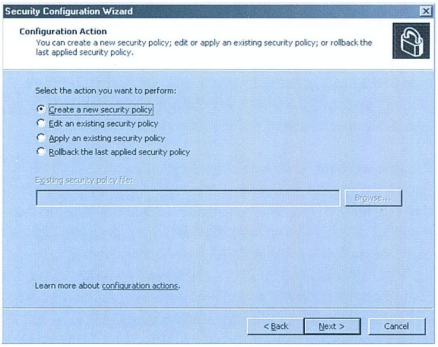
Felhívás a frissítések telepítésére

Az állomány- és mappa hozzáférés változások

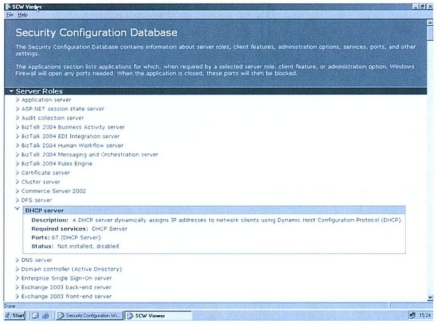
közül egyre már igen régóta szükség lett volna, sosem értettem miért is nincs meg ez a lehetőség a Windows kiszolgálók esetén. Hogy tisztuljon a kép, arról van szó, hogy a szerver megosztott mappáiban a felhasználó csak azt a tartalmat látja, amelyhez legalább olvasási jogosultsága van. Nos, ez az ún. „Access-based Directory Enumeration" lényege, amely az SP1 telepítése után a rendelkezésünkre áll. Ez az új lehetőség a megosztott mappában található állományokra és mappákra igaz, sajnos pl. a tallózásnál a megosztás mappák továbbra is látszanak bárki számára. Persze, ha egy megosztott mappa egy másikban helyezkedik el, akkor érdekes helyzet áll elő: a tallózott listában látszik, ám ha a preparált „szülő" megosztásba belemegyünk, akkor már nem. Alapértelmezés szerint ez a funkció nem áll rendelkezésre, hanem manuálisan kell bekapcsolnunk egyesével a megosztott mappákon. Elviekben ez az opció a NetShareSetInfo API **[4]** attribútuma (SHI1005_FLAGS_ENFORCE_NAMESPACE_ACCESS), amely bekapcsolására egy (egyelőre nem publikus) parancssori program áll rendelkezésre, amely használata nagyon egyszerű, meg kell nevezni az adott megosztást és 0/1-gyel jelölni a ki-bekapcsolást. Az még egyelőre nem tisztán látható, hogy a végső változatban ezt az attribútumot kivezetik-e a GUI-ra vagy marad a parancssori eszköz, mindenesetre kipróbáltam és tanúsíthatom hogy remekül működik, azaz tényleg csak azt látja a felhasználónk, amihez joga is van.

Valószínűleg a Security Configuration Wizard

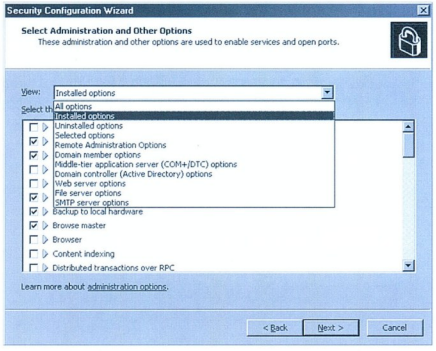
bizonyul a legnagyobb dobásnak ebben a szervizcsomagban. Az viszont biztos, hogy ez a legkomplexebb az újdonságok közül. Emlékszünk az IISLockdown-ra az IIS5-nél? Igazából a párhuzam kicsit esetleges, és csak a működési elv alapján húzható meg, azaz varázsló > sablonok > egyéb extra beállítások. Ám az SCW az egész kiszolgálóra vonatkozó biztonsági beállítások egyszerű beállításában segít, és valóban nagyon alapos. A varázsló egy bővíthető XML adatbázis alapján, a szerepkörök definiálása után állapítja meg, hogy a közel 60 féle kiszolgálói szerepkör közül (pl. Domain Controller, DHCP Server, DNS Server, WINS Server, File Server, Web Server, Exchange Back-End Server, Exchange Front-End Server, SQL Server, stb.) melyikhez milyen szolgáltatásoknak, portoknak és egyéb funkcionális elemeknek kell engedélyezve lenniük.
Ezután, a létrehozott biztonsági szabályok mentén, letiltja a szolgáltatásokat, zárolja a portokat, módosítja a registry tartalmát valamint behangolja a naplózási beállításokat. Persze, a nyitva hagyott portok is korlátozhatók meghatározott hatókörökre, illetve kiaknázva az IPSec lehetőségeit, megvédhetőek. Egy szó mint száz, az SCW minden olyan funkciót letilt, amelyre nincs szükség a kiszolgáló által betöltött szerephez. A varázslás az eszköz telepítésével kezdődik, a szokásos módon. Az indítása után rögtön négy lehetőségünk lesz, azaz készíthetünk egy új szabálycsomagot; belepiszkálhatunk egy létezőbe; alkalmazhatunk egy korábban elkészítettet; illetve visszaállíthatjuk a legutolsó állapotot.



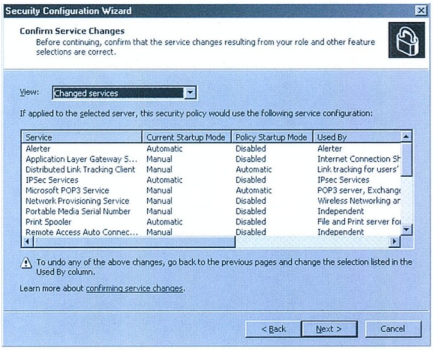
Ezután kiválaszthatjuk azt a szervert, amelyikkel dolgozni szeretnénk. Természetesen ez lehet egy távoli szerver is, azzal a feltétellel, hogy szinten telepítve van rajta az SCW (mind a négy alapművelet lehetséges lesz távolból is).



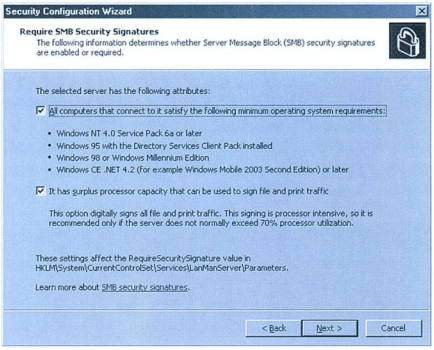
Amit viszont nem tehetünk meg az, hogy a 32 bites kiszolgálóról egy 64 bitest konfigurálunk és fordítva.
A következő lépésben jön a felmérés, majd a kapunk egy termetes leírást az összes szerver szerepkör jellemzőiről, a szükséges szolgáltatásokról és portokról, a kliens alkalmazásokról, egyszóval ez az ún. biztonsági konfigurációs adatbázis (lásd előző ábra).
Most kezdődik az igazi munka, első körben az SCW felajánl a vizsgálatok alapján egy szerepkör listát, amelyhez hozzátehetünk illetve elvehetünk attól függően, hogy milyen szerepet tölt be az adott szerver. Nyomban ezután jön a kiszolgáló kliens szerepeinek kiválasztása, ahogy az ábrán látszik is (ahogy az is, hogy a „View" alatt mindig kellemesen finomíthatjuk a látnivalót).



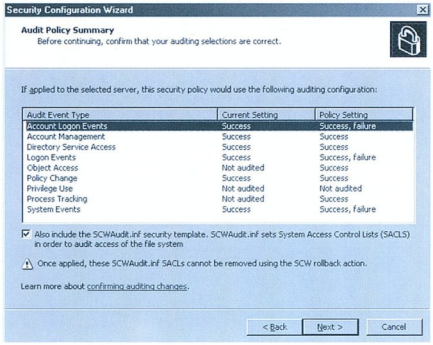
Ezt követi az egyéb gyári szükséges szolgáltatások illetve szervizek kiválasztása, majd külső szoftverek által telepített szervizek felsorolása. Ezután kérhetjük a nem használt szervizek letiltását és végül az első kör végén egy áttekinthető listában tekinthetjük meg az összegzést.



A második kör a hálózati biztonsági beállításokra vonatkozik, a portokat tilthatjuk illetve hatókör vagy az IPSec segítségével korlátozhatjuk az elérést. A harmadik körben a regisztrációs adatbázis néhány kiemelt kulcsa segítségével további fontos korlátozásokat eszközölhetünk. Első opció az SMB aláírás kikényszerítése, amellyel a közismert „The man in the middle" típusú támadást lehet kiküszöbölni. Óvatosan bánjunk ezzel, mert a régi kliensek ezt a módszert nem ismerik, ezért már a tartományba belépés sem sikerülhet az AD kliens nélkül. A másik opcióval az összes állomány- és nyomtatási forgalom digitális aláírása váltható ki.

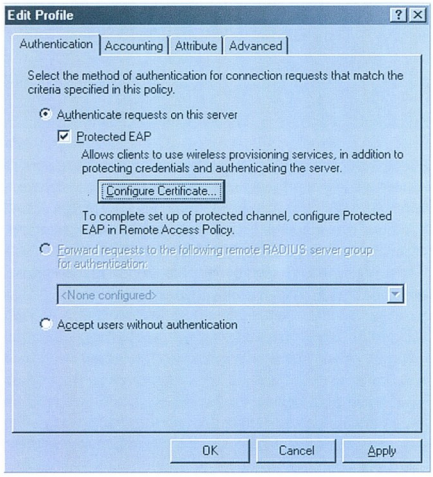


Ezután az LDAP signing (aláírás), azaz az LDAP forgalom titkosított és/vagy aláírt változatára térhetünk át (a kliensek a Windows 2000 SP3-tól ismerik) majd kiválaszthatjuk a LAN Manager hitelesítési eljárások számítógépekre alkalmazott változatai közül a megfelelőt (LM/NTLM/NTLMv2).
Végül a negyedik körben következnek a naplózással kapcsolatos beállítások és készen is vagyunk.



A folyamat végén (ha van IIS, akkor lesz még egy az IISLockdown-hoz hasonló 5. kör is), a beállításokat elmenthetjük (.xml), egy másik gépen alkalmazhatjuk, beimportálhatjuk, stb.

A vezeték nélküli hálózatlétesítési szolgáltatás, azaz a Wireless Provisioning Services az XP SP2-ben jelent meg először, ahol a WLAN kliens szoftver bővítményeként szerepel. A Windows Server 2003 SP1-ben pedig az Internet Authentication Service (IAS) szolgáltatást egészíti ki. Elsősorban hotspot szolgáltatóknak, WLAN hozzáférést nyújtó cégeknek készült, illetve nagyvállalati környezetben is használható, pl. vendéghozzáférés biztosítása céljából (akár automatikusan egy erre a célra tartott VLAN-ra átirányítva). Röviden összefoglalva ez a szolgáltatás leegyszerűsíti a WLAN kliensek életét, hiszen a kapcsolódáskor az IAS (mint proxy) közvetítésével automatikusan tölthetik le a hálózat adatait, paramétereit a kiszolgálótól, így beavatkozás nélkül csatlakoznak illetve váltanak a szolgáltatók között (barangolás). Egyúttal az SP1-ben némi wireless biztonsági szint emelés is történt, hiszen a Windows Server 2003 immár támogatja a Protected Extensible Authentication Protocol-t (PEAP) és a WPA-t is.



Ahhoz viszont, hogy az IAS-ban az előbbit az ábrán látható módon a grafikus felületen lássuk, némi registry módosításra van szükség. A következő helyen vegyük fel a „EnableWPSCompatibility" DWORD kulcsot 1-es értékkel és az IAS-ban és lőn.

```
HKLM\System\CurrentControlSet\Services\
RemoteAccess\Policy
```

A Microsoft egy nagyon alapos és hosszú dokumentumot máris kiadott az IAS illetve egyéb RADIUS szerverek valamint a WPS kapcsolatáról, amit innen **[5]** letölthetünk.

Végül, de nem utolsósorban

beszéljünk kisebb/nagyobb változásokról, fejlesztésekről, ömlesztve, ám nem súlyozva:

- 64 bit: az SP1 után a Windows Server 2003 a 64 bites hardverre is kiterjeszti a lehetőségeit
- Network Access Quarantine Control: azaz a VPN karantén, amely a VPN-en bejövő kapcsolatok ellenőrzésére szolgál. Ha az általunk megszabott kapcsolódási feltételeket (szervizcsomag, bekapcsolt tűzfal, aktuális frissítések, vírusirtó megkövetelése, stb.) a VPN kliens nem teljesíti, akkor a kapcsolódás megtagadása lesz az eredmény. A VPN karantén ISA 2004 esetén is létezik, és a Windows Server 2003 RTM-ben is működhet a Resource Kit Tools-ból feltelepítve, most viszont bekerült az egyből telepíthető komponensek közé. További info a VPN karanténról itt **[6]**, illetve letölthető példaszkriptek innen **[7]**.
- Ugyanígy „járt" az RsoP (Resultant Set of Policy = Eredő házirend), azaz az eddig csak bővítményként megjelent eszközt beintegrálták. (*Úgy néz ki, talán végre a GPMC is belekerülhet az R2-be*).
- Az SP1-es RIS támogatja a 64 bites image-eket.
- A Dcdiag.exe egy remek eszköz volt eddig is, ám most számtalan DNS szerver/zóna teszttel bővült:

```
Dcdiag /test:DNS [/DnsBasic | /DnsForwarders |
/DnsDelegation | /DnsDynamicUpdate |
/DnsRecordRegistration | /DnsResolveExtName
[/DnsInternetName:InternetName] | /DnsAll]
[/f:Logfile] [/ferr:Logerr] /S:DCName[/e] [/v]
```

- Az ADPrep.exe okosabb lett, bizonyos Exchange és (a Windows 2000) AD sémaelemek konfliktusát megelőzi azzal, hogy nem hajlandó addig futni, amíg ki nem javítjuk. A leendő „*Changes to Functionality in Microsoft Windows Server 2003 Service Pack 1*" című dokumentumban, részletesen le lesz jegyezve, hogyan korrigáljunk (manuálisan) ebben az esetben.

GÁL TAMÁS
MCT, MCSE, MCSA, MVP
gtamas@tjszki.hu

A cikkben szereplő URL-ek:

[1] http://bink.nu/?ArticleID=3126
[2] http://tinyurl.com/6kxv4
[3] http://tinyurl.com/5ae2j
[4] http://tinyurl.com/6lbdg
[5] http://tinyurl.com/4utbp
[6] http://tinyurl.com/eyna
[7] http://tinyurl.com/7275e

Active Directory hozzáférés .NET komponensek segítségével

Az ADSI és a .NET

A .NET Framework igen könnyen használható formában biztosítja a címtár elérését (nemcsak az Active Directoryét), listázhatjuk a kiválasztott objektumokat, tetszőleges objektumot hozhatunk létre, és lekérdezhetjük, módosíthatjuk tulajdonságaikat is.

Bevezetés

Számos olyan eszköz létezik, amelyekkel hozzáférhetünk az Active Directoryban tárolt objektumokhoz, módosíthatjuk tulajdonságaikat, vagy akár a teljes tartalmat fájlokba exportálhatjuk. Vannak parancssori eszközök is, amelyek igen rugalmasan paraméterezhetők és szkriptekből, vagy időzítve is jól használhatók.

Mégis egészen más „élmény" az, ha saját magunk írunk olyan programot, amely képes az AD írására és olvasására, mert ekkor gyakorlatilag MINDENT megtehetünk (persze csak a megfelelő jogosultságok birtokában), lehetőségeinknek csak a fantáziánk szabhat határt. Természetesen arra is lehetőségünk van, hogy programunk felhasználóit az AD hitelesítse, és a megfelelő jogosultságok kiosztásához lekérdezhetjük a hitelesített felhasználó csoporttagságát is.

Az **[1]** címről letölthető mintaprogram bemutatja a .NET Framework címtárakkal kapcsolatos legfontosabb funkcióit, mintát ad a fent említett műveletek végrehajtására. A program konzol alapú, menüszerkezete a következő:

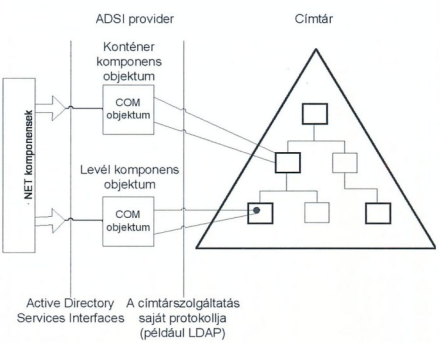
```
Parancssor - ad.exe

D:\>ad.exe
============================================================
0 - Felhasználók, csoportok és számítógépek listája
1 - Letiltott felhasználói fiókok listája
2 - Új felhasználó
3 - Új számítógépfiók
4 - Felhasználó törlése
5 - Számítógépfiók törlése
6 - Felhasználó tulajdonságainak listája
7 - Számítógép tulajdonságainak listája
8 - Felhasználó autentikációja
9 - Kilépés
```

A mintaprogram menüszerkezete

Active Directory Service Interfaces, ADSI

Az ADSI segítségével a különféle címtárakat egységes formában kezelhetjük, mivel az általa biztosított csatolófelületek felhasználásával alkalmazásunk a különféle szerkezetű címtárakat is azonos módon érheti el. Az ADSI hasonló szerepet tölt be a címtárak elérésében, mint az ODBC a különféle adatbáziskiszolgálók esetében. Segítségével a címtárakkal kapcsolatos gyakori feladatok – felhasználók kezelése, különféle erőforrások keresése – az elosztott, több címtármegoldást használó környezetekben is könnyen elvégezhetőek. Az ADSI objektumai valójában COM objektumok, amelyek a kezelni kívánt címtár objektumait jelenítik meg, és COM csatolófelületeken keresztül érhetőek el. Az objektumok két csoportba sorolhatók; a konténer típusú objektumok más konténereket, és levél típusú objektumokat tartalmazhatnak. Az ADSI providerek valósítják meg a konkrét címtártípus eléréséhez szükséges funkciókat. Amint az alábbi ábrán is látható, az ADSI-t használó ügyfeleknek (így a .NET Framework komponenseinek is) csak az ADSI provider által biz-



Címtár elérése ADSI provideren keresztül

tosított csatolófelület megfelelő használatával kell törődniük, a háttérben lévő címtár egyedi adottságainak megfelelő műveletek kezdeményezése már a provider feladata.
Az ADSI által biztosított legfontosabb providerek a következők:

- WinNT:// – a provider segítségével a Windows NT 4.0 tartományokhoz férhetünk hozzá, és ez teszi lehetővé az önálló (Windows NT, 2000, XP) számítógépeken levő helyi címtár-adatbázisokhoz (SAM) való hozzáférést is. Meg kell jegyeznünk, hogy a WinNT provider kompatibilitási okokból használható Active Directory esetében is, de ez számos korlátozással jár (a keresés nem támogatott, szervezeti egységek megadása nem lehetséges, stb.) ezért használata nem célszerű.
- LDAP:// – ez a provider az LDAP protokoll segítségével elérhető címtárak (például az Active Directory) kezelésére szolgál, ezt használjuk a mintaprogramban is.
- NDS:// – segítségével a Novell Directory Services szolgáltatáshoz férhetünk hozzá.

Minden ADSI objektum ügyfél oldali attribútum-gyorsítótárral rendelkezik, amelyben ideiglenesen tárolódnak az objektumhoz tartozó tulajdonságok nevei és értékei. A gyorsítótár jelentősen növeli az attribútumokkal végzett különféle műveletek teljesítményét, mivel nélküle programunknak minden egyes attribútum értékének beállítása után a címtárhoz kéne fordulnia.

Az AD hozzáférést biztosító .NET komponensek

A .NET Framework két komponenst biztosít a címtárak eléréshez, mindkettő a System.DirectoryServices névtérben található:
A DirectoryEntry komponens a címtár egy meghatározott levél, vagy tároló objektumának reprezentációja, létrehozásakor meg kell adnunk a felhasználandó ADSI providert, és a megcélzott objektum azonosítóját a provider által megkövetelt szintaktika szerint. Az alábbi kódrészlet olyan DirectoryEntry objektumot hoz létre, amely a megadott nevű számítógép rendszergazda felhasználóját reprezentálja a WinNT:// provider segítségével:

```
DirectoryEntry admin = new
↳ DirectoryEntry("WinNT://ezagep/rendszergazda");
```

Ha Active Directory tartomány felhasználóját megjelenítő objektumot szeretnénk létrehozni, a kód a következők szerint módosul:

```
DirectoryEntry admin = new
↳ DirectoryEntry("LDAP://CN=rendszergazda,
↳ CN=Users,DC=falatrax,DC=hu");
```

Hasonló módon választhatjuk ki a címtár bármelyik levél- vagy konténerobjektumát, amelyet ezután a DirectoryEntry komponens tulajdonságainak és metódusainak segítségével érhetünk el.
Már a konstruktorban megadhatjuk az objektum néhány további fontos tulajdonságát is. A második és harmadik paraméter a csatlakozáskor elküldendő felhasználónév és jelszó lehet, ha ezeket nem adjuk meg, a komponens automatikusan az aktuálisan bejelentkezett felhasználó adatait fogja használni. Negyedik paraméterként az AuthenticationTypes enum értékei közül adhatjuk meg valamelyiket. A DirectorySearcher komponens a címtárban való keresésre szolgál (csak LDAP providerrel használható). Létrehozásakor paraméterként kell adnunk egy DirectoryEntry objektumot, az így megadott konténer lesz a keresés kiindulópontja. Ha nem adjuk meg ezt a paramétert, akkor alapértelmezés szerint a keresés az aktuális tartomány gyökerében indul.

```
DirectoryEntry ADroot = new
↳ DirectoryEntry("LDAP://falatrax.hu");
DirectorySearcher s = new
↳ DirectorySearcher(ADroot);
```

A konstruktor további lehetséges paraméterei a következők (a kódrészletekben a tulajdonságokat az objektum létrehozása után állítjuk be, de megadhatók a konstruktor paramétereként is):
Filter – második paraméterként egy LDAP szintaktika szerinti szűrőfeltételt adhatunk meg (string), a keresés eredményében csak a feltételnek megfelelő objektumok fognak szerepelni (alapértelmezés szerint minden objektum átjut a szűrőn):

```
s.filter = „(objectCategory=computer)";
```

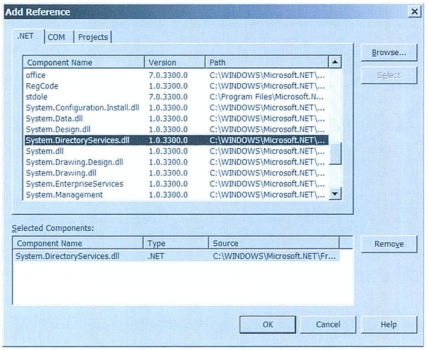
PropertiesToLoad – harmadik paraméterként a keresés eredményobjektumaiba bekerülő tulajdonságok halmazát adhatjuk meg (string tömb). Alapértelmezés szerint minden tulajdonság bekerül, ha csak meghatározott tulajdonságokat szeretnénk visszakapni, az „add" metódust kell meghívnunk:

```
s.PropertiesToLoad.Add(„Description");
```

SearchScope – utolsó paraméterként a keresés hatókörét állíthatjuk be, a SearchScope enum értékei közül valamelyikre:

- Base – a keresés a kiindulópontként megadott objektumra (konténerre) korlátozódik.
- OneLevel – csak a megadott kiindulópont alatt egy szinttel (a gyermekobjektumok között) keres, a kiindulópontban nem.
- Subtree (alapértelmezés) – a kiindulópontban kezdődő teljes részfában keres (a kiindulópontban is).

Ha Visual Studióban szeretnénk használni a komponenseket, akkor a megfelelő assembly referenciát (System.DirectoryServices.dll) hozzá kell adnunk projektünkhöz.



Assembly referencia hozzáadása

A következőkben megvizsgáljuk a komponensek segítségével végrehajtható különféle alapfeladatok megvalósítását, a mintaprogram kódrészleteinek alapján. Természetesen a kódok ebben a formában nem használhatók „éles" programban, ehhez mindenkinek el kell végeznie a megfelelő módosításokat.

Az AD tartalmának listázása

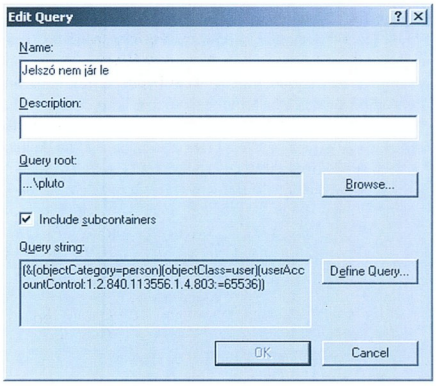
Az alábbi kódrészlet kilistázza a teljes tartományból származó objektumokat a paraméterként megadott LDAP string szerinti szűrés után. Elsőként létrehozzuk a tartomány gyökerére mutató DirectoryEntry objektumot, majd a Directory Searcher konstruktorában ezt adjuk paraméterként. A szűrő beállítása után a DirectorySearcher objektum FindAll() metódusának meghívásával SearchResult objektumokból álló tömböt kapunk vissza, ennek elemein végiglépkedve lekérjük a bennük lévő DirectoryEntry objektumokat, és kiírjuk ezek Path tulajdonságát.

```
static void ListAD(string filter){
   string domainName = GetName(„Kérem a
↳ tartomány nevét: „);
   //A gyökér, vagyis a listázás kiindulópontjának
↳ megadása
   DirectoryEntry ADroot = new
↳ DirectoryEntry(„LDAP://" + domainName);
   //A keresést végző objektum létrehozása
↳ //a paraméter a gyökér
   DirectorySearcher search = new
↳ DirectorySearcher(ADroot);
   //keresési feltétel
   search.Filter = (filter);
   foreach(SearchResult res in search.FindAll()) {
   Console.WriteLine(res.GetDirectoryEntry().
↳ Path.ToString());
   }
}
```

Már csak egyetlen dolog hiányzik a lista elkészítéséhez: meg kell határoznunk a kívánt elemeket leválogató LDAP szűrőt. A következő kódrészlet néhány egyszerű szűrőpéldát tartalmaz:

```
//felhasználó vagy csoport
„(|(objectCategory=user)(objectCategory=group))"
//letiltott felhasználók
„(&(objectCategory=person)(objectClass=user)
↳(userAccountControl:1.2.840.113556.1.4.803:=2))"
//olyan felhasználók, akiknek telefonszám mezője
↳ //nincs kitöltve
(&(objectCategory=user)(!telephoneNumber=*))
```

Bonyolultabb lekérdezések összeállításához segítségül hívhatjuk a Windows 2003 Server Active Directory Users and Computers eszközét; az elmentett lekérdezések (Saved Queries) csomópontban grafikus felületen állíthatjuk össze az LDAP szűrőnket reprezentáló sztringet.



Felhasználók, akiknek jelszava soha nem fog lejárni

Ha sok objektumot tartalmazó tartományban (vagy tartományokban) keresünk, érdemes lehet beállítani a SizeLimit és ServerTimeLimit tulajdonságokat is. A SizeLimit (int) tulajdonság megadásával maximálhatjuk a keresés során visszaadott objektumok számát. Ha a talált objektumok száma eléri a megadott értéket, a keresés megszakad, és visszaadja az addig összegyűjtött objektumokat. Erre vonatkozóan a kiszolgáló is alkalmaz korlátozást (1000 objektum), így ennél nagyobb érték megadásának nincs értelme.
A ServerTimeLimit (TimeSpan) tulajdonság segítségével egy időintervallumot adhatunk meg; a kiszolgáló maximálisan ennyi időt fog a keresés végrehajtására fordítani. Itt is van a kiszolgáló által meghatározott érték (120 másodperc), ezt semmiképpen nem léphetjük túl.

Az AD objektumok tulajdonságainak lekérése

Az objektumok tulajdonságkészletét a címtár sémája határozza meg. Az attribútumok egy része több adat tárolására is képes, és a sémától függ az is, hogy az egyes attribútumok milyen típusú adatokat tárolhatnak.
Az alábbi kódrészlet segítségével egy megadott user objektum tulajdonságait listázhatjuk ki. Csak azok a tulajdonságok jelennek meg, amelyek beállított értékkel rendelkeznek, az üres tulajdonságok nem kerülnek be a listába.
Első lépésként létre kell hoznunk az adott felhasználóhoz kapcsolódó DirectoryEntry objektumot. A kódrészlet természetesen nemcsak a felhasználók, hanem bármilyen AD objektum tulajdonságainak lekérdezésére képes, ehhez csak a DirectoryEntry konstruktorában megadott paramétert kell módosítanunk. A kódban látható domainNameToLDAP() függvény a felhasználótól bekért adatokat (tartománynév és a felhasználó neve) alakítja át a megfelelő LDAP sztringgé:

```
„LDAP://CN=rendszergazda,DC=falatrax,DC=hu"
```

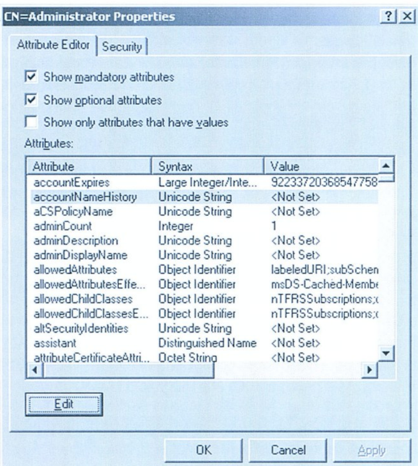
A programnak természetesen a tartomány DNS nevét (például falatrax.hu) kell megadnunk.
A kiválasztott felhasználó (vagy más objektum) tulajdonságait a DirectoryEntry komponens Properties tulajdonsága által

visszaadott PropertyCollection gyűjtemény tartalmazza. Ennek PropertyNames tulajdonsága egy újabb kollekciót ad vissza, amelynek elemein végiglépkedve kiíratjuk az AD objektum egyes tulajdonságának nevét. Mivel egyes tulajdonságok több értéket is tartalmaznak (gondoljunk csak például a felhasználó csoporttagságára), egy újabb iterációval kell kilistáznunk az egyes nevekhez tartozó tulajdonságértékeket.

```
static void ListUserProperties() {
   string domainName = GetName(„Kérem a
↳  tartomány nevét: „);
   string userName = GetName(„Kérem a felhasználó
↳  nevét (Users konténer): „);
   //a felhasználónévnek megfelelő DirectoryEntry
↳  //létrehozása
   DirectoryEntry user = new
↳  DirectoryEntry(domainNameToLDAP(domainName,
↳  „Users”, userName));
   //végiglépkedünk a kiválasztott felhasználó
↳  //tulajdonságain
   foreach(string propertyName in
↳     user.Properties.PropertyNames) {
      //egyes tulajdonságok több értéket is
↳     tartalmaznak
      int count =
↳     user.Properties[propertyName].Count;
      Console.WriteLine(propertyName + „: „);
      //végiglépkedünk az adott tulajdonság összes
↳     //értékén
      for( int i = 0; i < count; i++)
         Console.WriteLine(user.Properties
↳        [propertyName][i].ToString());
   } //foreach
}
```

AD objektumok létrehozása

Minden AD objektum számos tulajdonsággal rendelkezik, hogy pontosan melyek ezek, azt az AD séma határozza meg. Ha új objektumot hozunk létre, természetesen bármely tulajdonság értékét beállíthatjuk, de ehhez pontosan tudnunk kell a tulajdonság nevét.



Az ADSIEdit az AD séma alapján a nem létező (nem beállított) tulajdonságokat is megmutatja

A legfontosabb tulajdonságok nevei az alábbi kódrészletben is szerepelnek, de az ADSIEdit program bármelyik tulajdonság nevét (sőt a beállítandó érték típusát is) megmondja nekünk.

Az Active Directory nem tárolja az üres attribútumokat, az objektumokban csak a valóban létező (beállított) tulajdonságok nevei és értékei jelenek meg, de az ADSIEdit a séma alapján a nem létező tulajdonságok neveit is megjeleníti.

Az ADSIEdit programot a Windows 2003 CD \SUPPORT\TOOLS mappájában található mini Resource Kit részeként telepíthetjük fel.

Az alábbi kódrészletben kiválasztjuk azt a konténert, amely az új objektumot tartalmazni fogja (DirectoryEntry létrehozása), és annak Children kollekciójához hozzáadjuk a létrehozandó objektumot. Az Add() metódus paramétereként az objektum, és a létrehozásához felhasznált séma nevét (user, computer, stb.) kell megadnunk.

Ezután következhetnek az új felhasználó tulajdonságai. Windows Server 2003 esetén egyetlen tulajdonságot sem kötelező megadnunk, hogy a felhasználó objektumot létrehozhassuk; a samAccountName értéke is automatikusan generálódik bár az így létrejött objektum használhatósága erősen kérdéses.

Ha mégis beállítanánk a tulajdonságokat, ismét a Properties kollekciót használhatjuk; az egyes tulajdonságok nevéhez tartozó Value értékét kell megadnunk a bekért adatoknak megfelelően.

Képzeljük csak el, mennyit érhet egy jól megírt program, ha mondjuk egy szövegfájlban kapott lista alapján, másnap reggelig 1000 új felhasználót kell létrehoznunk, külön home mappával és a felhasználónévből képzett induló jelszóval.

```
static void AddUser() {
   string domainName = GetName(„Kérem a tartomány
↳  nevét: „);
   string userName = GetName(„Kérem az új
↳  felhasználó nevét (Users konténer): „);
   string pwd = GetName(„Jelszó: „);
   //Az itt létrehozott DirectoryEntry által
   //meghatározott tárolóban fog létrejönni az új
   //felhasználó (Users konténer)
   DirectoryEntry ADroot = new
↳  DirectoryEntry(domainNameToLDAP(domainName,
↳  „Users”, „”));
   //a kiválasztott tárolóban létrehozzuk az új
   //objektumot
   DirectoryEntry newUser=
↳  ADroot.Children.Add(„CN=”+userName,”user”);
   //és beállítjuk a tulajdonságait
   newUser.Properties[„name”].Value = userName;
   newUser.Properties[„samAccountName”].Value
↳  userName;
   newUser.Properties[„userPrincipalName”].Value =
↳  userName + „@” +domainName;
   newUser.Properties[„Description”].Value = „.NET
↳  progam által létrehozott felhasználó”;
   //engedélyezett felhasználó legyen
   newUser.Properties[„userAccountControl”].Value =
↳↳    66048;
   //a jelszó beállítása előtt létre kell hoznunk
   // az új objektumot
   newUser.CommitChanges();
   //jelszóállítás natív metódus segítségével
   newUser.Invoke(„SetPassword”, new
↳  object[]{pwd});
```

```
   Console.WriteLine(„Az új felhasználó („ +
↳ newUser.Properties[„distinguishedName"].Value +
↳ „) létrejött.");
}
```

A létrehozott objektum, és a tulajdonságok változásai nem íródnak azonnal fizikailag az AD-ba, a megadott értékek alapértelmezés szerint előbb a helyi gyorsítótárba kerülnek. A gyorsítótár használatát a DirectoryEntry objektum UsePropertyCache tulajdonságának beállításával szabályozhatjuk (alapértelmezés szerint van gyorsítótárazás). Miután mindent beállítottunk, a gyorsítótár tartalma a CommitChanges() metódus meghívásával kerül be az AD-ba.
A jelszó tulajdonság különleges elbánásban részesül, értékét nem lehet csak úgy megváltoztatni, és persze egyáltalán nem lehet kiolvasni. Hogy a jelszót beállíthassuk, az AD natív SetPassword metódusát kell meghívnunk, a DirectoryEntry Invoke() metódusának segítségével.
A metódus csak akkor hívható, ha az új objektum már fizikailag is létezik, ezért a jelszó beállítása előtt meg kell hívnunk a CommitChanges() metódust.

AD objektumok törlése

Az objektumok törlését úgy végezhetjük el, hogy létrehozunk egy DirectoryEntry-t, amely a megfelelő konténerre mutat, és meghívjuk a hozzá tartozó Children kollekció Remove() metódusát. A metódus paraméterként a törlendő elemre mutató DirectoryEntry objektumot vár.
Az alábbi kódrészletben bekérjük a tartomány és a törlendő számítógép nevét, létrehozzuk a „Computers" konténernek, és a törlendő számítógépfióknak megfelelő DirectoryEntry objektumokat, majd a Remove() metódus meghívásával töröljük a fiókot.

```
static void DeleteComputer() {
   Console.WriteLine(„Számítógép törlése:\n");
   string domainName = GetName(„Kérem a tartomány
↳ nevét: „);
   string computerName = GetName(„Kérem a törlendő
↳ számítógépfiók nevét (Computers konténer): „);

   DirectoryEntry ADroot = new
↳ DirectoryEntry(domainNameToLDAP(domainName,
↳ „Computers", „"));

   DirectoryEntry computerToRemove = new
↳ DirectoryEntry(domainNameToLDAP(domainName,
↳ „Computers", computerName));

   ADroot.Children.Remove(computerToRemove);
   Console.WriteLine(„A ‘„ + computerName + „‘
↳ számítógépfiók törölve.");
}
```

Felhasználó autentikációja az AD adatai alapján

Természetesen egyszerűen megoldható az is, hogy alkalmazásunk felhasználóit az AD autentikálja. Az alábbi kódrészletben bekérjük a tartomány nevét és a felhasználói fiók adatait, majd ezek felhasználásával létrehozunk egy DirectoryEntry objektumot.
A natív ADSI objektumhoz való csatlakozással kikényszerítjük a hitelesítést. Ha ebben a sorban nem keletkezik hiba, a megadott adatok jók voltak, a felhasználó jogosult a belépésre (az esetleges hibát egy szinttel feljebb kapjuk el).
Ha a már megismert módon lekérdezzük a felhasználó csoporttagságát is, ennek megfelelően engedélyezhetjük, vagy tilthatjuk a programunk egyes funkcióihoz való hozzáférést.

```
static void AuthenticateUser(){
   string domainName = GetName(„Kérem a tartomány
↳ nevét: „);
   string userName = GetName(„Kérem a
↳ felhasználónevet: „);
   string pwd = GetName(„Jelszó: „);
   string domainAndUsername = domainName + @"\" +
↳ userName;
   DirectoryEntry user = new
↳ DirectoryEntry(„LDAP://"+domainName,
↳ domainAndUsername, pwd);

   Object obj = user.NativeObject;

   Console.WriteLine(„A felhasználó azonosítása
↳ sikeres!");
}
```

Szerényi László
szerenyi.l@met.hu

A cikkben szereplő URL-ek:
http://store.netacademia.net/mshu/OTHER/technet_code/AD.zip

Kalandozás az Internet Explorer körül

Miért jó, illetve miért lehetne még jobb?

A kérdés talán költőinek hat, bár nem annak szántam. Úgy érzem, érdemes foglalkozni a Microsoft böngészőjével, hiszen egy sor olyan előnnyel rendelkezik, amelyeket nem szívesen áldozunk fel, csak azért, mert van pár olyan hiányossága, ami azért kezelhető.

Mindenekelőtt szeretném leszögezni, hogy cikkemnek nem célja a browser-háborút erősíteni. A személyes tapasztalatok alapján meghozott választás joga minden felhasználót megillet, abba nem szabad beleavatkozni! Írásomban én az Explorer azon előnyeit szeretném megmutatni, amelyek miatt úgy gondolom, hogy továbbra is ez a böngésző a legjobb egy adott tartományi környezetben.

Tények

Való igaz, hogy kezelőfelületüket tekintve a konkurens böngészők fejlesztése kicsit elszaladt az IE mellett, talán a biztonságtechnikai fejlesztések úgyszintén. De szerintem még mindig egyértelműen az Internet Explorer a leghatékonyabban használható, integrálható, szabályozható böngésző egy Windows szerverekből felépülő tartományban. Sokak örömére már hallható volt némi információ a következő generációs Internet Explorer 7-es fejlesztéséről és mihamarabbi kiadásáról. Mondhatjuk, hogy tán kicsit többet késett ez a bejelentés, mint kellett volna, de előfordult már ilyen a Microsoft történetében. Ám a hasonló esetek végén legtöbbször egy nagyon korrekt terméket vehettünk birtokba, ami bőven kárpótolt az elvesztett időért. Az időközben esetlegesen meggyengült piaci részesedés pozíciójának visszaszerzése miatt pedig aggódjanak a marketingszakemberek, mi, rendszergazdák, amúgy sem unatkozunk.

Az Internet Explorer tartományi környezetben vitathatatlan előnyökkel bír.

Miért szeretjük az IE-t jobban?

Erre a kérdésre egy kis listával válaszolnék, aminek a pontjait a továbbiakban részletesebben is kifejtem.

- Beépített NTLM és Kerberos támogatás
- GPO-val szabályozható, amihez jókora gyári GPO készlet áll rendelkezésre
- WindowsUpdate site elérése, használata.
- Beágyazott VBS scriptek futtatása

A többi kiegészítő Explorer képességre most nem térnék ki, mert ezek nem is ugyanabban a súlycsoportban vannak, mint a lista elemei, illetve más böngészők is kisebb-nagyobb eltérésekkel ugyan, de eléggé hasonló kiegészítő funkciókat hoznak egymáshoz képest. A felsorolást tekintve pedig nekem már ennyi is elég volt ahhoz, hogy úgy döntsek, nálunk ez marad a hivatalos böngésző a hálózatban.

NTLM

Első pontként a „Windows NT LAN Manager" hálózati autentikációs protokoll és a Kerberos alapú felhasználó-azonosítás beépített támogatása szerepel. Ennek tartományi környezetben érezhető előnyeiről nem kell sokat mesélni. Tegyünk fel egy egészen egyszerű esetet, amikor is az intranet információit egy belső SharePoint Portal Server biztosítja. Egyetlen központi adatbázisban, az Active Directory-ban tárolható az összes jogosultság, hozzáférés-szabályozás, így nem kell a felhasználónak minduntalan jelszóablakokkal megküzdenie, amikor valamihez hozzá szeretne férni az intraneten. Jó esetben egyetlen egyszer adja meg az azonosítóját és jelszavát, amikor belép a tartományba. Onnantól, ha bármilyen webes elérésre van szüksége a tartományon belüli IIS szervereken szolgáltatott adatokhoz, nem szükséges újból azonosítást végeznie, mert az Explorer képes kezelni átadni a szükséges információkat (ideértve a helyi tanúsítványokat is). Ez a mindennapos munkát nagyon megkönnyíti, és központilag is egyszerűbb szabályozni: Mindent egy helyen!

GPO-val szabályozható

Vigyük tovább az előző példát, és mondjuk azt, hogy szeretnénk az Explorer beállításait központilag szabályozni. Azaz legyen automatikus a Proxy beállítás, és az Intranet site-hoz is kapcsolódjanak a rendszergazda által megálmodott biztonsági szintek. Ezenkívül szeretnénk azt is, hogy minden felhasználónak megjelenjen a Favorites/Kedvencek folderben egy új könyvár, a cégen belül használatos linkekkel (mondjuk éppen az intranetet szolgáltató SharePoint szerver egyik portáljának útvonalával). Ezen kívül még azt is szeretnénk, hogy a felhasználók ne tudják elállítani a központilag meghatározott jellemzőket. A feladatok megoldására számos GPO áll a rendelkezésünkre, és még egy nagy halom megoldás a további szabályozási, testreszabási igények kielégítésére. Minden rendszergazda megtalálja a neki tetszőt. Itt azért fontos megemlíteni, hogy szükség lehet teljesen egyedi elképzelések alapján kitalált beállításokra is. Példaként említeném a Temporary Internet Files folder méretének központi korlátozását. Erre egyelőre nincs központi házirend, de remélem, hogy a következő frissítések alkalmával majd ez is belekerül az Explorert vezérlő GPO-k népes táborába.

Ha valaki elmélyül az említett Policy objektumokban, láthatja, hogy szinte mindent lehet szabályozni, ami fontos lehet. Egy nagyvállalati környezetben pedig éppen ez a lényeg: a központi szabályozás. Ettől akár használhatnak a kliensek bármilyen más böngészőt is, egyedi beállításokkal. A lényeg, hogy mindig van egy, ami pont olyan, amilyet a fejlesztőkkel közösen megálmodtak a rendszergazdák. Ha valami hiba adódik és az egyedileg használt böngésző nem birkózik meg a feladattal, máris érkezik a segítség: „indítsd el az Explorert, és a kedvencek között megtalálod a linket, ami biztosan működni is fog". Egy olyan helyen, ahol a fejlesztőcsapat egy sor céges alkalmazást készít, ami webes felületen érhető csak el, elengedhetetlen, hogy legyen a cég hálózatára nézve globális böngészőbeállítási lehetőség. Mégiscsak jobb, mintha a rendszergazdának egyenként kell ellátogatnia 300 géphez. bár így folyamatos személyes kontaktus lenne a kedves felhasználókkal, csak kétlem, hogy erre egy rendszergazda ráérne.

Elengedhetetlen, hogy legyen a cég teljes hálózatára szóló böngésző-beállítási lehetőség.

WindowsUpdate site elérése, használata

Ez a pont végül is csak azért került bele a listába, mert az elengedhetetlenül fontos frissítések manuális elérése csakis az Internet Explorer-en keresztül valósulhat meg. „Természetesen" van SUS szerver a tartományban, ami feleslegessé teszi a manuális frissítést, de mégis, ez olyan fontos képesség, ami miatt muszáj említést tenni róla. Egy új meghajtóprogramra van esetleg szükség, vagy csak szeretnénk megtudni, hogy van-e újabb kiadású driver a frissen vásárolt videokártyánkhoz? Ezt a SUS jelenlegi verziója nem tudja orvosolni, de mi magunk, az Internet Explorerrel és a Windows Update webhellyel igen.

Beágyazott VBS scriptek futtatása

Mivel a VBS nem egy nyílt szabványú környezet, ezért a HTML-be ágyazott Visual Basic Script-ek támogatása teljesen hiányzik az összes többi böngészőből. Persze nem ez a helyzet a JAVA-val, ami pont az ellenkezőjéről híres, hiszen már szinte az otthoni kenyérpirítóhoz is lehet JAVA plugint letölteni. Ám igencsak sarkalatos pont lehet a VBS kérdés, amikor egy már meglevő, a cégen belül kifejlesztett alkalmazást használunk webes felületen. Ilyen esetben nem kérdés, hogyha „ehető tojásra" van szükségünk, nem a tojásokat kezdjük el patch-ekkel, update-ekkel bombázni, hanem olyan „tyúkot" szerzünk, ami megfelelő tojást rak. Persze előfordulhat, hogy a kis fiatal csirke még nem felel meg minden aktuális biztonsági követelménynek. Van megoldás: zárjuk ketrecbe! A szakma jobbjai nem győzik hangsúlyozni, hogy nem akkor ér véget a telepítés, amikor a „kék csík a végére ér", hanem éppen akkor kezdődik! Egy kis odafigyeléssel nagyon sok probléma orvosolható!

Biztonság

Amikor a biztonság szót leírjuk a számítástechnikával kapcsolatban, egy igen bonyolult, buktatókkal bőven ellátott területre érkezünk. Rengeteg megközelítése van az informatikában használatos biztonsági megoldásoknak, ahol egy-egy apróságon szinte az egész rendszer áll vagy bukik. Az elvárások pedig, hogy mondjuk a böngésző telepítése után („kék csík 100%-nál") azonnal minden tökéletesen működjön,

és már rögtön teljesen biztonságos és behangolt is legyen, nos, ezek a gyakorlatban igencsak ütik egymást. Nézzük meg, hogy milyen is a biztonságos internetböngészés. Ugye ismerős az a felhasználói igény, hogy a számtalan weboldal, ezerféle megjelenítési technikája, mind-mind tökéletesen élvezhető legyen, de azért ne jusson be egyetlen kémprogram (spyware) se, és a vírusfertőzés is kizárt legyen? Ezt a feladatot nem bízhatjuk csak a böngészőre! Illetve ha mégis bekövetkezik a „fertőzés", nem tehetjük kizárólag csak a böngészőt felelőssé érte! Manapság elmondható, hogy akkor áll egy operációs rendszer legalább „három lábon", ha van hozzá víruskereső, kémprogram irtó és egy jó tűzfal. Plusz megoldott az aktuális biztonsági frissítések folyamatos letöltése. A két utóbbit (tűzfal és autoupdate) az XP SP2 automatikusan segít beállítani és a víruskereső szükségességére is figyelmeztet.

A harmadik „lábat" nekünk kell alátenni, mondjuk azzal, hogy letöltjük és telepítjük a Microsoft AntiSpyWare alkalmazást. Ilyenformán karbantartott munkaállomással (és ésszerű böngészéssel) fenntartható egy olyan biztonsági szint, amely megvédi a rendszert az ártalmaktól. Miért említem az ésszerű böngészést? Nos, senki se csodálkozzon, ha „kémprogramtalálkozó" szerveződik a gépén, mert nem tud elszakadni kedvenc „warez" oldalaitól! A használt programokat is érdemes inkább biztos helyről beszerezni, letölteni! Miért gondolná bárki is, hogy az internet feketepiaca bármiben különbözik a valós élet feketepiacaitól? Itt is ugyanolyan rizikós bármit venni/elfogadni, mint ott, és persze garanciát sem várhatunk semmire. Mindenképp szerencsétlen szituációnak tartom, hogy a túlzott sokrétűséget és a „mindent-egyszerre-támogassunk" filozófiáját követve az Internet Explorer jelenlegi verziói egy sor biztonsági hiba előtt nyitnak sajnos ka-

put. Ám ha minden beállítás túl szigorú lenne, akkor meg az egyszerű felhasználó nem tudna boldogulni. A legjobb megoldás itt is, mint sok más esetben, az arany középút.

A fejlesztők szemszögéből

Ez egy hálátlan témakör. Mivel itt is rengeteg megközelítési irány létezik, nagyon nehéz megtalálni azt a közös nevezőt, ami mindenkinek megfelel. A szabványok „nem egyforma" átvétele illetve megvalósítása azt eredményezte, hogy egy-egy korszerű, gondosan megtervezett weboldal legalább háromféleképpen van megírva, hogy minden böngészőtípus igényeit ki tudja elégíteni. Vagyis, ha a kedves megrendelőnek – jogosan – az a kívánsága, hogy mindenféle böngészőben ugyanúgy nézzen ki egy bonyolultabban megálmodott website, bizony a programozóknak el kell készíteniük a többféle változatot. Ez jelentős többletmunkát és nem utolsó sorban többletköltséget jelent! Talán ha a gyártók közelednének egymáshoz, illetve a kidolgozott szabványokhoz, ez a probléma is megoldódna. Ennek okán a beharangozott 7-es Explorer verziónak fejlesztői szemmel is nagy elvárásoknak kell eleget tennie. A sokat vitatott CSS (Cascade Style Sheets) böngészőprogramonként éppen csak egy kicsit eltérő értelmezéséből adódóan, igencsak jelentős mennyiségű anyag gyűlt már össze a fejlesztői témájú fórumokon. Egy problémásabb esetben ezek átolvasása, értelmezése szintén plusz idő, ami alatt például folytatódhatna a tényleges fejlesztés.

Összegzés

Nem célom az eltérő vélemények egyikét sem erősíteni, sem gyengíteni. Pusztán csak azt érdemes tényként elfogadni, hogy az a gyártó, aki elmondhatja magáról, hogy a világon több millió PC-n fut az operációs rendszere – beépített böngészővel – annak igenis van joga a szabványokat alakítani, formálni. Más kérdés, hogy egyes irányok vezethetnek tévútra is, amelyek gyors, rugalmas felismerésével megelőzhető egy-egy probléma. Talán ezt hivatott majd az új Explorer 7-es megoldani.

A jelenlegi Explorer 6-os is tökéletesen alkalmas arra, amire való, mindenféle megjelenítési metódussal képes megbirkózni, ismeri a vonatkozó szabványokat, és nagyvállalati környezetben az egyetlen olyan jó megoldás, amely szervesen illeszkedik a már meglévő, központilag szabályozott struktúrába.

Füzesi Szabolcs
fuzesisz@osi.hu
MCSA

Microsoft újdonságok a CeBIT-en

A „Your Potential. Our Passion." azaz „Neked lehetőség. Nekünk kihívás." jelmondat jegyében mutatta be újdonságait az idei CeBIT-en a Microsoft. A mottó a vállalat küldetésének rövid és tömör összefoglalása. Azt fejezi ki, hogy a Microsoft olyan új eszközöket igyekszik fejleszteni, melyek révén segítheti az embereket lehetőségeik kihasználásában, céljaik elérésében, a magukban rejlő képességek kiaknázásában.

A CeBIT-en jelentette be a Microsoft, hogy a Windows XP Media Center Edition alapú PC-k idén további 20 ország felhasználói számára válnak elérhetővé. Ennek révén 2005 végére a Windows XP Media Center Edition már több mint 30 országban és 17 nyelven lesz kapható. Hannoverben mutatták be az online együttműködést elősegítő új megoldást, a Microsoft Office Live Collaboration Platform-ot, amely a vállalaton belüli valamint a különböző vállalatok közötti valós idejű együttműködést támogató szervereket és kliens oldali termékeket foglalja magában. Bejelentettek új, valós idejű kommunikációt lehetővé tévő eszközöket: a Microsoft Office Live Meeting és a Microsoft Office Live Communications Server 2005 termékeket. Találkozhattak az érdeklődők az Istanbul kódnéven ismert Universal Instant Messaging Client, azonnali üzenetkezelést lehetővé tévő ügyfélprogram béta verziójával is.

ASP.NET 2.0 (Whidbey)

Mi várható a 2005. évi ASP.NET-ben?

III. RÉSZ

Életszagú teljesítményfokozás az ASP.NET Cache szolgáltatásaival

Bevezetés

Új hardverre lesz szükség, a mostani már nem bírja a terhelést! Nagyon gyakran hallom ezt az (ál)érvet fejlesztőktől, miközben csupán napi pár ezer felhasználót szolgál ki a webalkalmazásuk. Sokszor óriási tartalékok lapulnak még a szerencsétlen ócskavasnak nyilvánított hardverben, csak éppen ki kell használni őket.

Nemrégiben a NetAcademia webkiszolgálója is teljesítménygondokkal küszködött, de az ASP.NET Cache és az If-Modified-Since http fejléc felhasználásával sikerült újra 10 százalék alá vinni a processzor terhelését. (Egyébként egy 300 MHz-es, P3-as „monstrumról" van szó, ami egy leselejtezett '99-es tanfolyami hallgatói gép volt. Tényleg.)

A teljesítményproblémák okai

A leggyakoribb ok általában az adatbázis-lekérdezések lassúsága illetve a külső eszközökkel kapcsolatos műveletek, mint a fájlműveletek vagy valamilyen hálózati kommunikáció. Ez utóbbira egyre gyakoribb példát szolgáltatnak a Webszolgáltatás hívások.

Teljesítményproblémák esetén két-három nap alatt csodákat lehet művelni némi odafigyeléssel...

Sok problémára látszólag kézenfekvő megoldás lehet aszinkron műveletek használata, azonban webalkalmazások esetén ez nem mindig könnyen járható út. Ha például egy Webszolgáltatást kell meghívni, akkor hiába bízzuk rá a hívást egy másik szálra, a fő, weblapot futtató szálat általában úgyis blokkolni kell, amíg meg nem érkezik a válasz a másik szálra. Ezzel viszont blokkoljuk az ASP.NET egyik munkaszálát és az aszinkron híváshoz még egy plusz szálat is a ThreadPoolból. Mindkettőben csak pár tucatnyi szál lehet, így egy intenzíven terhelt rendszerben totális csődöt mondhat a profin kidolgozott aszinkron megoldásunk.

Az aszinkron hívásokat meg lehet oldani anélkül, hogy számtalan szálat pazarolnánk a hívás végére várakozva, ehhez még támogatást is kapunk az ASP.NET 2.0-ban, az Async Page direktíva képében. Egy későbbi számban még foglalkozom ezzel, a téma iránt érdeklődők az Asynchron HTTP Handler kulcsszavakra keresve találhatnak cikkeket az interneten (főleg 1.1-re). Maradjunk a szinkron megoldásoknál. Az adatbázislekérdezések teljesítményét drámai módon lehet fokozni megfelelő indexekkel, a lekérdezések átírásával, egyszerűsítésével, a probléma újragondolásával (pl. nem biztos, hogy minden látogató számára tízezer sort kell leválogatni, úgyse érdekli őket). Némi indexelt view, egy-két tábla denormalizálása, sőt sokszor pont a normalizálás is sokat segíthet. Pl. ha sokszor szerepel lekérdezésekben a SELECT DISTINCT * FROM blah, akkor nem kellene esetleg felvenni a hiányzó blah táblát az adatbázisba és így normalizálni az adatbázismodellt? Vagy biztos kell a SELECT *, nem lenne elég valójában csak két oszlop a kombóbox feltöltéséhez? Tényleg jó, ha lejön még feleslegesen 50 oszlop, alkalmanként 500 kbyte adatforgalmat okozva?

Nem olyan nagy dolgok ezek, két-három nap alatt csodákat lehet művelni célirányos odafigyeléssel. Gyakori az, hogy a munka hevében nem rakunk fel olyan indexeket, amelyek hiányát utólag nagyon könnyű felismerni, tehát érdemes pár órát rászánni a vizsgálódásra. A módszer nagyon egyszerű.

Minden lapot futtassunk le realisztikus paraméterekkel és nézzük meg SQL Server Profilerben a lefuttatott SQL parancsokat illetve tárolt eljárásokat. Amelyik végrehajtási ideje (Duration oszlop) nagyobb pár száz millisecundumnál, azt érdemes megvizsgálni. Ha sok sort visszaadó lekérdezésről van szó, akkor természetes a nagyobb végrehajtási idő, de pár tíz sornál általában nagyon kevés, akár 10 ms alatt (gyakran 0) kell látni.

Ha egy lekérdezés által olvasott (8 kbyte-os) lapok száma, amelyet a Reads oszlopban láthatunk, nagyobb pár ezernél, szintén gyanúra adhat okot. Egy tízezer lapolvasást kiváltó lekérdezés lehet, hogy néhányszor 10 millisecundum alatt fut le, de biztosak lehetünk benne, hogy egy erősen terhelt szerveren az ilyen lekérdezések fogják lefullasztani a szervert, és a végrehajtási idejük is fel fog szökni akár több tíz másodpercre is, gyakran időtúllépést okozva a hívókon.

Konkrét példánkban, a tudástárban **[1]** jó pár lekérdezés volt, amelyet néhány index-szel alaposan fel lehetett gyorsítani. Élő, sokak által használt rendszert hangoltam, ilyenkor jól lehet azonosítani azokat a sokak által használt lekérdezéseket, amelyekkel látványos eredményeket lehet elérni.

Profilerrel egy nap termését SQL táblába mentve néhány SQL paranccsal könnyedén kielemezhetők a lassú illetve gyakran futtatott lekérdezések. A gyakran használtakkal mind a szerver erőforrások csökkentése miatt, mind a felhasználók által érzékelt késleltetések lefaragása céljából mindenképpen érdemes foglalkozni. Összeszoroztattam a napi végrehajtások számát (COUNT(*)) és a végrehajtási időt (Duration), és a legnagyobb összköltségű öt lekérdezést optimalizáltam. Voltak olyanok, amelyeket nehéz volt együtt optimalizálni, mert mindkettőt csak clustered index-szel tudtam volna felgyorsítani, amiből egy táblán csak egy lehet.

Ezért örültem, hogy már SQL Server 2005 volt a háttéradatbázis, mert abban egy nonclustered indexet ún. included oszlopokkal felvértezve két clustered indexhez hasonló teljesítményt tudtam elérni. Igaz, ezt a módosítások kárára tettem, de esetünkben ez nem volt gond, mert napi pár 10 módosítás van csak a rendszerben.

Sajnos maradtak olyan lekérdezések, amelyek költsége jelentős volt, de nem lehetett optimalizálni: a BLOB-okat lekérdező SELECT-ek.

A BLOB a Binary Large Object rövidítése, az image, text és ntext típusú oszlopokban tárolt maximum gigabyte méretű adatokat értik alatta. Mi a tudástárba bepostázott képeket és egyéb feltöltött fájlokat tároljuk SQL Serverben. Hogy ez a megfelelő tárolási mód vagy a fájlrendszer + adatbázis, az vita kérdése, esetünkben az egyszerűbb backup és a garantált adatépség érdekében döntöttünk az adatbázis mellett.

A BLOB-okat egy asp.net lap tölti be adatbázisból és szolgálja ki a megjelenítő lapok részére. Ehelyett egy saját HTTP Handler lenne a jobb megoldás, mivel az aspx lapoknak van egy jelentős saját költsége, ami szükséges, ha html tartalmat generálunk ASP.NET vezérlők segítségével, de semmi haszna direkt tartalomgeneráláskor. Ez a következő tudástár verzióban változni fog.

Első körben ez a lap (download.aspx) minden kérés esetén benyúlt az adatbázisba, lehozta a kívánt tételt, és kipakolta azt a kimenetére. Ez jelentős terhelést okozott az SQL Serveren, ráadásul olyan terhelést, amelyet indexeléssel nem lehet javítani. Ha egyszer egy kép 200 kbyte, akkor akárhogy is trükközik a szerver, csak fel kell olvasni és vissza kell küldeni az ügyfélalkalmazásnak.

Jobban belegondolva azonban teljesen feleslegesen dolgozott az SQL Server. A letárolt BLOB-ok praktikusan soha nem változnak meg (évi néhány törlés maximum, módosítás soha). Ha már egyszer lekérdeztünk, jó lenne letárolni őket valamivel közelebb az ügyfélhez (aspx lap), mint az adatbázis. Itt az ideje bevetni az ASP.NET Cache-t.

Az ASP.NET Cache

A Cache első ránézésre nagyon hasonló a korábbról ismert Application objektumhoz. Globális egy webalkalmazásra nézve, azaz minden lap ugyanazt a Cache objektumot látja. A Cache azonban szándékosan feledékeny: a belerakott tartalmat időnként elfelejti. Alapvetően akkor kezd el takarítani, ha memóriakényszere van. Hogy ez hány százaléknál következik be, arra nincs ráhatásunk, sőt ismeretünk se, bíznunk kell benne, hogy nem túl agresszíven allokálja a memóriát kiéheztetve más processzeket (eddig ennek nincsenek jelei). Emellett be lehet állítani, hogy egy bizonyos időpontban essen ki egy tétel belőle, például a következő órában, ha tudjuk, hogy a forrásadatokat mindig óránként frissítik. Ez az Absolute Expiration.

Használható a Sliding Expiration is, amelyben egy tétel x idő múlva kiesik a tárból, de az időzítő újraindul, ha időközben mégis felhasználták, kiolvasták a tartalmát.

Ami azonban sokkal izgalmasabb, az a Dependency-k használata. Gyakori, hogy a cache-ben tárolt adat fájltartalom, XSLT, XML, valamilyen szövegfájl, egy konfigállomány, stb. A cache-ben ezek jól érzik magukat, hisz sokkal gyorsabb egy referenciát visszakapni a beolvasott tartalomra a Cache-ből, mint az operációs rendszeren áthaladva felolvasni őket. Igen ám, de ezeket a fájlokat menet közben megváltoztathatják, ilyenkor azonnal ki kellene dobni a tartalmukat a Cache-ből. Erre valók a *CacheDependency-k*.

Az ASP.NET 1.x-ben fájlokra, könyvtárakra és másik Cache tételre lehet építeni függőségeket. Az első kettő esetén a FileSystemWatcher osztály segítségével a Cache monitorozza a fájlrendszert, és ha a megadott fájl vagy könyvtár változik, akkor kidobja a tételt a memóriából. Utóbbi egymástól függő tartalom illetve az Output Cache szabályozására használható, erről még lesz szó a cikk következő részében.

Nézzünk egy fájlrendszer alapú példát:

```
DataSet cachedData = (DataSet)Cache["d"];
if (cachedData == null)
{
  Cache["d"] = dg.DataSource = LoadFromFile();
}
else
{
  dg.DataSource = cachedData;
}

string dataSourcePath =
 HttpContext.Current.Server.MapPath("~/emps.xml");

private DataSet LoadFromFile()
{
  DataSet ds = new DataSet();
  ds.ReadXml(dataSourcePath);
  return ds;
}
```

Nagyon fontos, hogy NEM így kell csinálni!

```
//Rossz megoldás
if (Cache["d"] == null)
{
  Cache["d"] = dg.DataSource = LoadFromFile();
}
else
{
  //!!! Mire ideérünk már lehet, hogy kiesett
  //a Cacheből az adatunk !!!
  dg.DataSource = Cache["d"];  //BUGBUG
}
```

Ha azt akarjuk, hogy az emps.xml változásakor azonnal kiessen a tartalom a Cache-ből, akkor be kell vetnünk a CacheDependency-t:

```
Cache.Insert("d", cachedData,
  new CacheDependency(dataSourcePath));
```

Mekkora adatokat lehet a Cache-be rakni? Sokáig sulykolták a fejünkbe, hogy az Application és a Session objektumokat ne használjuk nagy fájlok, RecordSetek, DataSetek tárolására, mert hamar elfogyasztjuk a webszerver memóriáját.

A Cache egy kicsit más állatfajta. Ezt elég gátlástalan módon lehet használni, hisz elvileg úgyis kiveti magából a felesleget. Ennek ellenére azért észnél kell lenni ezzel is, ha túl gyorsan tárolunk el benne nagy objektumokat, akkor előfordulhat, hogy nem tudja elég gyorsan kidobálni őket, és elfogy az ASP.NET memóriája. Ez csak extrém körülmények között állhat elő, és csak azért mondom el, hogy később ne úgy jöjjön vissza, hogy „Soci azt mondta", bátran olvassuk fel másodpercenként a driver.cab-ot (50 mega) és rakjuk be a Cache-be. ☺

Tény viszont, hogy pár megabyte-os DataSetekkel nem kell szívbajosnak lennünk, bátran pakoljuk bele őket a Cache-be, ha egyébként is a felhasználók számára osztott adatokról van szó. Minden egyes felhasználó (Session) számára ennyi infót letárolni viszont már valószínűleg sok lenne a Cache számára is, ezt Önnek, kedves Olvasó, nem ajánlom.

Tudástár példánkban adatbázisban tárolt fájlokat szeretnénk Cache-elni. Maximum pár száz fájlról van szó, melyek mérete legfeljebb 256 kbyte. Így 300 darabbal számolva legfeljebb 77 Mbyte-nyi adatot akarunk a Cache-be rakni, ez nem sok, ha a szerverben van elég memória, amely esetünkben 512 Mbyte. Az adatbázisban tárolt fájlok megjelenítéséhez fontos azok néhány egyéb jellemzője is, ezért a Cache-be nem csak a fájlokat reprezentáló byte tömböket, hanem egy adatleíró struktúrát helyezünk el:

```
private class ContentDescriptor {
  public byte[] Content;
  public string ContentType;
  public string ContentName;
}
```

A Cache-kezelés logikája azonos az XML-es példánál látott módszerrel: megnézzük benne van-e a tartalom, ha nincs, belerakjuk, ha benne van, kiolvassuk.

```
//Adatbázis-tartalom Cache-elése
//id: a tartalom azonosítója, int típusú
ContentDescriptor contentDesc =
  (ContentDescriptor)Cache["D" + id.ToString()];
if (contentDesc == null) {
  //nincs benne a cache-ben
  contentDesc = LoadFileFromDatabase();
  Cache["D" + id.ToString()] = contentDesc;
}
Response.ContentType = contentDesc.ContentType;
Response.BinaryWrite(contentDesc.Content);
```

Az éles kód ennél valamivel hosszabb, mivel fel kellett készíteni hibás helyzetekre is, pl. amikor a megadott id-vel nincs is az adatbázisban a letöltendő adat. Hogy ekkor mi történik, érdemes megnézni az éles verzióban. ☺

Működő, de az eredetihez képest egyszerűsített és a pubs adatbázisra átírt példa a **[2]** címen található meg.

A Cache-módszerrel sikerült jelentősen csökkenteni az adatbázis terhelését. A memória-felhasználást megfigyelve nagyon érdekes volt látni, ahogy az ASP.NET Cache és az SQL Server Buffer Manager (Cache kezelő) egymással versenyzik a memóriáért. A szerverben az 512 megabyte memóriáért egy SQL Server 2000, egy SQL Server 2005, egy Exchange 2000, IIS, Apache és számos egyéb folyamat küzd. Mind az SQL Serverek, mind az ASP.NET Cache igyekszik élni a szabad memóriával. A teljesítményszámlálókat megfigyelve látszott, hogy az ASP.NET Cache mérete pár perc használat után felment pár 10 tételre, miközben az SQL Server összehúzta magát, azaz elengedte az általa Cache céljára használt memória egy részét. Ezek után egy nagyobb lekérdezésre az SQL Server harapott egy nagyobbat a RAM-ból, mire az ASP.NET Cache elengedte tételeinek egy részét, hadd örüljön az SQL Server (meg persze más egyéb folyamatok). Azóta is szépen játszanak egymással. Persze egy nagyobb szerver esetén az SQL Server maximális memóriafelhasználását lejjebb venném, így nem kellene versengeniük egymással, de egy ekkora vason és ilyen sok processz esetén nincs értelme ennek.

Amikor a Cache nem elég

Hiába a Cache mágia, bár a kiszolgáló terhelése jelentősen csökkent, böngészőből nézve mégse lettek gyorsabbak a lapok. Képzeljük el azt, hogy egy oldalon van öt kép, mindegyik az előbbi dinamikus letöltéssel egy aspx lap kimenetéből csurog ki (downloadtest.aspx). Mivel alapesetben az aspx lapok kimenete logikailag azonnal lejár, hisz dinamikus tartalomról van szó, ezért amikor ránavigál valaki az oldalra, minden kép újra letöltődik, a böngésző nem meri letárolni a tartalomgeneráló aspx lap által küldött tartalmat. Ilyenkor ugyanis az ASP.NET a következő fejléceket küldi a böngészőnek (megfigyelésük: **[3]**):

```
Kérés:
GET /WebSite1/download.aspx?id=0736 HTTP/1.1
Válasz:
HTTP/1.1 200 OK
Date: Tue, 22 Feb 2005 01:03:40 GMT
Server: Microsoft-IIS/6.0
X-AspNet-Version: 2.0.50203
Cache-Control: private
Content-Type: image/bmp
Content-Length: 643
```

A Cache-Control: private jelentése: útközben a Proxy-k nem tárolhatják a tartalmat, de a böngésző igen. Ennyi alapján azonban még nem tárolja el a letöltött képet a böngésző, mert nem tudja meddig érvényes az, így nem meri megkockáztatni, hogy régi tartalmat jelenít meg. Látszólag ezen könnyű segíteni, csak le kell küldeni HTTP fejlécekben, hogy a tartam tárolható. Az ASP.NET OutputCache direktíva automatikusan képes erre:

```
<%@ OutputCache Duration="864000" VaryByParam=
↳ "id" %>
```

A fenti sort egy aspx lap elejére elhelyezve azt közöljük az ASP.NET-tel, hogy 864000 másodpercre, azaz 10 napra eltárolhatja a lap legenerált kimenetét az OuputCache-ben, amely az eddig tárgyalt Cache egy részhalmaza. Első ránézésre nem triviális módon ez nem csak azt eredményezi, hogy az ASP.NET letárolja a generált tartalmat a szerver memóriájában, hanem ezzel együtt olyan fejlécet is küld az ügyfeleknek (akik általában böngészők), hogy barátom, 10 napig nyugodtan tárold a neked küldött adatokat.

A direktíván használható Location attribútum szabályozza a lehetséges tárolási helyeket:

```
Location="Any | Client | Downstream | Server |
ServerAndClient | None"
```

Alapesetben az Any beállítás működik, mindenki Cache-elhet. Így a szerver Cache-el, ezzel az egymás után érkező böngészőknek kis költséggel képes kiszolgálni a tartalmat, ha pedig egy böngészőnek újra szüksége van a tartalomra, de az már le van töltve a gyorsítótárába (pl. Temporary Internet Files az IE-ben), akkor nem kéri le újra a szervertől. A többi érték közül a Downstream a legrejtélyesebb. A szó a böngésző és a kiszolgáló közötti Proxy szerverekre utal. Ez azt jelenti, hogy a Proxyk és a böngésző is tárolhatja a már egyszer letöltött tartalmat.
Összegezve: Any: mindhárom résztvevő Cache-elhet. DownStream: a szervert kivéve mindenki, így a Proxy-k és a kliens is. ServerAndClient: Proxy-k nem. Az előbbieket megismerve azt gondolhatnánk, kidobhatjuk a korábbi házibarkács Cache implementációnkat, hisz az OutputCache segítségével nulla programsorral megoldható a tartalom kiszolgálásának gyorsítása, ráadásul az OutputCache még a szerveren kívül állókat is képes rávenni a tárolásra, ezzel sokkal hatékonyabb megoldást kínálva.
Persze az élet nem ilyen rózsaszín.
A korábbi OutputCache sor hatására az ASP.NET a következő releváns fejléceket küldi a böngészőnek:

```
HTTP/1.1 200 OK
Cache-Control: public, max-age=863971
Expires: Fri, 04 Mar 2005 01:06:57 GMT
Last-Modified: Tue, 22 Feb 2005 01:06:57 GMT
Vary: *
Content-Type: image/bmp
Content-Length: 643
```

Ebből okulnia kellene a böngészőknek, ám a sors fintora, hogy az IE nem fogja fel, mit sugallnak neki a fejlécek, és nem tárolja le a tartalmat, így minden egyes Refreshre újra leszívja a dinamikusan generált tartalmat. Lehet, hogy ez csak az én gépemen probléma, de én így jártam.
A Mozilla Firefox örömmel engedelmeskedett a cache fejléceknek, és emiatt azon látványosan gyorsabban töltődnek be azok a lapok, amelyeken sok dinamikusan generált kép van. Az alábbi lappal teszteltem a böngészőket (downloadtest.aspx):

```
<body>
  <img src="download.aspx?id=0736" /><br />
  <img src="download.aspx?id=0877" /><br />
.
</body>
```

A webszerver logjában ezt láthatjuk:
Firefox:

```
02:00:24 GET /WebSite1/downloadtest.aspx - 200
02:00:24 GET /WebSite1/download.aspx id=0877 304
02:00:24 GET /WebSite1/download.aspx id=0736 304
.
```

IE6:

```
02:00:27 GET /WebSite1/downloadtest.aspx - 200
02:00:27 GET /WebSite1/download.aspx id=0736 200
02:00:27 GET /WebSite1/download.aspx id=0877 200
.
```

A Firefox kérésére a webkiszolgáló valami „furcsa" 304-es kóddal válaszol, míg az IE kérésére egyszerűen visszaadja választ. A különbség abban van, ahogyan a kéréseket megfogalmazzák a böngészők. Hamarosan kiderül mi ez.

Valami olyan megoldás kellene, amely egyrészt megy IE-vel is, másrészt olyan okos is, hogy képes legyen lekezelni azt a helyzetet, ha frissül az adatbázisban tárolt tétel tartalma, így a forgalom minimalizálása mellett mindig friss tartalmat kapnak az ügyfelek. A megoldás kulcsa a Firefox által helyesen alkalmazott az If-Modified-Since header.

Az If-Modified-Since HTTP header

Ez a fejléc kiváló szolgálatot tehet a hálózati forgalom csökkentésében. A működése elég egyszerű. A kiszolgáló leküld egy fejlécet, amelyben jelzi, hogy a tartalom legutóbbi frissítési ideje ekkor és ekkor történt, és a tartalom tárolható.

```
HTTP/1.1 200 OK
Cache-Control: private
Last-Modified: Sat, 01 Jan 2000 08:00:00 GMT
.
```

A böngésző a tartalom második lekérésekor (pl. Refresh gomb megnyomása) nem tölti le újra azt HTTP GET metódus segítségével, hanem úgy küldi el a kérést, hogy „szerver, csak akkor kérem tőled a tartalmat, ha az módosítva van xy időpont óta":

```
GET /WebSite1/download.aspx?id=1622 HTTP/1.1
If-Modified-Since: Sat, 01 Jan 2000 08:00:00 GMT
.

Ha a tartalom nem változott, akkor a szerver
ezt válaszolja:

HTTP/1.1 304 Not Modified
Content-Length: 0
.
```

Azaz a böngésző csak felszól, van-e frissebb verzió, ha nincs, felhasználja a saját Cache-elt tartalmát. Az If-Modified-Since lekezeléséhez nincs beépített támogatás az ASP.NET-ben, de kézzel implementálni nem nagy tudomány. Pontosabban az OutputCache fix értékekkel kezeli, de az IE nem hajlandó szót fogadni neki. De a mi megoldásunkat szeretni fogja.
Figyelni kell a kérésben az If-Modified-Since fejléc jelenlétét. Ha jelen van, ki kell venni annak értékét. Az a dátum lesz stringként ábrázolva, 0. időzónában leírva, amely a kliens által tárolt tartalom dátumát tartalmazza. A feladatunk ezen dátum összehasonlítása a szerveren tárolt tartalom frissítési dátumával. Ha a két dátum nem egyenlő, akkor beállítjuk a Last-Modified fejlécet a forrás dátumára és beállítjuk, hogy a böngésző tárolhatja a tartalmat. Ha a böngészőnél elég friss tartalom van, akkor HTTP 304-es kóddal jelezzük, hogy a böngésző által tárolt tartalom friss, és nem küldünk neki semmilyen tartalmat a HTTP válasz törzsében (ezt láttuk a Mozilla kéréseinek logjában). Azaz ilyenkor csak fejlécek mennek vissza a böngészőhöz.
Az előbb leírtak kezelésére létrehoztam egy kis segédosztályt, amelyet 2.0-s ASP.NET-ben az App_Code könyvtárba kell elmenteni:

```
public class IfModifiedSinceHelper {
  private static bool IsCachedVersionOkay(
    DateTime lastModification) {
    string ifModified =
      HttpContext.Current.Request.
      Headers["If-Modified-Since"];
```

```
if (ifModified != null) {
    return ifModified ==
      lastModification.ToUniversalTime()
      ToString("r");
  }
  return false;
 }
public static bool HandleHeader(
 DateTime lastModification) {
  HttpResponse response =
    HttpContext.Current.Response;
  if (IsCachedVersionOkay(lastModification)) {
    response.StatusCode = 304;
    response.SuppressContent = true;
    return true;
  }
  else {
    response.Cache.SetLastModified(
    lastModification);
    response.Cache.SetCacheability(
      HttpCacheability.ServerAndPrivate);
      //Lehet Public is
    return false;
  }
 }
}
```

A HandleHeader metódusnak át kell adni egy dátumot, amely a kérdéses tartalom utolsó módosítási dátumát tartalmazza. Érdemes az osztály működését összevetni az előző oldal végén található algoritmusleírással. Ha a metódus true-val tér vissza, akkor a lapnak már semmi dolga, befejezheti a futását. Ha false-szal, akkor le kell generálni a kimeneti tartalmat, a fejlécek kezelését már elintézte a segédosztályunk. A korábbi adatbázis-tartalom Cache-elése című kódot az alábbi módon kiegészítve máris működik az intelligens cachelésünk:

```
if (!IfModifiedSinceHelper.HandleHeader(
  new DateTime(2000, 01, 01, 00, 00, 00))) {
  A korábbi adatbázistartalom Cache-elése kód
}
```

A paraméterként átadott dátummal kicsit csaltam. Ott a tartalom valós utolsó módosítási dátumát kellene kivennem az adatbázisból. Azonban esetünkben egy cikkhez feltöltött kép soha nem módosítható, legfeljebb törölhető. Ezért azt hazudom, a tartalom utoljára mindig 2000. 01. 01-én módosult. Más tartalom esetén az adatbázisból kérdeztem le egy adott tétel utolsó módosítási dátumát. Általában ez sokkal kisebb költségű lekérdezéssel is megoldható, mint a tényleges tartalom kiolvasása, megszerkesztése. Ezt a megoldást használtam a tartalmat összegző RSS kimenetnél, amely egy XML formátumú leírás a legfrissebb cikkekről. Mivel az RSS tartalmat nem böngészők, hanem RSS Aggregátorok, olvasók töltik le általában óránként, a Last Modified nélkül nagyon nagy terhelést kapott a szerver és ráadásul jelentős, redundáns hálózati forgalmat is generált. Az RSS-hez az utolsó módosítást egy egyszerű SELECT MAX(...)-szal lekérdezve és a fenti módon a headereket lekezelve fellélegzett a kiszolgáló. Az előbbi lekérdezés egy összetett Cover index-szel nagyon gyorssá tehető.

Cache az ASP.NET 2.0-ban

Az 1.x verzió Cache-ét nagyon szeretjük, de elzárták előlünk új CacheDependency-kkel való bővítés lehetőségét, a CacheDependency osztály sealed, azaz nem lehet leszármaztatni belőle. Egy osztályt leszármaztatásra tervezni egyáltalán nem könnyű feladat, és az 1.x-ben még nem volt ez fontos tervezési szempont. Közben azonban kaptak pár évet a fejlesztők, így kinyithatták és dokumentálhatták az osztály működését. Azért olyan kritikus egy CacheDependency jó megvalósítása, mert nagyon intim kapcsolata van az ASP.NET Cache rendszerével, így ha hibásan működik, megőrülhet tőle a teljes ASP.NET Cache. A kibővítési lehetőséggel az ASP.NET csapat is élt, és létrehozott egy SqlCacheDependency leszármazottat. A neve elég jól jellemzi, őt adatbázis-alapú függőségek felépítéséhez lehet felhasználni.
Hogyan kell ezt elképzelni? Lekérdezek valamit az adatbázisból: kérem az 1960 után született alkalmazottak listáját. Tárolom az eredményeket Cache-ben. Ezek után elvárom, hogy a Cache-ből kiessen a tartalom, ha a lekérdezés eredményhalmaza változik. Vagy ha a lekérdezést alkotó alaptáblák változnak? Az első megoldás sokkal finomabb felbontású, de jelentős erőforrást igényelhet az adatbáziszervertől az értesítendő Cache elemek nyilvántartása és értesítése. A második megoldás viszont elég bután és gyakran kiejti a tételeket a tárból, de nagyon kis adatbázis-erőforrásra van szükség a nyilvántartáshoz. A lekérdezés alapú módszer csak az adatbázis-kezelő belső támogatásával oldható meg – legalábbis átlagemberek számára — ezért az csak SQL Server 2005 alapon fog menni. A tábla alapú megoldás működik már SQL Server 7-tel is. A táblaalapú módszer működése elég egyszerű. Az adatbázisra és abban a megfelelő táblákra engedélyezzük a Cache-elést az aspnet_regsql.exe interaktív (Windows alapú) futtatásával vagy parancssori paraméterezésével, esetleg programból az SqlCacheDependencyAdmin osztály felhasználásával.

Adatbázis felkészítésre Cache-elésre:

```
aspnet_regsql.exe -E -ed -d Northwind
```

A változások követésére létrejön az adatbázisban egy AspNet_SqlCacheTablesForChangeNotification nevű tábla:

tableName	nvarchar	450
notificationCreated	datetime	8
changeId	int	4

Egy-egy táblára engedélyezzük a módosítások követését:

```
aspnet_regsql.exe -E -et -d Northwind -t Employees
```

Az előbbi táblában minden nyomonkövetendő táblához keletkezik egy sor:

Employees	2/21/2005 11:00:59 PM	0

A tábla változtatásait követendő a tábla kap egy triggert:

```
CREATE TRIGGER
[Employees_AspNet_SqlCacheNotification_Trigger] ON
[Employees]
FOR INSERT, UPDATE, DELETE
AS
BEGIN
SET NOCOUNT ON
EXEC
dbo.AspNet_SqlCacheUpdateChangeIdStoredProcedur
N'Employees'
END
```

A hivatkozott sp egyszerűen megnöveli a táblához tartozó sorban az utolsó, verziót tartalmazó oszlop értékét:

```
ALTER PROCEDURE
dbo.AspNet_SqlCacheUpdateChangeIdStoredProcedure
  @tableName NVARCHAR(450)
AS
BEGIN
UPDATE
dbo.AspNet_SqlCacheTablesForChangeNotification
WITH (ROWLOCK)
SET changeId = changeId + 1
WHERE tableName = @tableName
END
```

Szimpatikus, hogy a ROWLOCK table hinttel próbálják minimalizálni a nyomkövetés teljesítménycsökkentő hatását az adatbázisra. Látható, hogy a módszer tábla alapú, és nem lekérdezés alapú módosításkövetést tesz lehetővé. Az ASP.NET-ben valakinek figyelni kell a tábla változásait, mert az SQL Server 7 vagy 2000 nem fog visszajelezni az ASP.NET-nek (az SQL 2005 igen!).
A figyelést, pollozást a web.configban kell beállítani, a konfigurációs elemek értelme elég könnyen kitalálható.

```
<configuration xmlns=
„http://schemas.microsoft.com/.NetConfiguration/
↪ v2.0">
  <connectionStrings>
    <add name="localNorthwind"
     connectionString="Data Source=.;
     Initial Catalog=Northwind;
     Integrated Security=yes;"/>
  </connectionStrings>

  <system.web>
    <caching>
      <sqlCacheDependency enabled="true"
       pollTime="10000" >
        <databases>
          <add name="localNorthwindDB"
             connectionStringName =
             „localNorthwind" />
        </databases>
      </sqlCacheDependency>
    </caching>
  </system.web>
</configuration>
```

Ezen hosszadalmas, most nem tárgyalt, de némi security konfigurálással tarkított, egyszeri bemosakodás után a függőség felépítése már pofonegyszerű:

```
Cache.Insert(„d", cachedData,
  new SqlCacheDependency(
  „localNorthwindDB", „Employees"));
```

Azaz, amint változik az Employees tábla tartalma, a Cache-elt tartalom kiesik a memóriából (10 másodpercen belül). Ha a cikksorozatom második részében már ismertetett új DataSource objektumokat használjuk adatelérésre, azok automatikusan tudják használni az előbb bekonfigurált Cache függőségi lehetőséget.

```
<asp:SqlDataSource ConnectionString="<%$
ConnectionStrings:NorthwindConnectionString1 %>"
SqlCacheDependency="localNorthwindDB:Employees"
```

Zárszó

Az igazi csemege az SQL Server 2005 alapú függőségek kialakítása, ugyanis azzal pollozás nélkül is megoldható a tartalom eltávolítása a Cache-ből. Következő cikkemben – többek között – ezt veszem górcső alá.

Soczó Zsolt
zsolt.soczo@netacademia.net
A szerző a NetAcademia vezető fejlesztőoktatója
ASP.NET MVP, MCSE, MCSD, MCDBA, MCT

A cikkben szereplő URL-ek:

[1] netacademia.net/tudastar
[2] netacademia.net/tudastar/articlepage.aspx?upid=5233
[3] www.blunck.info/iehttpheaders.html

Windows szolgáltatások 6. rész

A LocalSystem

Haladva a kirakós játékban, ebben a cikkben 15 újabb szolgáltatás ismertetése következik, persze továbbra is kizárólag a Windows Server 2003 Standard verziójának alapértelmezett telepítése során felkerülő szervizek közül. Már csak két rész van hátra ahhoz, hogy a közel 100 alapszolgáltatás mindegyikét megismerjük.

IPSec Services
(IPSec-szolgáltatások)

A szerviz rövid neve: PolicyAgent
Az alkalmazás neve: lsass.exe
Függés: Remote Procedure Call, TCP/IP Protocol Driver, IPSEC Driver
Függesztés: –
Porthasználat: TCP: 50 (ESP), 51 (AH), UDP: 500 (ISAKMP), UDP 4500 (ISAKMP2>NAT-T)
Alapértelmezett indítás: automatikus

A Windows 2000-nél ezt a szervizt „IPSEC Policy Agent" néven azonosíthatjuk, ám nem valószínű, hogy a névváltoztatás sokakat zavarna, hiszen jól ismerjük ezt a szolgáltatást, jól cseng a neve.

Az IPSec egy olyan nyílt szabvány, amelyet a Microsoft kicsit felturbózott, így tartományi környezetben, Kerberos v5 hitelesítéssel és akár a Csoportházirendből vezérelve is használhatjuk. Ez persze nem jelenti azt, hogy e feltételek nélkül nem fog működni, sőt, ez annyira nem így van, hogy pl. azon kevés házirend opciók közé is bekerült, amelyek egy helyi házirendben is a rendelkezésünkre állnak – azaz tartomány nélkül is alkalmazhatóak.

Az IPSec alacsony hálózati rétegben működik, és mivel átlátható (azaz az alkalmazások semmit nem vesznek észre a jelenlétéből), takarékos is, ugyanakkor minimális átviteli sebesség veszteséggel jár.

Az IPSec hash algoritmusokkal és a nyilvános kulcsú titkosítással védi a végpontok közötti kommunikációt. Egy kulcsot használ az adatok aláírásához és titkosításához, és egy másikat az üzenet ellenőrzéséhez illetve a visszafejtéshez. Négyféle módszerrel is védelmet nyújthat:

- Hitelesítés: minden csomagot hitelesít, illetve minden csomag eredetét ellenőrzi, ergo garantált, hogy a csomag attól származik, akitől szeretnénk megkapni (a hitelesítés típusa lehet Kerberos, digitális aláírás, vagy egy megosztott titkos kulcs, azaz pl. egy jelszó.
- Integritás ellenőrzés: az IPSec garantálja azt is, hogy nem lehet módosítani a csomagot a végpontok között.
- Ismétlés-megtagadás: mivel teljesen egyedi módon ír alá minden egyes csomagot, nincs lehetőség újra használni vagy újraküldeni ezeket.
- Megbízhatóság: a csomagok titkosítása miatt, csak a megfelelő kulccsal rendelkező célpont tudja „elolvasni" a tartalmat.

Az IPSec két protokollt használ: az AH (Authentication Header) and ESP (Encapsulating Security Payload) protokollokat. Az AH autentikál, figyeli az integritást és az újraküldést illetve alá is írja a csomagokat az SHA (Secure Hash Algorithm) vagy az MD5 (Message Digest 5) algoritmus segítségével.

Az ESP mindent elvégez, amit az AH, plusz titkosít is a jó öreg DES vagy a fiatalabb és izmosabb 3DES algoritmussal. A két protokoll persze kombinálható is, pl. a felhasználási területtől (intranet/internet) függően. Fontos szerepe van az ún. IKE (Internet Key Exchange) vagy ahogy a protokoll fejlesztője Hilarie Orman anno elnevezte: ISAKMP/Oakley (merthogy az IKE a IETF, Internet Engineering Task Force **[1]** elnevezése) protokollnak is, ugyanis a felismert titkosítás esetén az IPSec driver utasítására előzetesen egyezteti a két fél közötti algoritmusokat, autentikációt és a kulcsokkal kapcsolatos szabályokat egyaránt.

Az IPSec használata során az igényeink szerint kijelölt gépek teljes IP forgalma szűrés áldozatává válhat, így egyszerűen meghatározható, hogy az adott kommunikáció engedélyezett, biztonságos vagy blokkolt-e az IP-címtartományok, IP protokollok vagy a megadott TCP és UDP portok alapján.

Az IPSec-et gyakran használjuk biztonságos csatornák támogatására illetve kialakítására, azaz pl. VPN kapcsolatnál, akár az L2TP protokoll „alá", akár IPSec alagutak formájában (az utóbbi pl. az ISA 2004-gyel már viszonylag egyszerűen beizzítható).

A szerviz maga tehát a végpontok közötti biztonságot fokozza, ott áll az IPSec házirendek végrehajtása mögött, kontrollálja az IPSec driver működését, akár az intraneten, akár a nyilvános hálózatokon keresztül.

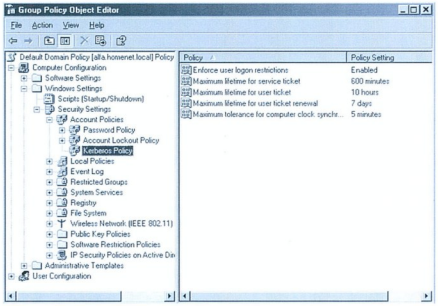
Abban az esetben, ha leállítjuk vagy letiltjuk, akkor az IPSec alkalmazhatósága természetesen minden folyamatban megszűnik, más ismert következménye viszont nincs ennek a lépésnek.

Kerberos Key Distribution Center (Kerberos kulcsszolgáltató)

A szerviz rövid neve: Kdc
Az alkalmazás neve: lsass.exe
Függés: RPC, AFD Networking Support Environment
Függesztés: -
Porthasználat: TCP: 88, 544; UDP: 88, 464
Alapértelmezett indítás: letiltva

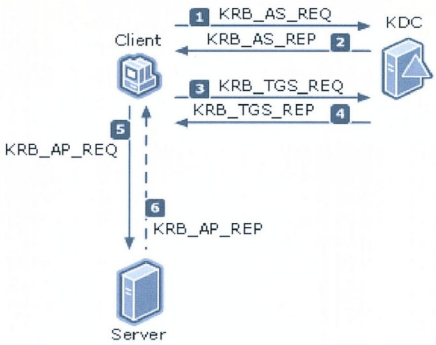
Ez a szerviz kulcsfontosságú, mivel ennek segítségével tudják a felhasználók a Kerberos v5 hitelesítési protokollt tartományi belépésre illetve és használni. A KDC két Kerberos v5 szolgáltatást is nyújt:

- Hitelesítés, azaz a TGT-k (Ticket Granting Ticket = jegymegadó jegy) kibocsátása tartományon belül, illetve a megbízható kapcsolatban lévő tartományok között. Erre elsősorban a tartomány szolgáltatásait igénybe vevő felhasználónak van szüksége, és azért, hogy egy ún. szolgáltatásjegyet tudjon majd kérni a második körben a jegymegadási szolgáltatótól (lásd lentebb). A TGT általában egy-egy munkamenetre kapható meg, ami az alapbeállítás szerint 10 óra (max. 7 nap lehet, a következő ábrán látszik, hogy hol lehet állítani a Csoportházirendben), de persze ennek lejártakor már egyszerűbben lehet újra igényelni.



A Kerberos házirend

- Jegymegadási szolgáltatás (Ticket-Granting Service = TGS) , azaz a TGT-k „begyűjtője" és ha minden klappol akkor a szolgáltatásjegyek kiosztója. Ez utóbbi azért fontos, mert ezt „mutatja" be a kliens a kért hálózati szolgáltatásnak. A szolgáltatásjegy a klienst hitelesíti a szolgáltatás számára és vica versa.



A hitelesítési folyamat lépései

A szolgáltatás akkor települ csak, ha a szervert tartományvezérlővé léptetjük elő, akkor viszont biztosan, ergo minden DC KDC is egyben (minden tartományi gép meg Kerberos ügyfél is). Ha viszont egy DC-n leállítjuk vagy letiltjuk, az a DC nem lépteti be a felhasználókat.

Logical Disk Manager (Logikai lemezkezelő)

A szerviz rövid neve: Dmserver
Az alkalmazás neve: dmserver.dll (svchost.exe)
Függés: Remote Procedure Call, Plug and Play
Függesztés: Logical Disk Manager Administrative Service
Porthasználat: –
Alapértelmezett indítás: automatikus

A Windows 2000 óta újfajta lemezkezelője van az operációs rendszernek, ez pedig a Logical Disk Manager (ami anno a VERITAS Volume Manager „könnyített" változataként debütált). Az LDM feladata detektálni és monitorozni a merevlemezekkel kapcsolatos változásokat és elküldeni a lemez- és kötetinformációkat a Logical Disk Manager Administrative szerviznek (lásd következő szolgáltatás). Tehát a szerviz figyeli az olyan Plug and Play eseményeket, amelyek a merevlemezekkel kapcsolatosak és változás esetén jelez.
Ha letiltjuk vagy leállítjuk, akkor a dinamikus lemezek állapot- és konfigurációs beállításai nem frissülnek tovább és a rendszerbe illesztett új lemezeket sem „veszi észre" a Windows. Emellett a Disk Management MMC is randa hibaüzenetekkel lep meg bennünket: „Unable to connect to Logical Disk Manager service", és így nem is használható tovább.

Logical Disk Manager Administrative Service (Logikai lemezkezelő felügyeleti szolgáltatás)

A szerviz rövid neve: Dmadmin
Az alkalmazás neve: dmadmin.exe /com
Függés: Logical Disk Manager, Remote Procedure Call, Plug and Play
Függesztés: –
Porthasználat: –
Alapértelmezett indítás: kézi, leállítva

Ez a szerviz az előbb említett jelzések lekezelését végzi. Mivel kézi indítású, csak akkor indul el, ha elindítjuk a Disk Ma-

nagement MMC-t vagy ha egy dinamikus lemez állapota megváltozik, azaz pl. konvertálás történik, a hibatűrő lemezek visszaállítása zajlik, de egy formattálás, illetve a pagefile változása is kiválthatja a szolgáltatás indítását. Ezekben az esetekben a szerviz tehát elindul, megtörténik a konfiguráció változás, majd a szerviz leáll.
Ha letiltjuk, akkor a Disk Management MMC ugyanazt produkálja, mint az előző szolgáltatásnál.

Messenger
(Üzenetkezelő)

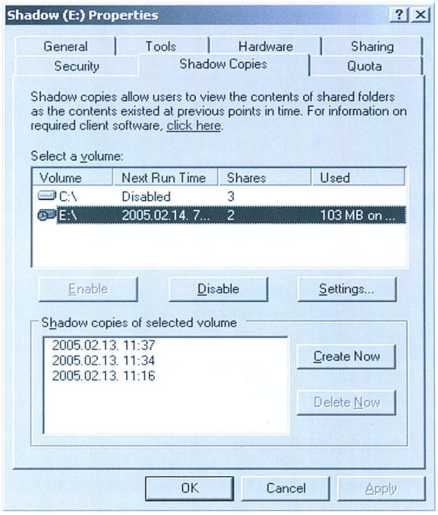
A szerviz rövid neve: Messenger
Az alkalmazás neve: msgsvc.dll (svchost.exe)
Függés: Remote Procedure Call, Plug and Play, Workstation, NetBIOS Interface
Függesztés: -
Porthasználat: -
Alapértelmezett indítás: letiltva

A Messenger szerviz közismert felhasználású: a konzolról a „net send" paranccsal indított illetve az Alerter szolgáltatás üzeneteit közvetíti két tartományi gép között (nyilván csak akkor ha a másikon is elindítottuk ez a szolgáltatást). Lehetőség van felhasználók csoportjainak illetve akár egy egész tartomány összes gépére is üzeneteket küldeni ezzel a szolgáltatással.
A Windows Server 2003-tól kezdve (a korábbi Windows-okkal szemben) már alapértelmezés szerint letiltott állapotú. Ennek a változásnak az (általam) valószínűsíthető oka egy kritikus sérülékenység felfedezése lehetett. Ez volt a „Messenger Service Spam", amely sok rémületet okozott egy időben, ugyanis valóban kéretlen üzeneteket kaphattunk bizonyos nyitott portok vagy hiányzó tűzfalak esetén az internet felől, éppúgy mintha a belső hálózatról jött volna **[2]**.
Ha a szervizt leállítjuk vagy letiltjuk, akkor a felhasználói illetve szoftveres üzenetek nem kézbesítődnek, a különböző programok riasztásai sem. Annyit még célszerű tudnunk erről a szolgáltatásról, hogy megtévesztő neve ellenére semmi köze nincs a Windows/MSN Messenger azonnali üzenetküldő alkalmazásokhoz.

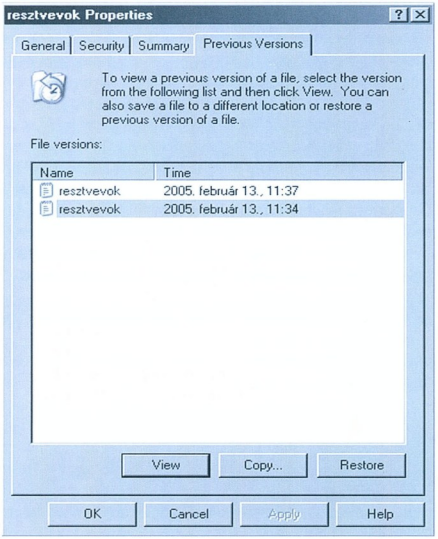
Microsoft Software Shadow Copy Provider
(Microsoft szoftverárnyékmásolat-szolgáltató)

A szerviz rövid neve: SwPrv
Az alkalmazás neve: swprv.dll (svchost.exe)
Függés: Remote Procedure Call
Függesztés: –
Porthasználat: –
Alapértelmezett indítás: kézi, leállítva

Ez a szerviz a Volume Shadow Copy (az ún. Kötetpillanatkép) szolgáltatás kiegészítője, azaz, az ezzel a technikával készített árnyékmásolatokat kezeli. Mint tudjuk, az árnyékmásolat szolgáltatás segítségével egész köteteket illetve megosztott mappákat tudunk a hagyományos mentési módszerek nélkül is megóvni a véletlen törlésektől és/vagy felülírásoktól, illetve az állományok időben különböző változatait is összehasonlíthatjuk ezzel a módszerrel.



Az árnyékmásolatokat tartalmazó kötet tulajdonságai



A kliens opciói

A használatához szerver oldalon engedélyeznünk kell az adott kötet tulajdonságai között, illetve kliens oldalon telepítenünk kell a szolgáltatás ügyfélprogramját, pl. a kiszolgáló erre a célra létrehozott mappájából, ami a következő helyen található (de alapértelmezés szerint nincs megosztva):

```
\\%Systemroot%\system32\clients\twclient
```

Az árnyékmásolatok kezelése a vssadmin.exe parancssori programmal történik, de sokat segíthetnek a Resource Kit kapcsolódó segédprogramjai, pl. a Volrest.exe, amellyel a szerver megosztott mappáiban árnyékmásolatokat tudunk keresni illetve vissza tudjuk állítani a törölt vagy felülírt állományok előző verzióit. Hasznos eszköz lehet még a szintén a ResKitben található VolPerf.exe (Shadow Copy Performance Counters), amely az árnyékmásolatokkal kapcsolatos tetemes mennyiségű teljesítményszámláló használatát teszi lehetővé. Ha a szervizt leállítjuk vagy letiltjuk, az árnyékmásolatok kezelési lehetősége „kiesik" a kezünkből.

Net Logon
(Hálózati bejelentkezés)

A szerviz rövid neve: Netlogon
Az alkalmazás neve: lsass.exe
Függés: Workstation
Függesztés: –
Porthasználat: TCP: 139, 888; UDP: 137, 138
Alapértelmezett indítás: kézi, leállítva (ám tartományi tagság esetén automatikusan indul)

Ez szerviz szintén kritikus jelentőségű, hiszen elsősorban a tartományvezérlő és a kliens közötti biztonságos csatornák létrehozásáért és fenntartásáért felel. Ez a speciális csatorna ahhoz szükséges, hogy megfelelő körülmények között hitelesíthesse magát a kliensről a felhasználó (amely folyamat során vissza is kapja a DC-től az azonosítóit és a jogosultságait > pass-through) illetve a szolgáltatások is. A számítógépek hitelesítésénél is fontos szerepe van, hiszen mindegyik tartományi fiókkal rendelkező gépnek szüksége van erre a biztonságos csatornára. És hogyan jön létre? Egyszerűsítve: a gép az LSA segítségével helyileg tárolt jelszava plusz az AD-ben tárolt jelszava összehasonlítása után, az adott jelszóval a Netlogon szerviz megteremti ezt a biztonságos csatornát. Persze, ha valamilyen okból már nem passzol a két jelszó (pl. a Csoportházirendben megadott változási kényszer hatása csak az egyik oldalon érvényesült azaz nem szinkronizálódik a jelszó), akkor a gép nem képes arra, hogy hitelesítse magát. Eme biztonságos csatorna létezését tesztelhetjük az

```
nltest /sc_query:tartomanynev
```

paranccsal és ha minden OK, akkor valami hasonló eredmény kapunk:

```
Flags: 30 HAS_IP  HAS_TIMESERV
Trusted DC Name \\a_DC_neve
Trusted DC Connection Status Status = 0 0x0
NERR_Success
The command completed successfully
```

Ha valami bibi van, akkor meg ezt:

```
Flags: 0
Trusted DC Name
Trusted DC Connection Status Status = 1311 0x51f
ERROR_NO_LOGON_SERVERS
The command completed successfully
```

Az nltest.exe a Support Tools csomag része, amelyet külön, a telepítő CD-ről kell felrakni (vagy létező SP esetén le kell tölteni, pl. XP SP2 esetén innen **[3]**).

Ezen kívül a Windows 2000 Server-től kezdve a Netlogon szolgáltatás egy másik fontos műveletet is elvégez: a DDNS-t felhasználva a különlegesen fontos, sokat emlegetett SRV (erőforrás-rekord) rekordokat publikálja a DNS zónákba. Ezért hangzik el sűrűn DNS hibák megoldására csodaszerként a „net stop netlogon" ill. „net start netlogon" parancs (azaz egy NetLogon szerviz újraindítás), amely egy helyes TCP/IP konfiguráció esetén hatásos gyógyszer. Azért az, mert így könnyedén újraépíthető egy hosszú nevű és rejtélyesnek tűnő bejegyzésektől (a DC-k szolgáltatásainak, pl. Kerberos, GC, LDAP nevét, IP címét, súlyát eláruló rekordok) hemzsegő tartományi zóna a DNS szerverben. Hogy honnan szedi az adatokat, amellyel feltölti a zónát? A \\%Systemroot%\system32\config\netlogon.dns file-ból, amelyben szépen összerendezett formában megtalálható minden egyes SRV rekord (ezért pl. ha nem lehet DDNS-t használni ezt az állományt kell ide-oda másolgatni).
De meg kell említeni még azt is, hogy a Net Logon szerviz felölel pár kiszolgáló generációt, hiszen ennek segítségével valósul meg az RPC alapú szinkronizáció az egy NT4-es PDC és a BDC-k között.
Ha netalántán leállítjuk vagy letiltjuk ezt a szolgáltatást, akkor ennek a lépésnek valóban kellemetlen következményei lesznek, hiszen nem működik tovább a felhasználók/szolgáltatások hitelesítése, és a DC-k nem tudnak rekordokat regisztrálni a DNS zónákba. Emellett az adott gépről a tartományhoz csatlakozás további problémákat okoz majd, valamint minden további NTLM alapú hitelesítés is megtagadásra kerül, a kiszolgáló pedig láthatatlanná válik a kliensek felől. Szóval ne kísértsük az ördögöt.

NetMeeting Remote Desktop Sharing
(NetMeeting távoli asztalmegosztás)

A szerviz rövid neve: Mnmsrvc
Az alkalmazás neve: mnmsrvc.exe
Függés: –
Függesztés: –
Porthasználat: TCP: 215, 731, 3389; RDP: 3389
Alapértelmezett indítás: letiltva

Kisebb jelentőségű szolgáltatásról van szó, amely csak annyit tesz hozzá a lehetőségeinkhez, hogy NetMeeting konferenciaszoftver használata közben engedélyezi a Windows Asztal átvételét a másik gépről. Ezt a lehetőséget közvetlenül a NetMeetingből érhetjük el, és az engedélyezése során a szolgáltatás el is indul, illetve a leállításakor letiltásra kerül.
Ha manuálisan leállítjuk vagy letiltjuk, akkor a szerviz a „visszaszívja" a memóriából NetMeeting képernyő meghajtót, ergo a távoli asztalmegosztás lehetősége megszűnik. Legyen is így (azaz maradjon letiltva), valószínűleg semmi szükségünk nem lesz rá.

Network Connections
(Hálózati kapcsolatok)

A szerviz rövid neve: Netman
Az alkalmazás neve: netman.dll (svchost.exe)
Függés: Remote Procedure Call
Függesztés: Internet Connection Firewall (ICF) / Internet Connection Sharing (ICS)
Porthasználat: –
Alapértelmezett indítás: kézi, elindítva

A „Hálózati kapcsolatok" ablakban szereplő „gyári" és általunk kreált kapcsolatokkal foglalkozik ez a szerviz. Felel a kapcsolatokhoz tartozó beállításokért, a Tálcán lévő kapcsolat állapotjelző ikonokért és az összes többi belső vagy külső alkalmazás csak ezen a szolgáltatáson keresztül érheti el az adott kapcsolat beállításait.
Annak ellenére, hogy manuális indítású, mindig elindul automatikusan, ha van legalább egy hálózati kapcsolatunk. Viszont ha letiltjuk, annak (többek között) a felsoroltak is következményei lesznek:

- Nem lehet új hálózati kapcsolatot készíteni
- A meglevő kapcsolatok ikonjai tovább látszanak, ám nem frissíthetőek, nem használhatóak illetve nem konfigurálhatóak
- A kapcsolatok állapotüzenetei megszűnnek
- Az Internet Connection Sharing szolgáltatás leáll
- A bejövő kapcsolatok/források észlelése (pl. WLAN) nem működik

Network DDE (Hálózati DDE)

A szerviz rövid neve: NetDDE
Az alkalmazás neve: netdde.exe
Függés: Network DDE DSDM
Függesztés: Clipbook
Porthasználat: –
Alapértelmezett indítás: letiltva

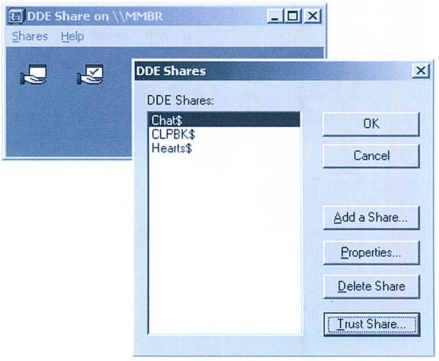
Hálózati támogatást és adatvédelmet nyújt a DDE (Dynamic Data Exchange) részére, amely régi ismerős: már a Windows 2.x óta velünk van. A dinamikus adatcserét (DDE) támogató programok adatokat és parancsokat tudnak egymással cserélni, a NetDDE és a NetBIOS alkalmazásával hálózatban is.
Alapértelmezés szerint tiltva van, tehát a NetDDE-t használó alkalmazások (a hálózat azon másik gépén is, ahonnan a kezdeményezés történik) gondban lesznek. Ha szükségünk van erre a szolgáltatásra, akkor elég, ha kézi indításúra tesszük, mert így csak igény szerint indul majd el, az adott alkalmazás kérése alapján. Ebben az esetben a távoli gépről is befolyásolható a szerviz indítása, tehát ez szintén nem okoz problémát.

Network DDE DSDM (Hálózati DDE DSDM)

A szerviz rövid neve: NetDDEdsdm
Az alkalmazás neve: netdde.exe
Függés: –
Függesztés: Network DDE
Porthasználat: –
Alapértelmezett indítás: letiltva

Az előző szervizhez szorosan kapcsolódó szolgáltatásról van szó, ugyanis mielőtt a NetDDE használatba kerülne, az ún. NetDDE megosztásoknak valahogyan a rendszerbe kell kerülniük. Ez a művelet kb. ugyanannyit jelent, mint amikor mappa megosztásokat készítünk, majd a felhasználóknak megengedjük, hogy kapcsolódjanak ezekhez. A különbség viszont az, hogy a NetDDE megosztásokat az alkalmazások többnyire maguk készítik, bár számunkra is nyitott a lehetőség, például a következő ábrán látható módon, a ddeshare.exe segítségével. A NetDDE megosztásokat megtekinthetjük a következő regisztrációs adatbázis kulcs alatt is:

```
HKEY_CURRENT_USER\Software\Microsoft\NetDDE\DDE
Trusted Shares
```



Műveletek a ddeshare.exe-vel

Az ilyen típusú megosztások kezeléséhez, a helyi és a megbízható megosztások nyilvántartásához, azaz egy adatbázis üzemeltetéséhez van tehát szükség a DDE Share Database Manager-re (más néven erre a szervizre), amelyet viszont kizárólag a NetDDE szerviz használhat.
Amennyiben leállítjuk, vagy letiltjuk, akkor a NetDDE adatbázishoz nem lehet kapcsolódni, így az ezt igénylő alkalmazások működése csorbul. Ha viszont nem használjuk, inkább maradjon az alapértelmezett módban, azaz letiltva.

Network Location Awareness (Hálózati hely felismerése)

A szerviz rövid neve: NLA
Az alkalmazás neve: mswsock.dll (svchost.exe)
Függés: AFD Networking Support Environment, TCP/IP Protocol Driver, IPSEC Driver
Függesztés: Internet Connection Firewall (ICF) / Internet Connection Sharing (ICS)
Porthasználat: –
Alapértelmezett indítás: kézi, elindítva

Gyűjti és tárolja a hálózati beállításokkal kapcsolatos információkat, mint pl. az IP cím és a tartománynév változás és informálja a kapcsolódó alkalmazásokat ezekről a változásokról. Gyakorlatilag ez az a szerviz, ami lehetővé teszi pl. a Windows XP óta bevezetett alternatív TCP/IP konfiguráció működését, valamint a gördülékeny (felhasználói beavatkozás nélküli) váltást a különböző WLAN hálózatok között.
(Kis kitérő: sajnos a vezetékes és vezeték nélküli hálózatok közötti váltás nem igazán gördülékeny, gondolok itt arra, hogy pl. a vezetékeshez kapcsolódva a WLAN kapcsolat nem válik le automatikusan illetve fordítva sem. Egyidejű működés esetén meg előjönnek/előjöhetnek névfeloldási és/vagy átjáró problémák, szóval egyáltalán nem fenékig tejfél.
Igaz, XP SP2 és a hamarosan elkészülő W2K3 SP1 esetén lehet fékezni az automatizmust azzal, hogy a WLAN kapcsolatoknál akár egyesével választhatunk: manuálisra tesszük a kapcsolódást vagy automatikussá, de ez még mindig nem az igazi, talán majd a Longhorn NLA-val… (lásd: Longhorn Network Location Awareness white paper **[4]**)
Ez a szerviz is igény szerint indul, ha manuálisra állítjuk. Ha viszont letiltjuk az észlelés és a konfigurációváltás értelemszerűen nem fog működni. Az NLA azért is igény szerint in-

duló szolgáltatás, mert néhány kritikus rendszerkomponens is használja (pl. a Winlogon.exe), azaz ha letiltjuk, ezek a rendszerkomponensek ezt érzékelik és újra és újra megpróbálják elindítani, amelynek csak egyetlen következménye lesz: telefirkálják az eseménynaplót. Leállítanunk is felesleges, hiszen mindig újra fogják indítani, ezért inkább hagyjuk meg az alapértelmezett állapotban.

NT LM Security Support Provider
(NT LM biztonsági támogatás szolgáltatója)

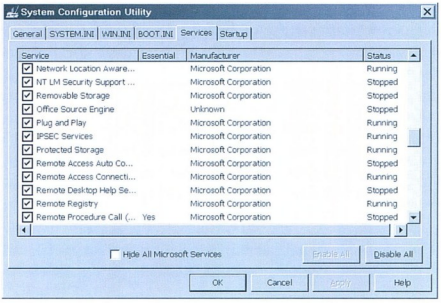
A szerviz rövid neve: NtLmSsp
Az alkalmazás neve: lsass.exe
Függés: services.exe
Függesztés: Telnet, WINS
Porthasználat: RPC/TCP > dinamikus porthasználat
Alapértelmezett indítás: kézi, leállítva

Ez a Windows NT korszakból megörökölt szerviz teszi lehetővé a felhasználók belépését a tartományba abban az esetben, ha a LanMan, NTLM vagy az NTLMv2 a hitelesítés módja a hálózatunkban. Ez tipikusan a Windows 9x és Windows NT operációs rendszerrel dolgozó számítógépek esetén van így. Ha leállítjuk vagy letiltjuk, akkor ezekről a gépekről a tartományi belépés és az erőforrások elérése lehetetlenné válik.

Plug and Play
(Plug and Play)

A szerviz rövid neve: PlugPlay
Az alkalmazás neve: Services.exe
Függés: –
Függesztés: Fax, Logical Disk Manager, Logical Disk Manager Administrative Service, Smartcard, Messenger, Telephony, Remote Access Auto Connection Manager, Remote Access Connection Manager, Internet Connection Firewall (ICF) / Internet Connection Sharing (ICS), Virtual Disk Service, Windows Audio
Porthasználat: –
Alapértelmezett indítás: automatikus

Ismerős vívmány a Plug and Play, értelemszerűen ez a szerviz a hardver eszközök felismerését és rendszerbe illesztését műveli, zéró vagy minimális felhasználói beavatkozással. Valóban nagy segítség a felehasználók számára ez a szolgáltatás, és mára szépen ki is kristályosodott, szemben a Windows 9x-es hőskorban, amikor még rendszeresen a „Plug and Pray" („Dugd be és imádkozz") kifejezéssel illették.
A szerviz kicsit rendhagyó módon működik, mert a Services MMC-ben nem lehetséges leállítani vagy letiltani. Ennek oka az operációs rendszer stabilitásának megőrzése (nézzük meg mennyi szerviz függ a jelenlététől). Ha viszont az msconfig.exe segítségével mégis leállítjuk, akkor egy újraindítás után a Device Manager-ből eltűnnek a regisztrált hardver elemek. Kipróbáltam, tényleg így van, de nem biztos, hogy jó ötlet ezzel játszani, mert nálam például az IntelliMouse driver olyan ciklikus Error Reporting manőverbe kezdett, hogy alig bírtam újraindítani a gépet.



Bánjunk óvatosan az egyébként hasznos System Configuration Utility-vel

Portable Media Serial Number
(Hordozható lejátszó sorozatszáma)

A szerviz rövid neve: WmdmPmSN
Az alkalmazás neve: mspmsnsv.dll (svchost.exe)
Függés: –
Függesztés: –
Porthasználat: –
Alapértelmezett indítás: kézi, leállítva

Végül egy kevésbé izgalmas, ám bizonyos esetekben mégis fontos szolgáltatás következik, amely már a Windows 2000-nél is létezett, csak másképp hívták (WMDM PMSP Service). Ez a szolgáltatás lehetővé teszi a WMDM (Windows Media Device Manager) számára, hogy a hordozható média lejátszó (mp3 lejátszók, Rio és társai) háttértárait azonosítsa, így a tartalom biztonságos körülmények között lesz másolható. A legtöbb hordozható háttértár rendelkezik egyedi azonosítóval, pl. a CompactFlash (az 1.3-as verziótól felfelé) vagy Iomega Zip, Jaz és az ún. Clik! lemezek is valamint a legtöbb SmartMedia kártya is ebbe a körbe sorolható, ugyanúgy mint az újabb és közismertebb típusok is, mint pl. a MultiMedia Card (MMC), Secure Digital és a SONY MemoryStick kártyák. E procedúra nélkül az ilyen típusú háttértárakkal rendelkező lévő védett tartalmakhoz hozzáférni nem lehetséges, és ez persze akkor is bekövetkezhet, ha a szervizt letiltjuk.

GÁL TAMÁS
MCT, MCSE, MCSA, MVP
gtamas@tjszki.hu

A cikkben szereplő URL-ek:

[1] http://www.ietf.org
[2] http://www.stopmessengerspam.com/
[3] http://tinyurl.com/6xrl7
[4] http://msdn.microsoft.com/longhorn/?pull=/library/en-us/dnlong/html/lhnla.asp

Ami a hivatalos Microsoft tanfolyamokból kimaradt...

SharePoint Portal Server 2003 – jogosultságok

Rovatunkban újra a „csinálj magad intranetet" témát járjuk körbe. A Windows SharePoint Services és a SharePoint Portal Server 2003 jogosultságait futjuk át röviden, megnézzük milyen lehetőségeink vannak felhasználóink jogosultságainak kezelésére, meghatározására. Hogyan tudunk csoportokat kezelni és ezeket hozzáférési, műveleti jogokkal felruházni? Mindez hogyan hat a csoportmunka-helyekre?

A vállalati intranetek rohamos tempóval állnak át a SharePoint technológia adta lehetőségekre. Az információs sztrádák egyre több sávosak lesznek, minek következtében az intranetek egyre nagyobb fontosságot kapnak egy vállalat életében. Ezért e cikkben megismerkedünk azokkal a koncepcionális nézetekkel, hogy miként leszünk, vagy inkább miként vagyunk képesek a portálhelyeink csoporttagságainak jogosultságait kezelni.

A SharePoint Portal Server 2003 biztonsági beállításai első nekifutásra nagyon komplexnek látszanak. Ha alaposabban átnézzük ezeket a lehetőségeket, akkor láthatjuk, hogy két-három lépésben mégis nagyon könnyen és egyszerűen tudjuk a biztonsági beállításokat konfigurálni. Miután végére érünk a cikknek, könnyedén felügyeljük cégünk intranetjén a SharePoint felhasználói jogosultságokat, hozzáféréseket portálhelyeinken. Definiálni tudjuk a portálhelyeinket elérő (a portálhelyhez hozzáférő) felhasználóink jogait. Kezelni tudjuk a felhasználókat, csoportokat, és a portálunk elérési útjait. Megismerjük röviden a kapcsolódó riasztási, értesítési beállításokat.

Csoportok és jogok

A SharePoint Portal Server 2003 (továbbiakban: SPS), és a Windows SharePoint Services (továbbiakban: WSS) alapértelmezett modellként használja a biztonsági csoportokat és azok jogait a rendszer egészét tekintve. A portálhely csoportok tulajdonképpen speciális gyűjteményei a rendszerünk NT alapú felhasználóinak és csoportjainak. Minden SPS, és WSS biztonsági csoportnak a hozzárendelt biztonsággal kapcsolatos kezelési joga van a portálhelyek tekintetében. Képesek vagyunk ezeket a gyári biztonsági csoportokat egyszerűen módosítani, továbbá hozhatunk létre új biztonsági csoportokat, vagy akár tetszés szerinti kombinált megoldást is mixelhetünk. Lépjünk az SPS „Webhely beállításai"-ra (ehhez rendszergazdaként essünk neki portálunknak, és a jobb felső sarok felé tekintsünk). Itt az „Általános beállítások" résznél válasszuk a „Biztonsági és egyéb beállítások kezelése" menüt.

Tovább haladva a megjelenő oldalon a „Felhasználók és engedélyek" résznél válasszuk a „Egyhelyes csoportok kezelése" hivatkozást. Az SPS beépítve tartalmazza az alábbi csoportokat:

- *Olvasó csoport*. A csoport tagjai képesek megnézni a portálhelyeinken elhelyezett listák, dokumentumtárak tartalmát, illetve rálátással bírnak számukra engedélyezett oldalakra, de csak olvasási joggal.
- *Munkatárs csoport*. A csoport tagjai képesek a kijelzőket listázni, azok tartalmait módosítani, a dokumentumtárakat felügyelni. Képesek továbbá módosítani, illetve készíteni saját nézeteket a weblapokon.
- *Webhelytervező csoport*. A csoport tagjai képesek az SPS weblapjainak szerkesztésére, listák és dokumentumtárak létrehozására, akár a Microsoft Office FrontPage 2003 segítségével.
- *Rendszergazda csoport*. A csoport tagjai teljes felügyelettel, korlátozás nélkül rendelkeznek a webhely felett.

A csoport minden tagja képes konfigurálni az összes beállítást, kezelni a felhasználók és a csoportok jogosultságait, tagságait.

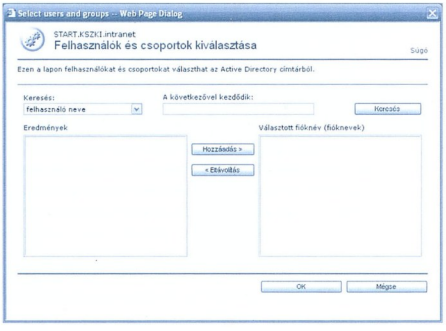
- *Tag csoport*. A csoport tagjai képesek megnézni, módosítani saját portálhelyeinek tartalmát, illetve ott tartalmat készíteni. A tagoknak lehetőségük van további webhelyek létrehozására.
- *Tartalomkezelő csoport*. A csoport tagjai készíteni tudnak alwebhelyeket, listákat, dokumentumtárakat, továbbá csoportmunka-helyeket, azaz WSS webhelyeket.

Egyedi csoportok

Lehetőségünk van portálhely tulajdonosként vagy rendszergazdaként egyedi csoportokat létrehozni a gyáriak mellé (vagy akár azok helyett). Kattintsunk az „Egyhelyes csoport hozzáadása" gombra:

A létrehozandó új csoporthoz tartozó jogköröket a rendelkezésre álló 24 alapjog szelektív kiválasztásával egyedileg konfigurálhatjuk annak függvényében, hogy miként szeretnénk a csoport tagjainak jogkörét megalkotni. A 24 alapjogot most itt nem részletezem, de elégséges a választék az egyéni igények kielégítéséhez.

Az így definiált és ezáltal életre keltett új csoport tagságának megadhatunk tartományi csoportokat, helyi csoportokat, illetve ezen csoportok tagjait egyenként is felvehetjük csoporttagnak. Ehhez kattintsunk az új csoportunk nevére, és kattintsunk a „Tagok felvétele" gombra. A megfelelő felhasználók kiválasztásához egy egyszerű kereső ablak áll rendelkezésünkre.



Jól látszik a folyamatban, hogy ugyanaz a menet áll rendelkezésünkre egy adott felhasználó csoportokhoz való hozzáadásához, vagy egy adott csoport tagjainak kiválasztásához. Ez jó így, hisz nagyban leegyszerűsíti, és átláthatóvá teszi a folyamat kezelését.

Optimálisan kétféleképpen lehetünk rendszergazdái az SPS-nek és a WSS-nek. Önálló szerveren a Windows rendszergazdai csoport helyi tagjaiként, illetve lehetünk tagjai a SharePoint rendszergazda csoportnak a portálon belül.

Helyi rendszergazda csoport

A helyi rendszergazda csoport tagjainak joga van végrehajtani minden olyan adminisztratív funkciót a szerveren, ami a helyi szerverre vonatkozik, de csak és kizárólag a helyi szerverre. Ha tartományi rendszergazdák vagyunk, akkor ez a tartományra értendő természetesen.

SharePoint rendszergazda csoport

Minden olyan személy vagy felhasználó, aki tagja a SharePoint rendszergazda csoportnak, képes a központi adminisztrációkat elvégezni anélkül, hogy ő a helyi szerveren (vagy a tartományban) egyébként rendszergazdai funkciókkal rendelkezne. Azaz elválasztható egymástól a SharePoint adminisztrációs jogosultság az általános Windows NT tartomány adminisztrációs folyamataitól. Amennyiben csak a SharePoint adminisztrátor csoportnak vagyunk tagjai, de a Windows NT tartománynak nem vagyunk rendszergazdái, akkor egy „futtatás másként" paranccsal nem tudunk jogosultságot szerezni a szerverünkön. Viszont ha a SharePoint adminisztrációs oldalain tevékenykedünk, akkor képesek vagyunk új helyet, helyeket létrehozni, módosítani a kvótákat, illetve a hely, helyek egyéb paramétereit átállítani. Ha a lokális szerverünk adminisztrátor csoportjában tagok vagyunk, de nem vagyunk tagjai a SharePoint adminisztrátor csoportnak, azzal, hogy a lokális szerver rendszergazdái vagyunk, automatikusan SharePoint rendszergazdákká is válunk egyben. Ez nagyon fontos, mert a Windows rendszergazdák (legyenek helyiek, vagy tartományiak) nem korlátozhatóak jogkörileg az SPS, vagy a WSS tekintetében.

Ellenben fordított helyzetben, azaz amennyiben tagjai vagyunk a SharePoint rendszergazda csoportnak, de nem vagyunk tagjai a helyi (vagy tartományi) rendszergazda csoportnak, úgy nem tudjuk módosítani az Internet Information Services (IIS) adatbázis beállításait, ahogy a helyi szerveren lévő fájlrendszer beállításait sem. Ennek következtében nem tudjuk az alábbi műveleteket végrehajtani az SPS, és/vagy a WSS rendszereken:

- Nem tudjunk kiterjeszteni virtuális szervereinket, azaz nem tudunk legfelsőbb szintű webhelyet létrehozni, illetve megváltoztatni annak beállításait.
- Nem tudjuk visszavonni (uninstall) a SharePoint Services-t.
- Nem tudjuk kezelni az elérési útvonalakat.
- Nem tudjuk megváltoztatni a rendszergazdai csoporttagságot.
- Nem tudjuk konfigurálni az adatbázis beállításainkat, illetve nem tudjuk használni az STSADM.EXE eszközt.

A SharePoint rendszergazda csoport tagjai képesek végrehajtani minden olyan adminisztrációs feladatot, melyek a központi adminisztrációs oldalakon elérhetők.

Ez a kettős rendszergazdai jogkör egy kellemes középmegoldás arra nézve, hogy szerverünket biztonságban tudjuk, és olyan portál rendszergazdákat jelölhessünk ki magunk mellé, akik esetlegesen az alaptechnológiához, mint pl. az SQL szerverhez, vagy a Windows szerverhez nem rendelkeznek megfelelő képesítéssel, tudással (azaz nem tudják az életlenül sem elrontani). Egy ilyen rendszergazda, aki a SharePoint rendszergazda csoport tagja, képes törölni a WSS csoportmunka helyeket, illetve minden egyéb olyan tevékenységet el tud végezni, ami a megfelelő helyek adminisztrálásához feltétlen szükséges, de nem többet.

Rendszergazdák

Mint fentebb már említettem, a rendszergazdák az összes joggal rendelkeznek, melyek a portálhelyekhez, webhelyekhez szükségesek.

Vigyázat! Nincs lehetőségünk módosítani, illetve törölni a gyári rendszergazdai csoport jogosultsági összetételét, viszont adhatunk hozzá és el is vehetünk belőle tagokat. Bárhogy is rendezzük a rendszergazdai csoport tagságát, a rendszergazdai csoportban mindig legalább egy tagnak lenni kell! A portálhely tulajdonosa és az esetlegesen definiált másodlagos tulajdonosa minden esetben a portálhelyhez tartozó rendszergazdai csoportnak a tagja kell, hogy legyen.

Nagyon előnyös az SPS jogosultsági szerkezetében, hogy portálhelyenként mindig definiálva vannak a csoportok különkülön, azaz eltérő lehet portálhelyenként a csoporttagság. Így pl. az egyik helyen lehetünk webtervező, a másikon meg csak olvasó. Az SPS adminisztrációs feladatai minden esetben elvégezhetőek bármely portálhelyen, vagy legfelsőbb szintű webhelyen, amennyiben a rendszergazdai csoport tagjai vagyunk a helyi szerverünkön, vagy esetlegesen a tartományi rendszergazda csoportban. Hisz ezen helyi rendszergazda csoportok, és tartományi rendszergazda csoportok minden esetben kötelezően tagjai az SPS rendszergazda csoportjának ugyebár.

Felhasználók és csoportok

Valahol ott tartunk, hogy a rendszergazdák, illetve a rendszergazdai csoport tagjai minden webhelyen (akár a legfelsőbb szintű portálhelyen) képesek kezelni a felhasználók hozzáférési és csoporttagsági jogait. Így attól függően, hogy melyik felhasználó melyik portálhelyen melyik csoport tagja, ennek megfelelően fér hozzá az adatokhoz. Vagy épp ellenkezőleg: nem fér hozzá azokhoz. A felhasználókat kétféle módon tudjuk hozzáadni a portálrendszerünkhöz, hogy valóban felhasználói legyenek annak. A kétféle modell közül csak az egyiket tudjuk választani:

- „Tartományi felhasználó" módban a felhasználók tagjai lesznek csoportjainknak feltételezve, hogy a tartományi felhasználók már létező felhasználók mondjuk az AD-ben.
- Az „Active Directory (AD) felhasználók készítési" módban a létrehozott felhasználók ugyanúgy tagjai lesznek az általunk kijelölt csoportoknak, de nem szükséges, hogy eredetileg az Active Directory-ban létezzenek. Amennyiben így hozzuk létre a felhasználóinkat, a létrehozáskor automatikusan létrejönnek az Active Directory-ban.

A két mód közül a telepítés folyamán kell választanunk. A későbbiekben nincs lehetőségünk áttérni egyik módról a másikra. Bármelyik módot is választjuk, minden esetben rendelkezésünkre állnak a HTML felületű rendszergazdai oldalak.

Tartományi felhasználó mód

Ha ezt a módot választottuk telepítéskor, úgy a WSS oldalain, illetve szervezeti felépítésében a felhasználók minden esetben Windows tartományi (vagy helyi) felhasználók is egyben. Tulajdonképpen így a meglévő felhasználókat tudjuk hozzáadni SharePoint alapú csoportjainkhoz vagy csoportmunka-helyeinkhez.

Amennyiben így indulunk neki rendszerünknek, nincs lehetőségünk az AcitiveDirectory-ban létrehozni felhasználókat SharePoint portálunkon keresztül. Ez a mód egyébként az alapértelmezett működési módja a WSS-nek, és az esetek 99 százalékában valóban ezt a módot használjuk.

AD felhasználó készítési mód

Amennyiben ezt az üzemmódot használjuk, úgy a WSS portálunkra való ránézéskor automatikusan készül el az AD-ben a felhasználó. Ha ezt a módot választjuk a portálunk üzemeltetéséhez, akkor nem tudjuk használni meglévő tartományi felhasználóinkat automatikusan.

Nagyon fontos, hogy az SPS ezt a módot nem alkalmazza, csak a WSS. Így nekünk sincs lehetőségünk az SPS esetében ezt a módot választani telepítéskor. Általában ezért az apróságért nem találkozunk ezzel a lehetőséggel az SPS telepítésekor.

Amikor ezt az üzemmódot használjuk a WSS esetében az AD felhasználók készítéséhez, a belépéskor meg kell adnunk a tartományban használandó nevet, a hozzá való e-mail címet, illetve csoporttagságot. A csoporttagság ilyen értelemben a SharePoint csoportokra vonatkozik, nem pedig a tartományi (helyi) csoportokra.

A WSS ellenőrzi, hogy az ActiveDirectory-ban van-e már ilyen nevű felhasználó (mármint amit épp megadunk), illetve e-mail cím, stb. Amennyiben nincs, úgy engedélyezi a létrehozást az AD-ben, ellenkező esetben figyelmeztetést ad, hogy a megadott paraméterekkel nem lehet létrehozni a belépni szándékozó felhasználót. Ha minden adatot sikeresen megadtunk, úgy a WSS üzenetben küldi meg a belépési jelszót a felhasználónak a megadott e-mail címre.

Amennyiben ezt a ritkán használt ActiveDirectory felhasználó készítési módot használjuk, a rendszergazdai feladatokat nem tudjuk ellátni a HTML adminisztrációs oldalakon. Pl. nem tudunk készíteni legfelsőbb szintű webhelyet, nem tudjuk engedélyezni portálhelyek létrehozását, és nem tudunk felhasználókat hozzáadni a központi felügyeleti lapokhoz.

Központi adminisztrációs oldalak

Ha rendszergazdái vagyunk a helyi szerverünknek, vagy tagjai vagyunk a SharePoint rendszergazdai csoportnak, akkor lehetőségünk van minden adminisztrációs jogot, és azok módosítását megtennünk itt a portálhelyeinkre.

A központi adminisztrációs oldalakról tudunk kapcsolódni a portálhelyek saját adminisztrációs oldalaihoz. Valahol itt kezdtük a cikket is. Körbeértünk ☺. Ezeken az oldalakon tudjuk az adott portálhelyhez kapcsolódó jogosultság-módosításokat elvégezni, illetve felhasználókat visszavonni, hozzáadni, módosítani a taglistákat, változtatni a tulajdonosokat.

Ugyanezen oldalakon módosíthatjuk a portálhelyhez tartozó leírásokat, elérési útvonalakat, illetve specifikus, speciális beállításokat. Továbbá meg tudjuk nézni, illetve ki tudjuk listázni mindazon felhasználókat és csoportokat, amelyek jogosultsággal rendelkeznek az oldalainkhoz. Az igények alapján az új felhasználókat a csoportokhoz, illetve a jogosultsági listákhoz tudjuk rendelni. Persze törölni is tudjuk a felhasználókat és csoportokat, valamint visszautasítani az igényléseket.

Amikor hozzáadunk egy új felhasználót egy csoporthoz, vagy portálhelyhez, lehetőségünk van arra, hogy az SPS szerverünk e-mailben küldjön értesítést a felhasználónak arról, hogy ténylegesen hozzáfér a portálhelyhez. Módunkban áll az alapüzenet e-mail formáját megváltoztatni, és/vagy a kérelmekre válaszolni.

Ha már említettem az e-mail alapú üzenetküldést, akkor végezetül, de nem utolsósorban nézzük meg ezeket az értesítéseket, riasztásokat.

Értesítések kezelése

Az értesítések kezelésével foglalkozó oldalra úgy jutunk el, hogy a „Webhely beállításai" oldalon, az „Általános beállítások" résznél kattintunk egy nagyot az „Értesítési beállítások kezelése" hivatkozásra. Pont a biztonsággal foglalkozó link alatt. Az elénk táruló oldalon két témában gyakorolhatjuk rendszergazdai jogainkat:

Értesítések lehetőségei

Törölhetjük, vagy visszavonhatjuk a már nem létező, vagy üzemen kívül helyezett felhasználók értesítéseit. Visszavonhatjuk az összes értesítést az őket tartalmazó adattárból a portálhelyünk összes felhasználójára nézve. Készíthetünk kvóta limiteket arra nézve, hogy hány értesítést állíthatnak be a felhasználók összességében portálhelyünkön. Azt is beállíthatjuk, hogy hány értesítést állíthat be egy felhasználó magának, ahogy azt is, hogy az értesítések hány helyre mehetnek maximálisan üzenetként. A portálhely automatikusan próbálja optimalizálni az értesítések összességét feladatuktól függően. Így amennyiben valamelyik értesítés értelmét veszíti, illetve hatályon kívül esik, úgy azokat deaktiválja, visszavonja a rendszerből.

E-mail értesítések kezelése

Konfigurálhatjuk az SMTP szervert és engedélyezhetjük a felhasználók e-mail-ben történő értesítését. Visszavonhatjuk az összes várakozó e-mail értesítést illetve a várakozó üres üzeneteket. Módosíthatjuk a felhasználók profiljaiban azt a beállítást, ami az alapértelmezett e-mail címet tárolja. Módosítani tudjuk az üzenet formátumát, kinézetét, illetőleg azt, hogy milyen szövegeket tartalmazzon az üzenet.
Ugye az is jól látszik, hogy a csoport és a rendszergazdai jogok milyen finom játékokra adnak lehetőséget az SPS és a WSS felügyeletére, elérésére nézve. Persze ami a cikkben szerepel, az csak egy része az információknak, a lehetőségeknek. Nem említettem például a felhasználók és az AD kapcsolatát, illetve a felhasználói profilok kezelését.

Aki ennél többet szeretne tudni a témáról, az bátran keresse fel az IQSOFT – John Bryce Oktatóközpont munkatársait!

Farkas Viktor
IQSOFT - John Bryce Oktatóközpont
farkas_v@hotmail.com
MCSE, MCT, HP-ASE

Dr. Watson

Napi feladatok az SQL Server 2000-rel

Sok cikk jelent már meg SQL Server témakörben, de eddig csupán fejlesztői szemszögből vizsgáltuk az adatbázis-kezelést. A többség számára ez a megközelítés teljesen haszontalan. Ők azok az adatbázis-rendszergazdák, akik egy módosíthatatlan késztermék üzemeltetéséért felelősek, abból kell kihozniuk a maximális teljesítményt, azt kell karbantartaniuk stb. Nekik szól ez a cikk.

Az SQL-adatbázis egy élőlény: megszületik, felnövekszik, általában hosszú életút után nyugdíjba vonul, majd szegényen és elfeledve hal meg valahol egy koszos PC-n a sarokban. Élete során adatokkal táplálkozik, meghízik, ettől ellustul, ekkor alakformáló szalonba küldjük, amitől megfiatalodik, ismét fürge lesz. Az eddigiek is szépen mutatják, hogy igencsak van olyan jellegű teendő vele, ami nem igényli az élőlény kibelezését, szívátültetést és hasonló műtéteket. Ezek a véres dolgok a fejlesztés témakörébe tartoznak.

Foglaljuk röviden össze, milyen feladatok várnak tehát azokra, akik egy SQL-alapú alkalmazás boldog tulajdonosainak tekinthetik magukat:

- Üres adatbázis létrehozása, telepítőscript futtatása
- Felhasználók létrehozása/beengedése adatbázisokba
- Az adatbázis helyreállítási üzemmódjának beállítása (recovery model)
- Bizonyos adatok publikálása a webre
- Időzített mentések, indexkarbantartás és hasonló feladatok (jobok) készítése
- Adatpumpa (export-import) különböző forrásokból, -ba
- Kritikus hibákról automatikus értesítés kérése emailben

Ezek nem mindegyikét fogjuk most kielemezni, mert például a legutolsó pont, az emailes értesítés megvalósítása nem két, hanem huszonkét lépésből áll. E nélkül is van munka elég!

Milyen feladatok várnak az SQL-alapú alkalmazások tulajdonosaira?

Adatbázis létrehozása

Tegyük fel, hogy megvásároltuk az Elba Soft által fejlesztett csodálatos SQL-alapú ügyviteli rendszert. A fejlesztő érti a dolgát, tehát egy A4-es papírlapon kapjuk meg a kiszolgálóoldali telepítés lépéseit (ezt hívják integrált telepítésnek, a papír is bele van integrálva). Ezt elolvasva kitűnik, hogy a telepit.sql fájlt kell lefuttatnunk, és ez létrehozza a táblákat, sőt, alapadatokkal is feltölti azokat. Igen ám, de hogyan?

Az sql kiterjesztésű fájlokat például az SQL Server menücsoportjában található Query Analyzer segítségével futtathatjuk le oly módon, hogy – megfelelő bejelentkezés után – megnyitjuk a fájlt, majd megnyomjuk a kis zöld nyilacskás ikont (execute).

Ennek a lépésnek két kimenetele lehetséges:

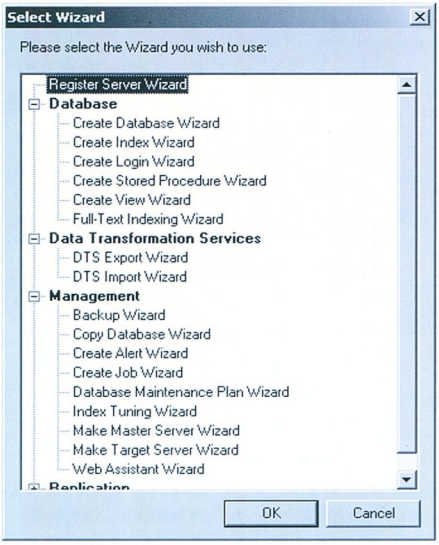
- Volt a scriptben CREATE DATABASE utasítás, tehát létrejött az adatbázis, és abba kerültek a táblák.
- Nem volt benne CREATE DATABASE, így az összes tábla és alapadat Master adatbázisban kötött ki. Ez igen örömteli esemény, most több órányi kutatómunkánk van, hogy hogyan is kell ezt meg nem történtté tenni.

Elba Soft a termék... De most mit kellene tennünk? A telepítő script egy ügyes hibaüzenetet produkált:

```
Server: Msg 911, Level 16, State 1, Line 1
Could not locate entry in sysdatabases for
↳ database 'Elba'. No entry found with that name.
↳ Make sure that the name is entered correctly.
```

Ilyenkor következik az adatbázis létrehozása – kézzel. Az SQL Server esetében általában minden művelet végrehajtására kétféle módszer áll rendelkezése. Az egyik a grafikus mód, az Enterprise Manager segítségével. A másik a megfelelő SQL-parancs kiadása a Query Analyzerrel. Nézzük meg először a grafikus módszert!

Az Enterprise Managerben ássunk le a szerverig, jelöljük ki, majd keressük meg az eszközsoron a varázspálca ikont. Jó hírem van a rendszergazdák számára: gyakorlatilag minden értelmes feladatra van varázsló!



SQL-varázslók az Enterprise Managerben

Válasszuk ki a „Create Database Wizardot", és indítsuk el! A második képernyőn adhatjuk meg a létrehozandó adatbázis nevét (Elba) – és másra nincs is talán szükségünk. A finnyásabbak külön partícióra tehetik az adatbázis- és a naplófájlt, de mi most Next-Next-Next-Finish módszertan szerint dolgozzunk. (Mit lehetett volna még megadni? A fájlok nevét, méretét és növekedési paramétereit.)
Ha magyar SBS-sel van dolgunk, az új adatbázis automatikusan magyar nyelvi beállításokkal születik (ékezetérzékeny, kis-nagybetű-érzéketlen).
Ugyanez Transact SQL-utasítással:

```
CREATE DATABASE ELBA
```

Ha angol SBS-t használunk, az adatbázis is ilyen nyelvű lesz, viszont szerencsére utólag ezt át lehet állítani magyarra:

```
ALTER DATABASE ELBA COLLATE Hungarian_CI_AS
```

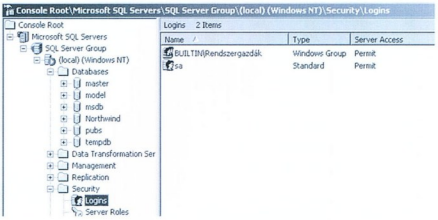
(Megjegyzem: ez esetben a varázsló nélküli adatbázis-létrehozás jobb lett volna, mert egy lépésben meg lehetett volna adni a nyelvi opciókat.)
Ez után már sikeres lehet az sql kiterjesztésű fájl lefuttatása.

Felhasználók létrehozása/ beengedése adatbázisokba

A következő lépés az lesz, hogy – akár le van írva az A4-es lapon, akár nincs – felhasználókat kell létrehozni az SQL Serveren, és el kell látni őket a megfelelő jogosultsággal. (Előfordulhat, hogy a telepítőscript ezt is megcsinálja, de ha nem, íme a kézi módszer.)
Mielőtt nekiesnénk a feladatnak, röviden meg kell ismerkednünk az SQL Server jogosultsági modelljével. Két tényezőt kell figyelembe venni:

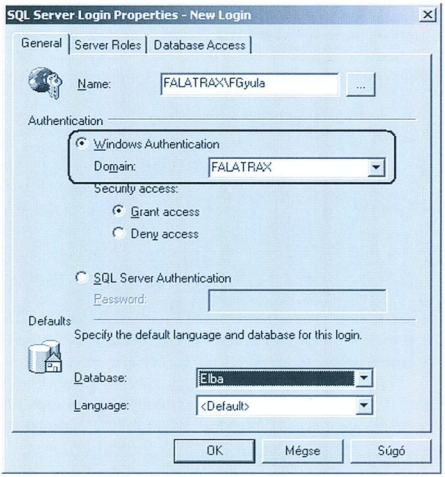
- Az SQL Server teljesen különválasztja a felhasználó azonosítását (autentikáció) a jogosultság megszerzésétől (autorizáció). Az első lépés csupán azonosítja a felhasználót (tudni fogjuk, hogy ő biztosan Fontos Gyula), de ezzel még jogosultsághoz nem jut. Tehát mindössze az előszobáig jut, nem ugorhat fejest az ágyba.
- Az SQL Server kétféle autentikációs eljárást tartalmaz: önmaga is képes a nevek és jelszavak ellenőrzésére (SQL Authentication), vagy kész tényként elfogadhatja a Windows által elvégzett felhasználóazonosítás eredményét (Windows Authentication). Ez utóbbi esetben nem kell újra begépelnünk jelszavunkat, hanem ripszropsz beléphetünk az SQL Serverbe.

Telepítés után mindössze a Windows autentikáció van bekapcsolva, ezért erre tekintettel kell lennünk a felhasználók létrehozásánál: felesleges SQL-felhasználókat *létrehozni,* mert úgysem jelentkezhetnek be mindaddig, amíg az SQL Serveren nem engedélyezzük az SQL-autentikációt (nem fogjuk engedélyezni). Mi marad ebben az esetben? Nem kell felhasználókat létrehozni, csupán a meglévő Windows-fiókokat (vagy csoportokat) kell *„beengednünk"* az adatbázis-kezelőbe (az előszobába). Tegyük fel, hogy Fontos Gyulát kell beengednünk. Az Enterprise Managerrel nyissuk ki a bal oldali fában a Security ágat, majd kattintsunk a Logins elemre.
A jobb oldalon mindössze két „felhasználó" árválkodik, amelyek közül az egyik nem is felhasználó, hanem Windows-csoport (BUILTIN\Rendszergazdák), a másikkal viszont (SA) nem lehet bejelentkezni, mert ez egy sima SQL-felhasználó, ami csak akkor használható, ha az SQL-autentikáció engedélyezve van (nincs). Amit itt látunk, azt jelenti, hogy aki tagja a Windows-féle Rendszergazdák csoportnak, az bejöhet az előszobába. Sőt, megnyitva ezt a sort, kiderülne, hogy ezek a fickók egyben az SQL Server teljhatalmú rendszergazdái is, mivel ehhez a csoporthoz hozzá van rendelve a System Administrators SQL-szerep.



Beépített felhasználók az SQL Serverben

Máris látunk egy utat Gyula beengedésére: ha beledobjuk a Rendszergazdák Windows-csoportba, automatikusan SQL-adminná válik! Csak ezt ne!
Engedjük be személyesen Gyulát az SQL Serverbe, és adjunk neki jogot az Elba adatbázisra! Jobb klikk a Logins ágon, New Login...



Windows-felhasználó beengedése az SQL Serverbe

A felső mező melletti gombbal válasszuk ki Gyulát, az autentikáció maradjon Windows (ahogy a keret mutatja), és ha még ebben a lépésben be szeretnénk engedni az Elba-adatbázisba, kattintsunk a Database Access fülön, és ott pipáljuk be, hogy hozzáférhet a megfelelő adatbázishoz. (A fenti ábrán látható Database mező, ahol szintén kiválasztottam az Elbát, *nem* ad jogosultságot!)
Ezzel Gyula hozzáfér az Elba adatbázishoz, mégpedig olyan jogosultsággal, ami ott a Public csoportnak jár. Azaz semmivel. A jogosultságállítás (autorizáció) adatbázisonként külön-külön történik. Nyissuk ki a baloldali fában az Elba adatbázis, kattintsunk a Users elemre. A jobb oldalon két felhasználó látható: dbo és FGyula. Nyissuk meg Gyulát, és a teljesen egyértelmű felületen adjuk meg neki azokon a táblákon a jogosultságokat, amit az A4-es említ.
Ugyanez Transact SQL-utasítással:

```
EXEC sp_grantlogin 'FALATRAX\FGyula'
USE Elba
EXEC sp_grantlogin 'FALATRAX\FGyula'
GRANT ALL on valamelyik_tábla to [FALATRAX\FGyula]
```

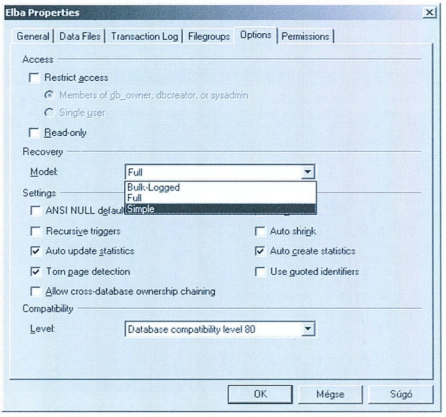
Természetesen a GRANT ALL helyett finomabb jogosultságszabályozás is elképzelhető, hiszen a jogosultságokat műveletenként állíthatjuk be (például GRANT SELECT..., DENY INSERT... stb.)

Az adatbázis helyreállítási üzemmódjának beállítása (recovery model)

SBS-környezetben nem szokás különösebb figyelmet fordítani a mentési stratégiára. Sokan így gondolkodnak: „ha mindennap csinálunk egy teljes mentést, abból nem lehet baj!" Pedig mekkorát tévednek!
A teljes mentés ugyanis nem szabadítja fel a tranzakciónapló inaktívvá vált részét (részletes magyarázat SQL-tanfolyamon), így az gyűlik, gyűlik, gyűlik addig, amíg meg nem zabálja a merevlemez teljes területét. Ha ez bekövetkezik, minden megáll.
Két módszer áll rendelkezésünkre a tranzakciónapló dagadásának megakadályozására:

- Néha csináljunk tranzakciónapló mentést, akár értelmesnek tekintjük ezt a lépést, akár nem. Ez ugyanis törli a napló inaktív részét.
- Ha viszont nem mentjük a tranzakciónaplót, szóljunk az SQL Servernek, hogy ne örizgesse a végtelenségig, hanem időnként törölje.

Ez utóbbi módszert fogjuk most kipróbálni egérrel és scripttel. Nyissuk meg az adatbázis tulajdonságlapját (jobb klikk Elbán, tulajdonságok), menjünk az Options fülre, és állítsuk be a Recovery Mode mezőt Simple-re.



Simple Recovery beállítása az adatbázison

Ugyanez Transact SQL-utasítással:

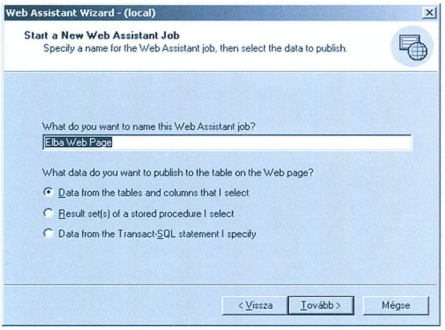
```
sp_dboption 'Elba', 'trunc. Log on chkpt.', true
```

Ugye, milyen egyszerű volt? Bármelyikünk kitalálta volna, pusztán az opció nevéből (Simple) :-)

Adatok publikálása a webre

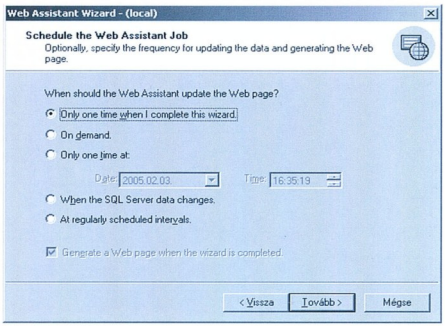
Ma már gyakori igény, hogy egy vállalat a termékeiről naprakész információt jelenítsen meg a weblapján. Kisvállalatoknál ez az igény azzal egészül ki, hogy lehetőleg ingyen, és programozás nélkül oldjuk meg a feladatot. Erre teszünk most kísérletet a Web Assistant Wizard segítségével.
Az eszközről annyit kell tudni, hogy statikus HTML-weblapokat generál ugyan, de többféle módon képes ezeket frissíteni. Ha a naprakészség a fontos – az adatok változását egy triggerrel figyelve – akár minden egyes adatmódosítás hatására képes újragenerálni a lapot. Ha azonban a módosítások száma ezt a módszert már gazdaságtalanná teszi (például másodpercenként száz módosítás történik), időzített módon is képes ugyanerre.
A legfontosabb tudnivaló azonban nem ez, hanem hogy hogyan képes az eszköz olyan weblapokat faragni, amelyek beilleszkednek a cég webhelyén kialakított sémába. Ennek az a titka, hogy a varázsló egy előre elkészített sablonba „bo-

rítja" az adatokat. Nincs más dolgunk, mint egy gyönyörű, céglogós-menüs sablon-lap megfelelő helyére beírni az <%insert_data_here%> címkét, és a varázsló ide fogja beszúrni az adatokat tartalmazó táblázatot.
Ha magát a táblázatot, annak cellát, sorait is „díszíteni" szeretnénk, tegyünk egy üres táblázatot a sablonba, és minden mezőjébe be kell írni az <%insert_data_here%> kulcsszót. Az ismétlendő minta kezdetét (a sor elejét) <%begindetail%>, a végén <%enddetail%> jelzéssel adhatjuk meg. A varázsló innentől ki fogja találni a gondolatainkat.
Most nézzük végig a varázsló által feltett kérdéseket! (Maga a Web Assistant Wizard az Enterprise Manager varázspálcás ikonja mögül csalogatható elő.)
Elsőként meg kell adnunk a publikálandó adathalmazt, ami lehet egy egész tábla, egy tárolt eljárás vagy akár egy általunk megfelelően megfogalmazott SELECT-utasítás eredményhalmaza.



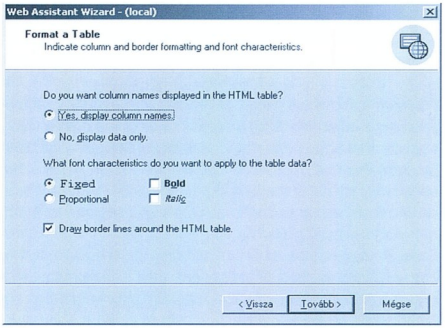
Mi lesz a webes publikálás adatforrása? Tábla? Tárolt eljárás? Lekérdezés?

Az egyszerűség kedvéért most legyen egy tábla, amiből a következő lapon kiválaszthatjuk a megfelelő mezőket. Ez a módszer is lehetővé teszi egyébként meghatározott sorok kiválasztását, így általános esetben nincs szükség saját SQL-utasítás kifaragására. Miután ügyesen kiválasztottuk a kívánatos mezőket, lássuk az „időzítési" beállításokat! Az idézőjel azért indokolt, mert az egyik beállítás (When the SQL Server data changes) valójában nem időzítést jelent, hanem ebben az esetben egy olyan trigger kerül a kiválasztott táblára, ami minden sikeres tranzakció után újragenerálja a HTML-lapot.
Triggerelt esetben ki kell még választanunk, hogy mely mezők változását vegye figyelembe, egyébként a varázsló később azonos lapokkal folytatódik.
Ha szeretnénk, hogy a kész weblap minden teketóriázás, későbbi bonyolult művelet nélkül azonnal megjelenjen a weben, változtassuk meg az alapértelmezett útvonalat, és mentsük inkább a C:\Inetpub\wwwroot könyvtárba. Akár ez lehet a főlap, ha default.htm nevet adunk neki. Ha más nevet választunk, természetesen nekünk kell gondoskodnunk majd arról, hogy a céges főlapról egy linkkel el lehessen érni ezt a lapot is.
A következő kérdés azt feszegeti, vajon a weblapot a varázslónak kell összetákolnia, vagy van egy sablonunk, benne a már említett <%insert_data_here%> jelöléssel. Az egyszerűség kedvéért most az első eseten megyünk végig, a formázást is a varázslóra bízzuk.



Mikor készüljön el a weblap? Időzítve? Triggerelve?

Ekkor meg kell adnunk a weblap és a táblázat címét, valamint hogy tegyen-e a lapra dátumot, majd következik a formázás:



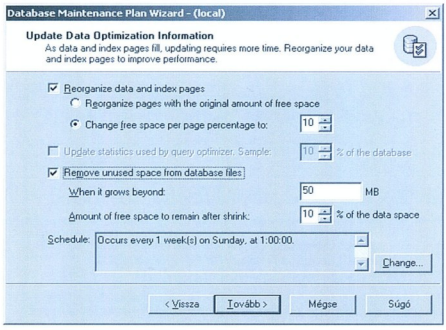
A Web Assistant formázási lehetőségei

Az egyszerűség gyönyörködtet. A következő lapon beállíthatjuk, hogy az összes sort látni szeretnénk-e, vagy csak az első x darabot (gondoljuk végig, mekkora terhet jelent mind az SQL Server, mind a potenciális látogatók számára egy tízezer soros HTML-lap!), illetve hogy kérünk-e lapozást – mondjuk 20 soronként. És ezzel végeztünk is. Az utolsó lapon még lehetőségünk van sql fájlba menteni az egész Web Assistant Jobot, hátha később scripttel szeretnénk újabb lapokat generáltatni. (Az sp_makewebtask tárolt eljárást kell meghívni egy jó tucat paraméterrel, a varázsló is ezt teszi egyébként.)

Időzített mentések, index-karbantartás és hasonló feladatok (jobok) készítése

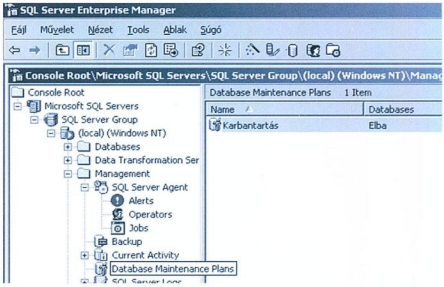
Még egy fontos feladatunk van, különböző karbantartási műveleteket kellene időnként futtatni. Az SQL Server adatbázisai meghálálják a törődést. A töredezettség-mentesítés és az indexek statisztikáinak frissítése érezhetően fürgébbé tehet egy ellustult adatbázist, a mentések pedig egyébként is létfontosságúak.
Ezekre a tipikus feladatokra is van külön eszköz, a neve: Database Maintenance Plan Wizard. (Ott találjuk, ahol a többi varázslót.) Ha csak egyszerűen végighajtjuk Next-Next-Next-

Finish-sel, pusztán egy időzített teljes mentést alakítunk ki, érdemes figyelmesen végigolvasni és kiválasztani a további lehetőségeket is.
Elindítása után ki kell választanunk, melyik adatbázisokra készítjük a karbantartási feladatot. (Természetesen az Elba adatbázisra.)
Mindjárt a következő lap érdekes beállítási lehetőségekkel szolgál. Itt találjuk ugyanis a töredezettségmentesítést, csak éppen nem ez a neve, hanem „Reorganize data and index pages". Ez nemcsak átrendezi a rekordokat, hanem arra is képes, hogy a helyfoglalási egységként használt 8 kilobájtos úgynevezett lapokon (page) bizonyos százaléknyi üres helyet képez, hogy ha a sorrendnek megfelelő további rekordok jönnek létre, azoknak legyen fenntartva hely, vagyis a töredezettség minél később következzen be ismét. A töredezettségmentesítés „mellékhatása", hogy az újraépülő indexeknek felfrissül a statisztikája is, így ha ezt kiválasztjuk, a második opció („Update statistics...") beszürkül.
De mire is jó az „Update statistics..." lehetőség? Abban segít, hogy az SQL Server minél pontosabban meg tudja határozni egy lekérdezésről *előre*, még annak konkrét lefuttatása *előtt*, hogy hány sort fog visszaadni. Ez lehetővé teszi, hogy mindig a megfelelő indexet választhassa ki a lekérdezés felgyorsítására. Ez a témakör messzire vezet, az SQL-guruk földjére, most elégedjünk meg ennyivel, és higgyük is el minden további magyarázat nélkül, hogy ez igaz, így működik és szükség van rá. (Az SQL Query Optimizer mindig dinamikusan, az adatok és a lekérdezés paramétereinek függvényében választja ki a megfelelő indexeket és illesztési (join) stratégiát.)
A harmadik opcióval a varázslat a töredezettségmentesítés hatására felszabaduló üres helyet adja vissza az operációs rendszernek, ha megkérjük rá.
Alapértelmezésben ezek a feladatok minden vasárnap éjjel 1 órakor hajtódnak végre.



Töredezettségmentesítés minden vasárnap éjjel egykor!

Haladjunk tovább! A következő lapon az adatbázis integritásának (hibátlanságának) ellenőrzése kérhető. Mi az, ami elromolhat? Elvileg semmi, és több éves tapasztalataink szerint is egyáltalán semmi, de ha mégis, akkor például a 8k-s lapok közötti láncolt lista sérülhet meg. Az ilyesmit a redundáns adatok segítségével javítani is tudja az SQL Server, hát adjunk esély neki erre, pipa be! Ez a feladat alapértelmezésben az előzőekben említett töredezettség-mentesítés előtt egy órával fog lefutni, tehát vasárnap éjfélkor, hogy már hibátlan rekordokat tologasson ide-oda a rendezgetés során.
A következő lapon nem kell tennünk semmit, az egy teljes mentést fog beütemezni akkorra, amikor az előző folyamatok már remélhetőleg véget értek, tehát vasárnap éjjel 2-re. Egy Next után beállíthatjuk, hogy maximálisan hány mentést őrizzen meg és haladhatunk tovább.
A következő lap a tranzakciós napló mentéséé, pipáljuk be, és figyeljük meg, hogy ez – vasárnap kivételével – a hét minden napján le fog futni, mégpedig ismét éjfélkor. Itt is be lehet állítani, hogy hány mentést őrizzen meg az utókor számára. Haladjunk tovább.
Már csak a feladatok futásáról készített jelentést kell összekattintanunk (fájlba, MSDB-be, email címre lehet jelentést továbbítani) és készen is vagyunk. A Web Assistant Jobokkal ellentétben a Maintenance Plan utólag is könnyedén megváltoztatható, mert ügyesen feltüntették az Enterprise Manager konzolfájában, itt:

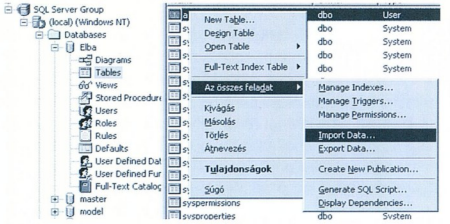


A karbantartási varázslás eredménye

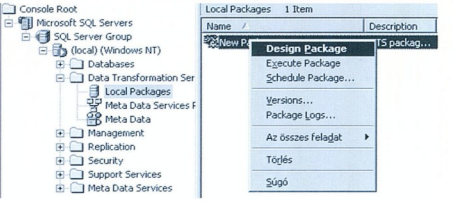
Egyébként maguk a feladatok négy különböző tartalmú és ütemezésű SQL Agent jobban futnak, és az xp_sqlmaint külső tárolt eljárást hívogatják ezernyi paraméterrel.

Adatpumpa (export-import) különböző forrásokból, -ba

Utolsóként a sorban még egy nagyon fontos feladat: adatok átvitele egyik adattárolóból a másikba. Ezt a munkát az SQL Server mellé csomagolt, azzal szorosan együttműködő, de mégiscsak önálló program, a Data Transformation Services (DTS) végzi. Onnan lehet tudni, hogy független eszközről van szó, hogy zokszó nélkül visz át adatot akárhonnan akárhová (mondjuk Paradoxból DB2-be) anélkül, hogy akár a forrás, akár a cél kötelezően SQL Server lenne. Nem finnyás: amihez van OLEDB provider, az jó neki.
A DTS teljes körű kihasználása ismét nagyon messzire vezet, ugyanis egy adatfolyam-vezérelt igencsak alternatív programozási eszközről van szó. De szerencsére az egyszerűbb export-import feladatok végrehajtására itt is van – na mi? – varázsló!
Válasszunk ki az Enterprise Managerben egy tetszőleges táblát, és suhintsunk oda neki a jobb egérgombbal! A gyorsmenüben találjuk az Import Data... és Export Data. lehetőségeket:



Adatátvitel DTS-sel

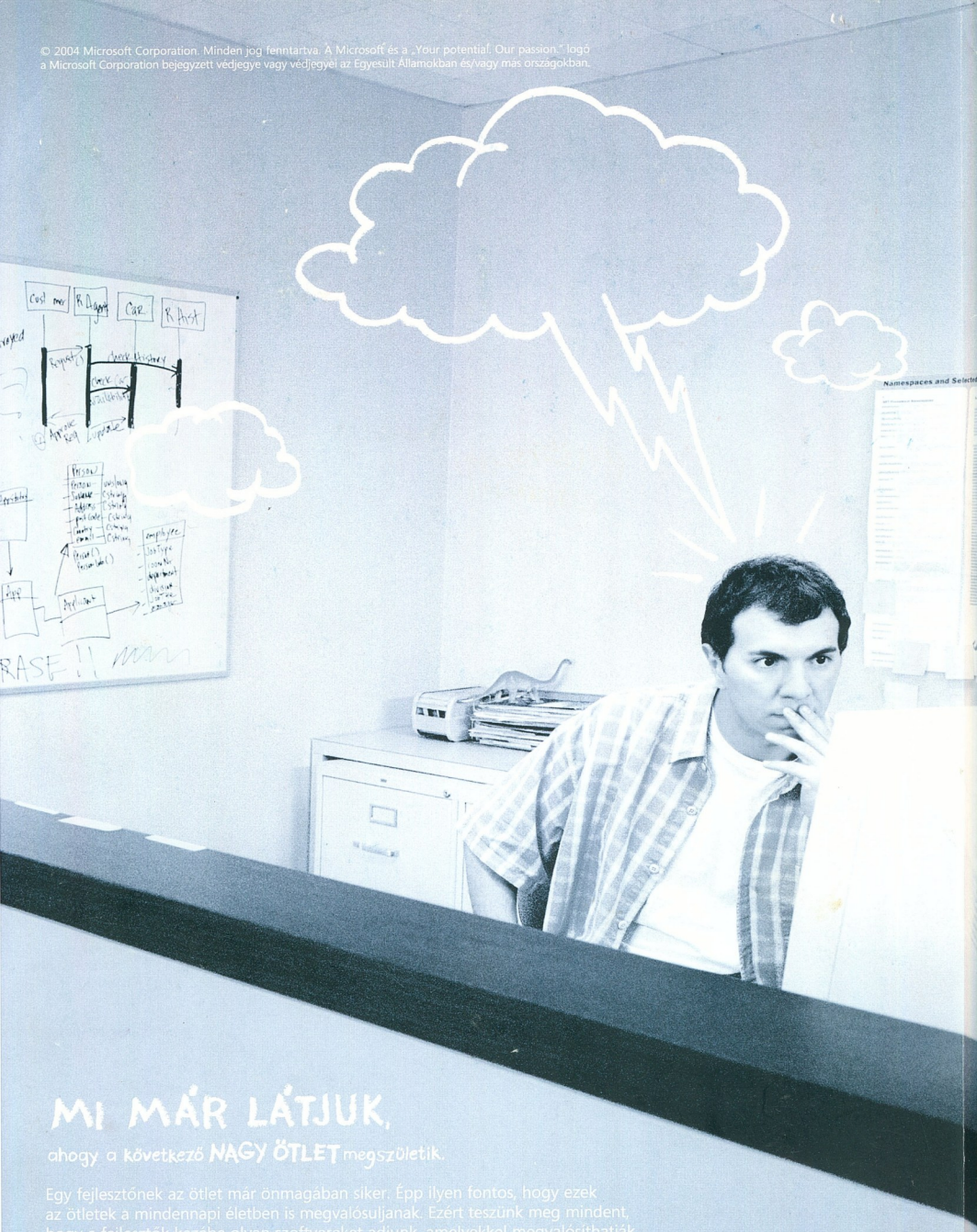


A varázslóval készített DTS-csomag

Terjedelmi korlátok miatt ezt az eléggé egyértelmű varázslót nem nézzük végig tételesen, csupán a végén lévő mentési lehetőséget emelném ki: itt derül ki, hogy egy DTS-csomagot hoztunk létre, amit nemcsak azonnal lefuttatni, időzíteni, hanem elmenteni is lehet, akár további módosítás, akár későbbi futtatás céljából. A csomag itt lesz fellelhető a későbbiekben:

Érdemes megtekinteni a csomagot a DTS-szerkesztőben, hogy lássuk, a lehetőségeknek csupán 1%-ánál járunk ebben a szép témakörben.

Fóti Marcell
MCSE, MCSA, MCDBA, MCT, MZ/X
marcellf@netacademia.net



MI MÁR LÁTJUK,

ahogy a következő **NAGY ÖTLET** megszületik.

Egy fejlesztőnek az ötlet már önmagában siker. Épp ilyen fontos, hogy ezek az ötletek a mindennapi életben is megvalósuljanak. Ezért teszünk meg mindent, hogy a fejlesztők kezébe olyan szoftvereket adjunk, amelyekkel megvalósíthatják elképzeléseiket. Az ötleteket, amelyekkel később mindenki nyer.
Neked lehetőség. Nekünk kihívás.™

Your potential. Our passion.™

Microsoft