ISSN 1586185 05
9 771586 518005
V./5. szám
2004. december

A 2005-ös év újdonságai

Microsoft GOLD CERTIFIED Partner | Learning Solutions

A SZÁMALK Továbbképzés a 2005-ös évben is számos új képzéssel, szolgáltatással és versenyképes árakkal várja tisztelt ügyfeleit. A teljes kínálatról, oktatási konstrukciókról további információkat weboldalunkon találhat, vagy, kérjük, keresse szervezőinket!

Microsoft CRM képzések és oktatási csomagok

SMS 2003, MOM 2005, SharePoint 2003 tanfolyamok

SQL Server 2005 és ISA Server 2004 tanfolyamok

Kedvezményes Windows 2003 rendszeradminisztrátor és rendszermérnök (MCSA/MCSE) képzések

Megújult .NET fejlesztői képzések (Visual Studio 2005)

A SZÁMALK Továbbképzés a Microsoft első magyarországi aranyfokozatú partnere az oktatási megoldásokban

Minden tisztelt partnerünknek ezúton is kellemes karácsonyt és ereményekben gazdag, boldog új esztendőt kívánunk! Reméljük, a jövő évben is igénybe veszik szolgáltatásainkat.

SZÁMALK

További információ: Simon Ferenc, tel.: 203-0304/3050, e-mail: simonf@szamalk.hu

www.szamalk.hu/tisza

Microsoft TechNet

TechNet Magazin
V. évfolyam, 5. szám
2004. december

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)

Lapmenedzser:
Kolma Kornél
(Microsoft Magyarország)

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

Borítógrafika:
Molnár Ferenc

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

Világszerte gyorsan hódít az SP2

A Microsoft kimutatásai alapján Magyarországon is dinamikusan növekszik az SP2-t letöltő és telepítő felhasználók száma

A Microsoft 2004 augusztusában tette elérhetővé a Windows XP Service Pack 2 javítócsomagot. Az ingyenes szervizcsomag a Microsoft legújabb biztonsági frissítéseit és innovációit tartalmazza, illetve olyan új biztonsági funkciókkal bővíti a rendszert, amelyek segítségével jobban megóvható a számítógép a hackerekkel, a vírusokkal és az egyéb biztonsági kockázatokkal szemben. A szervizcsomag nyilvánosságra hozatalával egy időben a Microsoft megkezdte a honosításokat, így a Microsoft Magyarország már 2004 szeptember 10-én bejelenthette, hogy a Windows XP Service Pack 2 magyar nyelvű verziója gyártásra kész állapotba került.

A Microsoft kimutatása alapján Magyarországon november elejéig mintegy 200 ezren töltötték le és több mint 90 ezren telepítették az SP2-t, ezen kívül mintegy 50 ezer példányban, ingyenes CD formájában is eljutott a felhasználókhoz, újságok mellékleteként. Az SP2 hazai kedveltségének biztos növekedését jelzi, hogy október második felében megugrott a regisztrációk száma, a felmérés idejének utolsó két hete alatt regisztrálták az összes addigi letöltés 40 százalékát. A terjedés sebességét tekintve hazánk a középeurópai országok élvonalába tartozik.

Az SP2 kedveltsége világviszonylatban is töretlen: az augusztusi bevezetés óta eltelt két és fél hónap alatt elérte a 100 milliót azon felhasználók száma, akik számítógépük biztonságának növelése érdekében letöltötték a Microsoft javítócsomagját.

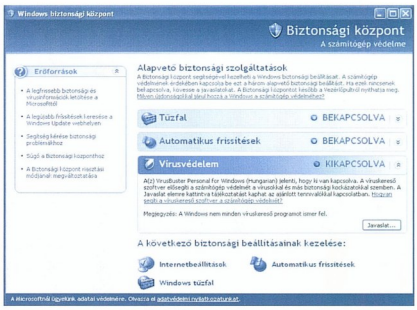
A Technet e számában folytatjuk az SP2 szervizcsomag által nyújtott új lehetőségek bemutatását.

A Windows XP Service Pack 2 javítócsomag a

http://www.microsoft.com/hun/winxpsp2

weboldalon vagy a Windows XP automatikus frissítésszolgáltatásán keresztül érhető el, ezen kívül kiskereskedelmi terjesztéssel, újságok mellékleteként, ingyenes CD-ken, valamint új számítógépekre előtelepítve jut el a számítógépgyártókhoz, a vállalati ügyfelekhez és az egyéni felhasználókhoz. Akik esetleg nem rendelkeznek megfelelő interneteléréssel, telefonon is megrendelhetik az SP2-t a Microsoft Magyarország ügyfélszolgálatától.

Telefon: 06-1-267-4636



Tartalom

A Windows Media DRM Platform

...AVAGY DIGITÁLIS TARTALOMVÉDELEM FELSŐFOKON

A digitális kor beköszöntével egyre nagyobb az igény egy olyan, átfogó keretrendszerre, amellyel a tartalomszolgáltatók biztosíthatják tartalmuk védelmét és ezen biztonság tudatában olyan multimédiás anyagokat tehetnek közzé, amelyeket a védelem hiánya miatt eddig nem mertek.

SUS és WUS

A KÖZTES PATCH MANAGEMENT MEGOLDÁS

Mindenki tisztában van vele, hogy manapság a Windows platform különböző elemei a legkeresettebb célpontjai a rosszindulatú támadásoknak, behatolásoknak. Ebben a cikkben nem ezen tény okairól folytatunk akadémikus jellegű elmélkedést, hanem a praktikus védekezés egyik ismert módszerére koncentrálunk. A kis- és közepes hálózatokra szabott, támogatott és ingyenes megoldás jelenlegi és következő változatát vesszük górcső alá.

Rendszermonitorozás a .NET-ben — 2. rész

TELJESÍTMÉNYSZÁMLÁLÓK

A .NET komponensek segítségével hozzáférhetünk a Windows teljesítményszámlálóihoz, megjeleníthetjük, vagy feldolgozhatjuk a kapott értékeket. Lehetőségünk van arra is, hogy saját, egyedi számlálókat hozzunk létre, amelyek gyűjtik az alkalmazásunk különféle teljesítménymutatóira vonatkozó adatokat.

Windows XP SP2

CSOPORTHÁZIREND ÚJDONSÁGOK

Az új szervizcsomag révén az XP sokat változott, mert a szándékosan szigorú megoldások beépítése mellett új lehetőségek is rendelkezésre állnak. Ezek a változások akkor érvényesülnek sikeresen tartományi környezetben, ha az üzemeltetést segítő támogatással is rendelkeznek. Szemezgessünk tehát: milyen változások történtek a Csoportházirendben?

ASP.NET 2.0 (Whidbey)

MI VÁRHATÓ A 2005. ÉVI ASP.NET-BEN?

III. RÉSZ: PERSONALIZATION ÉS MEMBERSHIP API – TESTRESZABHATÓ WEBLAPOKHOZ

Miért vásárolok az „amazonon"? Azért, mert mindig ajánl nekem újabb és újabb könyveket, amelyek tényleg érdekesek a számomra. Figyeli miket vásároltam, mások miket vásárolnak, figyeli a szokásaimat és ezek alapján nagy valószínűséggel megtippeli, mi érdekelhet. A weboldal teljesen az én ízlésemnek megfelelő adatokat szolgáltat nekem, és ugyanezt teszi másoknak is. A titok kulcsa: Personalization, azaz személyes ízlésre szabás. A sorozat mostani részében megnézzük, hogyan lehet ezt megoldani az ASP.NET 2.0 eszközeivel.

Tippek & trükkök

FÓKUSZBAN A WINDOWS TCP/IP

Egy-egy „trükk" megismerése sokat segíthet napi munkánkban. Ezúttal a Windows TCP/IP témaköréhez adunk tippeket, és folytatjuk az SQL Server 2000-ről szóló tippek korábbi számunkban megkezdett sorozatát.

Microsoft CRM v 1.2 — 3. rész

A MICROSOFT CRM TESTRESZABÁSÁNAK LEHETŐSÉGEI

Mielőtt még fejlesztésen kezdenénk gondolkodni, vegyük sorra, miket szabhatunk testre a CRM beépített eszközeivel. Ezeket a területeket tekintjük át átfogóan cikksorozatunk harmadik részében.

Magyarországon is elérhető a token alapú azonosítási technológia

A Microsoft Magyarország és az RSA Security bejelentette, hogy Magyarországon is elérhetővé vált az RSA SecurID token alapú azonosítási technológia, amely még nagyobb biztonságot nyújt a Windows operációs rendszerrel dolgozó vállalatok számára.

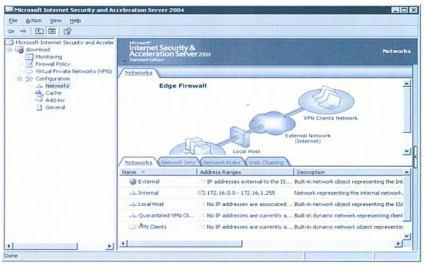
Ami a hivatalos Microsoft tanfolyamokból kimaradt...

MICROSOFT OFFICE 2003 – FRONTPAGE

Folytatjuk a „csináld magad" megoldásokat a Microsoft Office 2003 eszközeinek segítségével. Rovatunk mostani cikkében egy, a FrontPage 2003 által elkészíthető megoldás kerül nagyító alá. A FrontPage 2003 egyike az egyre népszerűbb kis- és középvállalatok részére kínált intranetes „csináld magad" megoldások eszközeinek.

Dr. Watson

ISA SERVER 2004 ÁTTEKINTÉS



Dr. Watson most nem kevesebbre vállalkozik, mint hogy néhány oldalban összefoglalja az ISA Server számos újdonságát. Ez azért számít kihívásnak, mert az ISA 2000-nek halvány nyoma sem maradt az új termékben. Új a felhasználói felület, a tűzfalkoncepció, a szabályok kiértékelése, a hálózatok kezelése, a naplózás és még sok minden más. Ahogy elnéztem, egyedül a jelentéskészítés hasonlít az elődmegoldására.

A Windows Media DRM Platform

...AVAGY DIGITÁLIS TARTALOMVÉDELEM FELSŐFOKON

A digitális kor beköszöntével egyre nagyobb az igény egy olyan, átfogó keretrendszerre, amellyel a tartalomszolgáltatók biztosíthatják tartalmuk védelmét és ezen biztonság tudatában olyan multimédiás anyagokat tehetnek közzé, amelyeket a védelem hiánya miatt eddig nem mertek.

A digitális kor, avagy mi az, ami eddig kimaradt.

Kétségtelen, hogy a digitális világ korát éljük, amely kitágítja a látóterünket minden szempontból. Elkezdtünk MP3-akat letölteni, CD-re kiírni őket, letölteni filmeket, a felhasznált tartalomért viszont nem mindenkinek akaródzott megfizetni a megfelelő jogdíjakat a szolgáltatóknak. Az természetes, hogy ez így nem volt biznisz a szórakoztatóipari cégeknek, ezért online nem publikálták termékeiket.

Megszületik...

Szinte szomjazott a piac egy olyan technológiára, amely megnyitja a digitális tartalomszolgáltatás kapuit, erre azonban egészen 1999-ig várni kellett, ekkor jelent meg ugyanis a Windows Media DRM (Digital Rights Management), amely megfelelő biztonságot nyújt, nagyfokú testreszabhatóság mellett. A platform egy rugalmas rendszer ami lehetővé teszi digitális média (videó és audio) tartalmak védelmét és biztonságos továbbítását akár számítógépre, akár egyéb eszközre.

Ez a platform új lehetőségeket nyit meg felhasználók és kiadók előtt egyaránt.

Ez mindenkinek a hasznára válik, hisz a kiadók biztosíthatják anyagaik védelmét és így olyan új tartalmakat tehetnek elérhetővé, amelyekről eddig álmodni sem mertünk, mindezt úgy, hogy a védelem a lehető legkisebb fennakadást okozza – sőt, ha szerencsénk van, észre sem vesszük. (Képzeljük el például, hogy díjfizetés ellenében élőben nézhetjük kedvenc zenekarunk koncertjét a számítógépünk előtt ülve.)

Szerva itt...

Amennyiben abban a szerencsés helyzetben vagyunk, hogy mi rendelkezünk a művekkel, valószínűleg már eljátszottunk azzal a gondolattal, hogy az Internet segítségével új bevételi forrásokra tegyünk szert. Tegyük fel, hogy mi vagyunk az *Olcsó Csokoládé* zenekar promóciójáért felelős személyek. Az eddigi csatornákon kívül olyan új dolgokba vághatunk bele mint például:

- Kiadás előtt már egy számot az új albumról (díjfizetés ellenében) publikálunk, amolyan kedvcsinálónak.
- Az együttes koncertjére magunkkal viszünk egy kamerát és egy internetkapcsolaton keresztül élőben közvetíthetjük.
- Ha nem látjuk garantáltnak azt, hogy a kiadott CD jól fogyna, választhatjuk a csak digitális kiadást/terjesztést, ebben az esetben egy kisebb vagyont is megspórolhatunk, viszont a jogdíjakhoz ugyanúgy hozzájutunk.
- Elérhetővé tehetjük a videoklipeket akár díjfizetés, akár mondjuk egy ingyenes regisztráció ellenében.
- A CD megvásárlásakor a felhasználó azonnal hozzáférhet a számokhoz (ezzel is áthidalva a szállítási időt) és egy fix idő után ez a digitális kópia nem hallgatható többé (feltételezzük, hogy a felhasználó X nap után már megkapja a CD-t.)
- Egy ellenőrző rendszerrel összeköttve az eredeti lemezek tulajdonosainak plusz számokat tehetünk elérhetővé, így is ösztönözve őket a vásárlásra.

A lehetőségek száma – mint látjuk –, szinte korlátlan. A technológia úgy van tervezve, hogy az esetleges biztonsági hibák ellen is védve van a tartalmunk.

...csere ott.

Azzal, hogy a szórakoztatóipar nagyfőnökei nagyobb védelemben tudják az anyagukat, könnyebben rávehetőek arra, hogy ne csak CD-rendelésre lehessen használni a világhálót. Csak egy pár a már létező megoldások közül:

- **Online zeneáruház:** Képzeljük el, hogy kedvenc zenekarunk CD-jéhez digitális formátumban a korong árának töredékéért juthatunk hozzá. (Általában 2500-3000 Ft-ért.) Ez az olcsóság két részről van megtámogatva, egyrészt megspóroljuk a CD gyártási, szállítási, valamint egyéb költségeit és a CD boltokat is („sajnos") megfosztjuk a maguk haszonkulcsától.
- **Jutalmazás:** Ha mégis a CD vásárlás mellett döntünk, ideális esetben azt is észrevehetjük, hogy a CD-t számítógépbe helyezve egy oldalra irányít minket, ahol megtekinthetünk egy pár videoklipet, vagy esetleg pár plusz új számot hallgathatunk meg (a lemez eredetiségének ellenőrzése után).
- **Osszuk meg az élményt:** Amennyiben kedvünk támad, a zene CD-ről elküldhetünk egy-két számot legjobb barátunknak (legálisan!) meghallgatás céljából (korlátozott számú lejátszással).

Nézzük ezt a gyakorlatban!

Vegyük azt az esetet, hogy Kis Jolán zeneimádó középiskolásként felkeresi kedvenc zenekara (Olcsó Csokoládé) weboldalát, ám még csak egy CD-jükkel rendelkezik. Észreveszi, hogy az új album már készül, de az első maxi már most letölthető az internetről, Jolán a következőket teszi:

- Rákattint a böngészőben a megfelelő linkre („Töltsd le a legújabb számunkat, mely az Étcsoki címet kapta!")
- A szerver egy fizetési oldalra irányítja őt, ahol választhat mondjuk az emelt díjas SMS vagy a bankkártyás fizetés közül, fiatal lévén előbbit választja és elküldi az SMS-t.
- Az összeg levonódik az egyenlegéből és átutalásra kerül a tartalomszolgáltatónak.
- A weblapon a megfelelő helyre beírja az SMS-ben kapott „válaszkódot".
- A szerver (a megfelelő pipeline lefutása után) előállít Jolánnak egy licencet a tartalomhoz, mely körülbelül így néz ki:

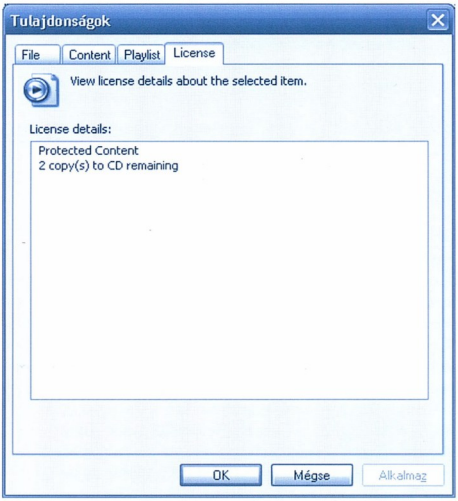
```
LICENC a következő tartalomhoz: 01 Étcsoki.wma
Érvényes: 2005. 01. 03. <= ekkor jön ki az új
album, utána már vásárolja meg a lemezt.
Lejátszások száma: korlátlan <= ha már fizetett
a használatért, hallgathassa annyiszor, amennyi-
szer szeretné.
CD-re kiírható: Nem. <= megelőzendő a további
illegális felhasználást.
```

- A licencet a szerver a számítógépre telepíti, majd ezután letölti a szóban forgó WMA fájlt a számítógépre.
- Jolán megnyitja a fájlt és jogosult a licencben megszabott műveletek végrehajtására.

És most a mélyvíz...

Nézzük most meg a folyamatot technikai oldalról, avagy mi történt Jolán műveletei alatt:

- A Windows Media DRM egészen a sikeres fizetés utáni pontig nem avatkozik be, addig minden a megszokott rend szerint megy (SMS fizetés ellenőrzése ASP-n, COM+on keresztül, SQL műveletek).
- A fizetési művelet ellenőrzése után a licenckiszolgáló előállít egy licencet a meghatározott üzleti logika alapján.
- Miután a biztonsági verziószámot (ld. később) és azt, hogy a kliens már rendelkezik egyéni „Black Box-szal" (ugyancsak később) ellenőrizte, a böngészőn keresztül a kliensre telepíti a licencet.
- A Windows Media Player a tartalom megnyitásakor észleli, hogy védett tartalomról van szó, ezért licencet keres hozzá a számítógépen. (Amennyiben nem találna hozzá, abban az esetben egy, a tartalomkódolás alatt meghatározott URL-re irányítja Jolánt, ahol megvásárolhatja a tartalmat, vagy letöltheti a licencet. (Ez az ún. License Aquisition URL, továbbiakban LAU.)
- Ha a WMP megtalálja a licencet, feldolgozza az abban található jogokat és tiltásokat és amennyiben a lejátszás engedélyezve van, lejátssza.



Egy védett fájl tulajdonságai

Természetesen, mint minden technológiánál, itt is derülnek ki biztonsági hiányosságok, ezért a DRM platformon is meg kell valamilyen módon különböztetni a verziókat, mely különkülön vonatkozik a kliensre és a szerverre is. Minden kliens alkalmazás (nemcsak a WMP) rendelkezik egy ún. biztonsági verziószámmal, mely arról is felvilágosítást nyújt, hogy mely biztonsági lyukak ellen van védve az adott program. A maximális biztonság érdekében szerveroldalon ellenőrizhetjük és korlátozhatjuk a hozzáférést verziószám alapján, így is elkerülve a jogosulatlan felhasználást.

Minden számítógépre alkalmazásonként az első DRM-védett fájl lejátszása előtt egy ún. egyéni *„Black Box"* DLL kerül (ezt a folyamatot a WMP individualization-nek [~egyéniesítés] hívja), mely hardver- és szoftverspecifikus és csak arra a számítógépre jellemző adatokat tartalmaz, a licencek ehhez a Black Box DLL-hez kötődnek. (Ugyancsak ez tartalmazza a licenctranzakcióknál használt publikus kulcsot is.) Ez azzal az előnnyel is jár, hogy ha valami vé-

letlen folytán feltörhetővé válna a DRM komponens az alkalmazáson belül, az csakis kizárólag azt az egy programot érintené.

A licencek

A licencek a tartalom „kulcsai", ezek tartalmazzák védett formában a tartalom privát kulcsát. A licenctelepítésnek három formája létezik:

1. **Csendes, ún. „Silent" licenctelepítés:** Ekkor a felhasználót mindössze egy pár másodperce felvillanó „Licenc igénylése..." nevű szöveg tájékoztatja arról, hogy voltaképpen egy védett fájlt hallgat
2. **Interaktív, „Non-silent" licenctelepítés:** A védett fájl megnyitásakor a Media Playerben egy ablak nyílik meg, amely egy weboldalt hoz be. Itt különböző adatokat kérhetünk be (pl. SMS visszaigazoló kód), majd ezek ellenőrzése után települ a licenc.
3. **Előtelepítés (Pre-delivery):** Ekkor a licenc már a megnyitás előtt a számítógépen van, ez akkor lehetséges, ha a fájlt böngészőből töltötte le és az már előtte felmásolta a szükséges kulcsokat.

A WMDRM Toolkit számos lehetőséget kínál annak beállítására, hogy ki, mikor, meddig és mit csinálhat a tartalmunkkal, ezeket nevezzük jogosultságoknak, melyek közül szemezgetünk egy picit:

WMRMRights.AllowBackupRestore
(alapbeállításban: True)
Amennyiben ez engedélyezve van, a felhasználó jogosult a licencek biztonsági mentésére. Ezt a funkciót ne használjuk, ha valamilyen számlálóhoz (lejátszás száma) kötött korlátozásunk van, mert a licenc lementésekor ezeket a változókat aktuális állapotukban tárolja le és visszaállításkor is a tárolt értékre állnak vissza.

WMRMRights.AllowBurnToCD
(alapbeállításban: True)
Ez a jogosultság Windows Media Audio (WMA) fájloknál lehetővé teszi a felhasználónak hogy CDDA formátumban normál zenei CD-ként kiírja. Ezt nem ajánlott engedélyezni, hisz a tartalom kiírása után védelem nélkül marad.

WMRMRights.AllowPlayOnPC
(alapbeállításban: True)
Magáért beszél, a számítógépen való lejátszást engedélyezi.

WMRMRights.AllowTransferToNonSDMI
(alapbeállításban: True)
Ha ez engedélyezve van, a felhasználó jogosult arra, hogy a jogkezelést nem támogató hordozható eszközre másolja a tartalmat (ugyancsak nem ajánlott engedélyezni, a CD-íráshoz hasonló okokból).

WMRMRights.AllowTransferToSDMI
(alapbeállításban: True)
Engedélyezi a másolásvédelmet támogató hordozható eszközökön a tartalom használatát.

WMRMRights.BeginDate
Megszabja azt a dátumot, amely *előtt* a tartalom nem használható/játszható le.

WMRMRights.ExpirationDate
A tartalom *végső* felhasználásnak ideje, ezután nem lesz használható.
WMRMRights.BurnToCDCount
Megszabja, hányszor írható ki a zenei tartalom CD-re.
WMRMRights.DeleteOnClockRollback
(alapbeállításban: False)
Ha ezt engedélyezzük (az időhöz/dátumhoz kötött licencek esetében), a rendszeróra visszaállítása („buherálása") esetén a licenc automatikusan törlődik a számítógépről, javasolt engedélyezni.

WMRMRights.ExcludeApplication:
Ezzel a funkcióval kizárhatunk meghatározott lejátszó alkalmazásokat az engedélyezett körből (Ha például csak Windows Media Player-re szeretnénk engedni a lejátszást, nem bízunk a WinAmp-ban).

WMRMRights.ExpirationAfterFirstUse
Megszabhatjuk (órában), hogy az első megnyitástól számítva meddig lehessen lejátszani a fájlt.

WMRMRights.ExpirationOnStore
Korlátozhatjuk, hogy a licenc telepítése után hány óráig legyen érvényes.

WMRMRights.MinimumAppSecurity
Beállíthatjuk a minimum biztonsági verziószintet.

WMRMRights.Playcount
Az engedélyezett lejátszások számát állíthatjuk be.

WMRMRights.SetSAPMode
A Secure Audio Path használatát engedélyezhetjük. A SAP használata esetén a hang titkosítva jut el a rendszer rétegein keresztül egészen a hangkártyáig és így megakadályozzuk a különböző hangfelvevő programokat (Windows Sound Recorder, SoundForge, stb.) abban, hogy tartalmunkat engedély nélkül rögzítsék.

A szerverről

Lássuk, mire van szükségünk ahhoz, hogy mi is be tudjuk üzemelni saját kis zeneáruházunkat (általánosságban elmondható, hogy mindegyik szerver minimum Windows 2000-et és SQL Server 2000-et igényel):

- **Webszerver:** Ez nyújtja az alap felületet letöltéshez.
- **Médiaszerver:** A védett médiafájlokat tárolja és engedi azok letöltését. (Ha streamelés is történik, a Windows Media szolgáltatásoknak telepítve kell lennie!)
- **Licenckiszolgáló:** Futtatja a WMDRM SDK-t, kiosztja, tárolja, naplózza a licenceket, válaszol a lejátszó alkalmazások kéréseire.
- **Adatbázisszerver:** Tárolja a licenceket, a védett médiák adatait, az üzleti logikát és a fenti három szerver naplóit.

Fejlesztgessünk...

A szerveren az SDK fizikailag egy COM+ objektumként létezik, amelyet ASP, vagy ASP.NET-kód hív meg és kommunikál vele. Ezek a komponensek akár egy Windows XP Pro-s, IIS-sel rendelkező gépen is működőképesek, így tesztcélra egy XP Pro+MSDE kombináció elegendő lehet.

A szerver futásának előfeltétele, hogy saját, generált privát kulccsal kell rendelkeznie, valamint egy, a Microsoft által kiállított tanúsítványnak is telepítve kell lennie. Az SDK-hoz tartozó súgó fájl részletes útmutatást nyújt az összes funkcióval kapcsolatban.

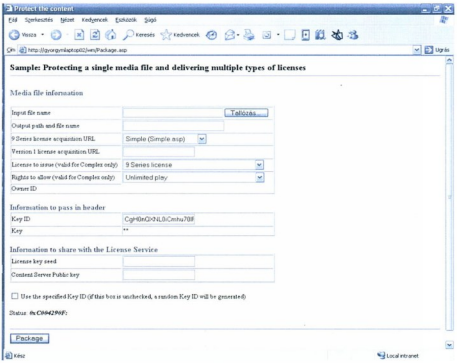
Egy mintarendszer felállítása:

1. **Az ASP fájlok másolása:** Az SDK-hoz kapott minta ASP-fájlokat másoljuk be a c:\inetpub\wwwroot\ könyvtárban létrehozott, WM könyvtárba.
2. **A kulcsok előállítása:** Az SDK súgójában szereplő VBS-kódot kimásolva és lefuttatva hozzuk létre a szerver privát és publikus kulcsát, valamint az ezekhez szükséges „seed" értéket.
   Megjegyzés: Ezekre a fájlokra úgy vigyázzunk, mint szemünk fényére, hisz ha valaki ezen kulcsok birtokába kerül, azon nyomban hozzáférhet bármilyen tartalmunkhoz!
3. **A kulcsok beszúrása, testreszabás:** A C:\Inetpub\wwwroot\wm\ könyvtár global.asa fájlját szerkesztve frissítsd a következő mezőket és mentsd el a fájlt.:

```
Siteurl: http://teszervered.hu/wm
Seed: "<ez tartalmazza a seed.txt tartalmát,
a script generálta a második pontban.>"
Contentserverpubkey: "<pubkey.txt tartalma>"
Contentserverprivkey: "<privkey.txt tartalma>"
```

4. **A global.asa engedélyezése:** Az SDK komponenseinek futtatásához szükség van a webszerveren a global.asa engedélyezésére, mely az IIS beépülő modulból (*Internet szolgáltatások kezelője*) végezhető el. Kattintsunk az alapértelmezett webhelyre, majd a WM könyvtáron a jobb egérgombbal kattintva válasszuk a *Tulajdonságok*at. A *Könyvtár* fülön, az Alkalmazás beállításoknál kattintsunk a létrehozás gombra. A végrehajtási engedélyeknél válasszuk ki a *„Parancsfájlok és végrehajtható fájlok"*-at, majd kattintsunk az OK gombra.
5. **Az IIS újraindítása:** A beállítások érvénybe léptetéséhez újra kell indítani a webkiszolgálót. (Start →Futtatás→CMD.EXE→net stop w3svc→net start w3svc)
6. **Tanúsítvány letöltése:** Az **[1]** címet meglátogatva kattints az *„Enroll to get a new certificate"*-re és kövesd az utasításokat. Egy email-t fogsz kapni, mely egy kódot tartalmaz, ugyanezen oldalra visszalépve a második menüpontot (*„Complete the enrollment with your e-mail confirmation"*) választva töltheted le a tanúsítványodat. Fokozottan ajánlott a visszavonási lista letöltése (ez tartalmazza a bizonyítottan „lyukas" alkalmazások listáját és azonosítóját), mely a *„Download the latest License Service Information"* linkről érhető el.

Ezek után a *http://teszervered.hu/wm/Package.asp* megnyitásával tesztelheted a frissen telepített SDK-dat.



A csomagoló eszköz

Végezetül

- Minden rendszer olyan biztonságos, mint annak leggyengébb pontja. (Jelen esetben a kulcsok védelméről gondoskodjunk!)
- Ne feledjük, a védett médiafájl licenc nélkül használhatatlan, nyugodtan terjeszthetjük őket CD-n, interneten vagy bármilyen más módon, anélkül úgyis a LAU-ra kerülnek a felhasználók.
- Nem Windows Media formátumú tartalmakat először ilyen formátumúvá kell átalakítani. (Erre eszköz pl. a Windows Media Encoder)

MOLDOVA GYÖRGY
i-gyomol@microsoft.com
Microsoft Magyarország
MVP, MCSE+I, MVTS

A cikkben szereplő URL:

[1] http://licenseserver.windowsmedia.com

SUS és WUS

A köztes patch management megoldás

Mindenki tisztában van vele, hogy manapság a Windows platform különböző elemei a legkeresettebb célpontjai a rosszindulatú támadásoknak, behatolásoknak. Ebben a cikkben nem ezen tény okairól folytatunk akadémikus jellegű elmélkedést, hanem a praktikus védekezés egyik ismert módszerére koncentrálunk. A kis- és közepes hálózatokra szabott, támogatott és ingyenes megoldás jelenlegi és következő változatát vesszük górcső alá.

A „patch management" (biztonsági javítások kezelése) nem a legszebb szókapcsolat egy magyar nyelvű újságban, de van úgy, hogy nincs jobb, mint az eredeti (az vesse rám az első követ, aki tud erre igazán frappánsat magyarul). Mindenesetre ez a kifejezés az elmúlt 1,5-2 évben eléggé sűrűn hallatszik a Microsoft felől. Mióta a „Megbízható számítástechnika" vezérelv érvényesül minden területen, azóta mindenképpen toplistás lett. Úgy ahogy a SUS (Software Update Services) is, amely a kritikus hibajavítások és szervizcsomagok központi (egyszeri) letöltését és automatikus terítését oldja meg egy Windows tartományban. Persze van termék, amely már korábban is (sőt, jóval nagyobb eszközkészlettel) lefedte a SUS funkcionalitását, ez pedig nem más, mint az SMS (Systems Management Server). Viszont ezzel van egy kis bibi: az SMS kifejezetten nagyvállalatoknak szánt termék, annak összes előnyével és hátrányával, így alkalmazásával előállhat az „ágyúval verébre, arany ágyúgolyóval' szituáció. Ezért mondhatjuk azt, hogy a SUS a köztes javasolt megoldás (úgy körülbelül 1000 kliensig biztosan), hiszen az otthoni vagy munkacsoportban működő számítógépekre szánt webes felületű manuális beavatkozást igénylő Windows/Office Update vagy a szintén szóló gépekre tervezett Automatic Update kliens és a nagyvállalati, komplexebb szolgáltatásokat nyújtó SMS között foglal helyet a patch management megoldásokat tekintve. Ezt alátámasztja az ezen a helyen található remek összehasonlítás is **[1]**.

Múlt, jelen, jövő

A SUS első változata a SUS 1.0 2002 júniusában született. Ahhoz képest, hogy mennyire fontosnak tekinthető egy jelentős vásárlói szegmens számára, minimális felhajtás volt körülötte, ami viszont talán rögtön érthetővé válik, ha néhány gyermekbetegségét is számba vesszük:

- csak tagkiszolgálóra volt telepíthető, sem SBS-re, sem tartományvezérlőre nem
- szervizcsomagot nem lehetett vele telepíteni
- minimális Csoportházirendben használható opció (2 db)
- minimális információ a telepítendő javításokról

De ezzel még nem volt vége a kényelmetlenségeknek, kedvelt jelenség volt pl. az is, hogy a javítás telepítése után egyből újraindult a kliens gép, a felhasználók legnagyobb örömére.
Fél év után, 2003 januárjában változott a helyzet, mert kijött a SUS SP1, amely önmagában is telepíthető volt, de az előző verzióról történő frissítés is szépen működött. Gyakorlatilag nem írták újra, hanem - többek között - csak a fentebb felsorolt problémákat, hiányosságokat javították illetve pótolták, valamint megújult a kliens alkalmazás is, és kiegészítették a Csoportházirend opciókat (4 db). E cikkben legnagyobb részben ezzel a változattal foglalkozunk, hiszen még ma is ez az aktuális verzió.

A technológia típusa	Microsoft ajánlás
Frissítés vizsgálat	Microsoft Baseline Security Analyzer (MBSA) 1.2.1
	Office Update Inventory Tool 2.1
Online frissítési szolgáltatások	Windows Update
	Office Update
Automatikus frissítés kezelés	Automatic Updates funkció a Windows-ban
	SUS 1.0 SP1
	SMS 2003 SP1

A Microsoft hibajavítási/frissítési eszköz ajánlása

Ám a legutolsó hírek szerint (a Windows XP SP2 miatt kissé csúszva) 2005 első negyedében megjelenik a per pillanat még WUS (Windows Update Services) néven futó utód. Erről az új és valóban komoly változásokat hozó verzióról a cikk utolsó oldalain ejtek néhány szót.

A SUS működéséről tömören

A SUS működése a kliens-szerver modell alapján történik. A SUS szerver ugyanazt a technológiát használja, mint amelyet a Microsoft alkalmaz a publikus Windows Update oldalnál, míg a SUS kliens gyakorlatilag ugyanazt az Automatic Update klienst jelenti, amellyel pl. a Windows XP (SUS hiányában) közvetlenül a Windows Update szerverekről végzi a frissítéseket.

Két komponens van tehát, ezek közül a SUS szervert csak és kizárólag Windows 2000/2003 kiszolgálóra tehetjük, célszerűen a tűzfal mögé. Az AU klienst pedig a Windows 2000 SP2-től bármelyik operációs rendszer futtathatja (természetesen a szerver verziók is). Az ügyfél operációs rendszerek között csupán annyi különbség van, hogy a Windows 2000 SP2, illetve a Windows XP RTM változatai nem tartalmazzák alapból az AU klienst, ezért fel kell rá telepítenünk (wuau22.msi), akár manuálisan, akár pl. a Csoportházirend segítségével. Még annyi érdekességképpen, hogy az Automatic Updates kliens 24 nyelven érhető el, míg a szerverből angol és japán változatok léteznek.

A működés egyik sarokpontja az, hogy kizárólag a SUS szerver fog szinkronizálni a Windows Update szerverekkel, az üzemeltető által kiválasztott módon (alkalomszerűen manuálisan indítva vagy időzítve). A szinkronizáció csak a fenti operációs rendszerek és gyári komponenseik (IIS, OE, Messenger, stb.) kritikus frissítéseire vonatkozik, plusz a csoportosított biztonsági javítócsomagokra (security rollups) és a szervizcsomagokra.

Amennyiben egy frissítést letöltünk, még nem települ automatikusan, hanem először engedélyeznünk kell (approve). Ez fáradságos munka, mert egyelőre egyesével kell megtennünk, a csoportos kijelölés a WUS egyik „jósága" lesz. Ha engedélyeztük, akkor a szerveroldalon kész vagyunk, viszont helyes beállítások esetén a kliensoldalon sem kell semmi továbbit cselekednünk, hiszen amikor a kliens legközelebb kapcsolatba lép a szerverrel, akkor egy gyors vizsgálat után kiderül, hogy a legutoljára letöltött csomagok közül jár-e neki valami, és ha igen, akkor a vonatkozó beállítások alapján letölti és/vagy telepíti is.

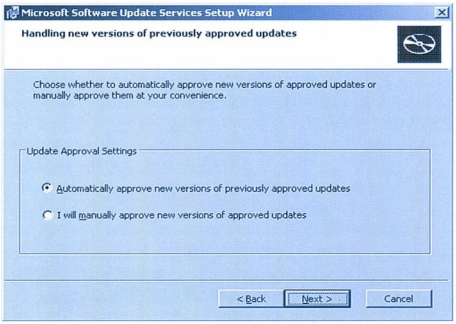
SUS előkészületek és telepítés

A SUS szerverhez feltétlenül kell ugyanazon a gépen egy IIS (és a szabad 80-as port, mert a kliensekkel csak ezen az egy porton keresztül tud kommunikálni), és pár GB-nyi háttértár. Ennek méretével kapcsolatban 6 GB az ajánlás, ám ez a nyelvektől függően kevesebb is lehet. Személyes tapasztalatom szerint egy kb. 100 gépes hálózatban, két nyelvvel, kétfajta klienssel és ugyancsak kétfajta kiszolgálószoftverrel a kezdetek óta működtetve, a helyfoglalás kb. 2,5 GB.

Az előkészületekkel kapcsolatban maximum a letöltendő tartalom (Update Storage) külön partícióra helyezését indokolt megemlíteni. Maga a telepítés (a letölthető csomagból a Sus10sp1.exe indításával **[2]**) „next-next-finish" típusú, nem okozhat problémát. Telepítés közben kiválaszthatjuk azokat a nyelveket, melyekhez passzoló nyelvű szoftvereinkhez a SUS a frissítéseket le fogja tölteni, valamint eldönthetjük azt is, hogy abban a kényes szituációban, amikor egy már engedélyezett javításhoz ad ki egy javítást a Microsoft, akkor annak terítését automatikusan engedélyezzük, vagy nem.

A telepítéskor megadott beállításokat (a háttértárra vonatkozó kivételével, bár trükközni azért ott is lehet, lásd később), utólag is megváltoztathatjuk.

Egy fontos közlendő még hátravan a Windows 2000 Serverre telepített SUS-sal kapcsolatban. Mivel az IIS5 nem rendelkezik azokkal a biztonságosságot támogató megoldásokkal, mint a Windows Server 2003-ba integrált IIS6, ezért külső programokkal kell megtámogatni. Ilyen program pl. az IIS Lockdown és az URLScan is, amelyek a SUS telepítése közben automatikusan felkerülnek a rendszerünkre. Korlátozó hatásuk azonban az egész IIS-re kiterjed, ezért ezt vegyük figyelembe az esetleges későbbi webszerver problémák során. Ha volt már a rendszerünkön URLScan, akkor a frissítések letöltése is problémás lehet, ezért vagy szedjük le előtte, vagy az URLScan naplói alapján korrigáljuk a beállításait. Ezekről az eszközökről ezen a címen lehet informálódni **[3]**.



Automatikus vagy kézi engedélyezés a javított javításoknál?

A telepítés után rögtön hozzáláthatunk a SUS szerver beállításához, amelyet egy IE-ből érhetünk el a következő címen:

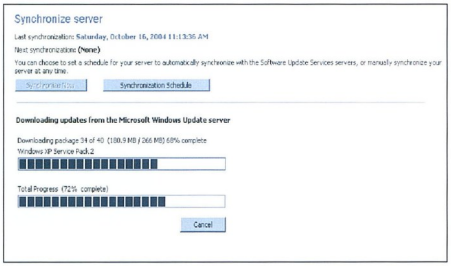
```
http://szerver_neve/susadmin
```

Célszerű a baloldali faszerkezetben a „*Set Options*" szakasszal kezdenünk. Először adjuk meg a szükséges proxy beállításokat, címet, portot és az esetleges fiókot is, amely nevében a SUS „kilát" majd a WU szerverekre. Ha a proxy szerverünk megköveteli (vagy nem tud mást) a Basic típusú hitelesítést, akkor pipáljuk ki a „*Allow basic authentication when connecting to proxy server*" négyzetet. A proxy hitelesítéssel kapcsolatban két fontos dologról kell még beszámolni:

1. Ha a SUS-t az ISA szerveren vagy egy (ISA-t is tartalmazó) SBS-en futtatjuk, akkor a felhasználói fiók kitöltésének formulája \\tartománynév\fióknév legyen.
2. A proxy konfigurációs beállítást a SUS csak akkor használja, ha a WU szerverekhez kapcsolódik. Abban az esetben, ha több SUS szerverünk van egy hierarchikus rendszerben, és szinkronizációra lesz szükség közöttük akkor a WinHTTP alapértelmezés szerint a WINHTTP_ACCESS_TYPE_NO_PROXY függvényt hívja meg, ezzel megtiltva, hogy az adott kliens SUS a netről töltsön le bármilyen tartalmat vagy engedélyezési listát egy esetlegesen hamis SUS szerverről.

A beállítások között továbbnavigálva a szerverünk NetBIOS nevét kell megadnunk, amelyről majd a kliens felismeri, valamint a „*Select which server to synchronize content from*" rész alatt el kell döntenünk, hogy a WU szerverekről vagy egy másik SUS szerverről történik majd a szinkronizáció (erről még később lesz szó). Ezután már csak a telepítés kapcsán már említett újraengedélyezést, illetve a frissítések nyelveit választhatjuk meg. Ám van még egy kicsit eldugott opció, közvetlenül a nyelvek felett, ez pedig a „*Select where you want to store updates*" amely választhatóvá teszi a frissítések elhelyezését, azaz két lehetőség van, vagy helyben vagy a WU szervereken hagyjuk

meg. Ez utóbbi esetén a SUS csak közvetít, az AU kliens fogja letölteni az adott gépre a speciális nyelvű frissítést.
Ezek után kezdődhet a szinkronizálás. Egyszerűen nyomjunk rá a „*Synchronize now*" gombra, és ezzel elkezdődhet a katalógus letöltése (Aucatalog1.cab és Aurtf1.cab állományok), majd egy többnyire hosszú-hosszú folyamat, merthogy a SUS a kezdetektől fogva ismert javításokat letölti ráadásul az összes kliens számára, azaz nincs lehetőségünk szűkíteni a letöltendő tartalmat az operációs rendszer vagy a komponenseik szintjén (ezért is kell óvatosan bánni a nyelvekkel).
Ezalatt bőven jut időnk pl. arra, hogy beállítsuk a szinkronizálás ütemezését, vagy megvizsgáljuk a panel alján található opciót, amellyel a szinkronizáció elakadása esetén bekövetkező automatikus újrapróbálkozások számát állíthatjuk be.
A másik teendőnk lehet – ebben a kieső időben – a kliensek tudtára adni, hogy van már SUS szerver a hálózatban, tessék használni! Ehhez a Csoportházirendbe kell belenyúlnunk.



És már tölt is, csak tölt, csak tölt...

Teendők a Csoportházirendben

Ketté kell választanunk teendőinket a Windows 2000 Server és a Windows Server 2003 különbözősége miatt, hiszen az első esetén teljesen üres lappal indulunk, azaz nincs beimportálva a szükséges sablon a Csoportházirendbe. De ezt könnyen megtehetjük, hogyha követjük az alábbi lépéseket:

- Másoljuk be az új sablont (a letöltött csomagból a wuau.adm állományt) a %windir%\inf mappába azon a tartományvezérlőn, amelyiken módosítani fogjuk a Csoportházirendet.
- Készítsünk egy új GPO-t, és nevezzük el egy tetszőleges, pl. „SUS beállítások" néven.
- Lépjünk rá erre az új objektumra, majd ezen belül a Computer Settings / Administrative Templates szakaszra.
- Jobb gombbal válasszuk az Add/Remove Templates parancsot, majd tallózzuk be a wuau.adm állományt, adjuk hozzá és kész is. Ezzel beértük a Windows Server 2003-at, mert abban alapból megtalálható a Windows Updates rész ugyancsak az Administrative Templates/Windows Components/Windows Update alatt, ugyanezzel a 4 opcióval.

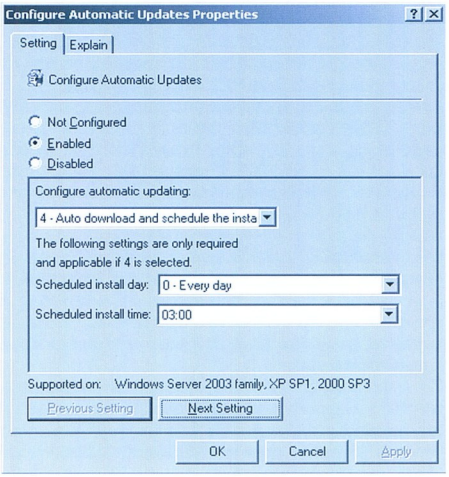
A SUS részletes tervezésére és méretezésére nem térek ki külön ebben a cikkben, mindenesetre a SUS GPO beállítások propagálása előtt azért át kell gondolnunk, hogy a hálózatunkban mely gépekre szeretnénk és melyekre nem a SUS szolgáltatásait igénybe venni. Mivel a Csoportházirend segítségével fogják észrevenni a kliensek, hogy ők SUS kliensek is egyben, legalább három alap megoldás körvonalazódhat szemünk előtt, ha elkezdünk tervezni:

1. Kis számú leendő SUS kliens esetén a kiválasztott számítógépek fiókjait berakjuk egy OU-ba és egy külön SUS GPO-val csak ennek az OU-nak tagjaihoz rendeljük hozzá a SUS beállításait.
2. Nincs külön OU, készítünk viszont szintén egy külön SUS GPO-t, majd a jelenlegi hierarchia szerint elrendezett OU-khoz hozzárendeljük egyesével.
3. Ha a tartomány minden gépére szeretnénk, hogy érvényesüljön a SUS hatása, módosíthatjuk akár a Default Domain Policyt is.

Meg kell még említeni azt is, hogy a Csoportházirend hatása alá soha nem kerülő klienseknél (pl. XP Home, vagy munkacsoportos variáció) is el lehet érni a SUS szerverre hangolódást, a regisztrációs adatbázis buherálásával, és persze gépenként egyesével.
Egyszóval a sablon és a GPO a helyén van, nézzük a kitöltését. Az első opció (*Configure Automatic Updates*) magára az engedélyezésre vonatkozik és arra, hogy a SUS kliensek milyen módon kapják meg a frissítéseket. Három számozott variáció létezik:

1. *Notify for download and notify for install:* ez a legkevésbé automata megoldás, hiszen mind a letöltés, mind a frissítés telepítése manuális a kliensen.
2. *Auto download and notify for install:* itt már csak a telepítéshez kell engedélyezés.
3. *Auto download and scheduled for install:* ennél a verziónál viszont mindkét esemény automatikus, pontosabban az időzítést a panel alján állíthatjuk be (a 2-es, 3-as opciónál ezek hatástalanok is). Ha ebben az időpontban nem lesz bekapcsolva a gép, akkor a következő belépés után történik meg a frissítés letöltése, illetve telepítése.

A második házirend beállítás (*Specify intranet Microsoft update service location*) a SUS szerver és a statisztikai szerver nevének (nem a SUS elérési útról van szó!) megadása. Teljes (FQDN) nevet használjunk a panelen lévő példával ellentétben, mindkét esetben.



Itt kiderül, hogyan és mikor húzhatóak le a frissítések

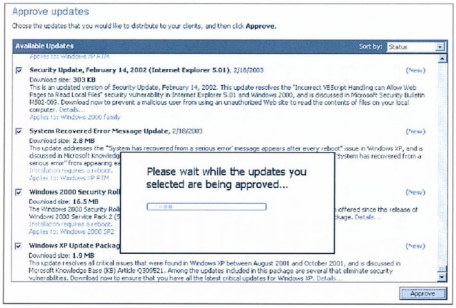
A harmadik választható beállítás (*Reschedule Automatic Updates scheduled installations*) az első új a SUS 1.0-hoz képest. Értéke percekben állítható 1-től 60-ig, és azért született, hogy a kliens bekapcsolása után időben kicsit eltolható legyen az elmaradt frissítések letöltése és telepítés kezdése. Az utolsó opció (*No auto-restart for scheduled Automatic Updates installations*) szintén egy felhasználó-kímélő megoldás: ha bekapcsoljuk, az AU kliens értesíti a telepítés után a felhasználót, annak szükségességéről. Ha nem kapcsoljuk be, akkor sem lesz szó nélküli automatikus újraindítás, hanem a kap egy közleményt a felhasználó arról, hogy 5 perc múlva lesz az újraindítás, akármit csinál, kapja össze magát.

Ha ezeket a beállításokat megtettük, készen is vagyunk, elvileg (és a Csoportházirend frissítési intervallumától függően) a kliensnek észre kell vennie a SUS szervert. De nézzük közben a fejleményeket a SUSAdmin oldalon, mert közben véget ért a szinkronizáció.

Az engedélyezés (Approve)

A SUS szerveren történő utolsó művelet az engedélyezés. Mint már korábban említettem ezt - az óvatosság miatt - mindig egyesével kell megtennünk az összes telepítendő csomag esetén, mert az „*Approve All*" gomb nincs, így marad a TAB > TAB > SPACE kombináció, ami a 2 perc után már egész dallamos ☺ (egy könnyítő tipp ehhez **[4]**). Van viszont rendező gomb, státusz, platform, dátum vagy cím szerint sorba rakhatjuk a csomagokat. Ha viszont tüzetesen megnézzük ezt a listát még az engedélyezés előtt, akkor két dolgot is kiszúrhatunk. Az egyik, hogy soronként az informális szöveg után a „Details" gombra kattintva megtekinthetjük a csomag részletes jellemzőit, a vonatkozó KB cikket, sőt itt kérhetjük a letöltést is, immár a saját háttértárunkból. A másik pedig az, hogy akármennyire is behatároltuk a nyelvi változatokat, fogunk találkozni pl. egy hasonlóval: „*Microsoft .NET Framework Service Pack 2, Japanese Version (SDK Applied)*". Pedig hát, ilyesmit abszolút nem kértünk. Sajna, arról van szó, hogy pl. a Framework vagy a Sharepoint Services komponens nem nyelvfüggő, ezért a SUS „a több jobb" alapon az összest letölti.

Az admin felületről még annyi érdekes elmondani, hogy mind a szinkronizáció, mind az engedélyezés naplózásra kerül, amelyeket meg is tekinthetünk, a baloldali faszerkezetből kiválasztva a megfelelőt. Még egy funkció érdekes lehet, a „*Monitor Server*" menüpont alatt megtekinthetjük, hogy mennyi (részben felesleges) csomagot szedett le összesen eddig a SUS. De nézzük, inkább mi történik a kliens oldalon ezután.



Éppen engedélyez a SUS

Az Automatic Updates kliensről

Amennyiben szükséges a kliens az első, SUS-hoz kapcsolódásakor automatikusan letölti a legújabb AU klienst (meg lehet figyelni pl. System Properties/Automatic Updates alatt a változást). Összesen egy esetben kell manuálisan telepítenünk az AU klienst, (azaz a letölthető csomagban található WUAU22.msi állományt), mégpedig akkor, ha Windows 2000 Professional SP2-vel rendelkezünk. Remélhetőleg azért ez nem túlságosan valószínű így 2004 végén. Az AU kliens további előnyei közé tartozik, hogy a BITS-et (Background Intelligent Transfer Service) használja a letöltések optimalizálására, amely viszont képes a sávszélesség háttérbeli, intelligens igénybevételére valamint arra, hogy letöltsön és telepítsen több frissítést is, egyszeri újraindítással.

Fontos tudni, hogy abban az esetben, ha az aktív felhasználó tagja a kliens gép helyi rendszergazdai csoportjának, akkor a totálisan háttérben zajló automatizmus sem működik, mert a felhasználónak kell engedélyezni a letöltést és a telepítést is, ergo ő láthatja egyedül az AU ikonját és a ballonokat is a tálcán. Ha viszont a felhasználó nem tagja ennek a csoportnak, semmilyen szinten nem avatkozhat bele interaktívan ebbe a folyamatba (kivéve, ha a Csoportházirendben az újraindítás késleltetést megengedjük, lásd fentebb).

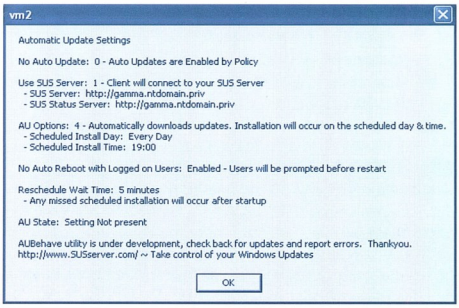
A kliens és a szerver közötti kapcsolatfelvétel első alkalommal a Csoportházirend beállítások magára húzását jelenti, majd ezek alapján a letöltés és a telepítés is megtörténhet. Innentől egy 17-22 órás intervallumban történik majd a kontaktus. Ezeket az adatokat (első kapcsolatfelvétel, utolsó, következő, stb.) ezen a helyen tekinthetjük meg a registryben:

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\
CurrentVersion\WindowsUpdate\Auto Update
```

A házirend beállítások azonnali „lehúzását" a szokásos parancsokkal tehetjük meg. Windows 2000 esetén a felsővel, XP és Windows 2003 esetén az alsóval:

```
- Secedit /refreshpolicy machine_policy /force
- Gpupdate /target:computer /force
```

A detektálás kierőszakolását tartományi körülmények között a SUS házirend hatásainak ignorálásával (mondjuk kivesszük az adott OU-ból, majd frissítünk a kliensen és visszatesszük) érhetjük el, munkacsoportos, azaz a helyi regisztrációs adatbázison keresztül konfigurált kliensnél pedig a következő KB cikk alapján **[5]**.



Az AUBehave.vbs hasznos ellenőrző eszköz

A kliens beállításainak ellenőrzéséhez pedig remekül használhatjuk a fenti képen látható eszközt (AUBehave.vbs), amely több más alkalmazással és információval együtt egy MVP kollégánk, Scott Korman weboldalán található. **[6]**
Van más módszer is az AU kliensek működésének ellenőrzésére, mégpedig az IIS naplóállományain keresztül. Amennyiben nem akarunk sokat mazsolázni, akkor megtehetjük, hogy megváltoztatjuk az IIS naplózást, úgy, hogy csak az AU kliensek forgalmát tartalmazza:

1. Nyissuk meg az IIS Admin MMC-t.
2. Válasszuk ki a Default Web Site-ot, majd a tulajdonságait, majd a Home Directory fülön töröljük a „Log Visits" négyzetet és OK.
3. A Default Web Site-ot kiválasztva keressük meg a jobboldali ablakban a wutrack.bin állományt.
4. A tulajdonságai között kattintsuk be a „Log Visits" négyzetet. Innentől más nem lesz az IIS naplókban, csak a SUS-AU forgalom részletei.

További SUS extrák

Több SUS szerver

Ha a tartományunk mérete, vagy földrajzi elhelyezkedése vagy éppen a hálózat szegmentáltsága megkívánja, akkor tetszetős SUS hierarchiát építhetünk fel. Ekkor mindig ki kell jelölni egy elsődleges SUS szerver-t, amely kapcsolatban lesz a WU szerverekkel, és a többi SUS szerverrel is. Ez a szerver fogja letölteni a frissítéseket, amelyet a többi leszinkronizál. Ilyenkor a kliens SUS-on a „*Set Options*" részben be kell állítanunk a hierarchiában fentebb lévő SUS szerver nevét. Megtehetjük még azt is, hogy csak az elsődleges SUS szerveren engedélyezünk frissítéseket és a kliensen meg bepipáljuk a „*Set Options"* alatt található erre vonatkozó jelölőnégyzetet szervereken (*Synchronize list of approved items updated from this location (replace mode)*). Ebben az esetben, az engedélyezési listában a frissítések pipálási helye „kiszürkül", azaz ez sem lesz változtatható a kliens SUS-ban. Laborkörülmények között kitűnően működött ez a felállás, és van meg egy előnye: ezzel a módszerrel költöztetni is lehet az egyik gépről a másikra a SUS-t, anélkül, hogy töltögetnünk kellene a WU szerverekről.

Betelt a partíció?

Ha netalántán elkövettük az azt a hibát, hogy egy kevés hellyel rendelkező partícióra tettük a SUS Update Storage-t és betelt, akkor akár a frissítések letöltése közben, akár később ezzel a kipróbált módszerrel tudunk helyet csinálni (feltéve, ha van szabad vagy felszabadítható partíciónk még a rendszerben):

1. Készítsünk egy új partíciót a Disk Management-ben.
2. Mozgassuk át a \SUS\Content mappa tartalmát az új partíció gyökerébe.
3. A Disk Management-ben kattintsunk a jobb gombbal az új partícióra, majd válasszuk ki a „Change Drive Letter and Path" menüpontot.
4. Válasszuk az „Add" parancsot, majd a „Mount in the following empty NTFS folder"-t.
5. Tallózzuk be a jelenleg már üres Content mappát (pl. C:\SUS\Content) és készen is vagyunk. Ezután a SUS vigan töltöget majd az eredeti helyre, persze mi tudjuk, hogy az már egy másik partíció.

A SUSAdmin HTTPS-en keresztül

Csak azok a felhasználók érhetik el a SUSAdmin oldalt, akik helyi rendszergazdai joggal rendelkeznek az adott SUS szerveren. Viszont nem mindig lehetséges a szerveren helyben dolgozni, vagy éppen sokkal kényelmesebb távolról, mondjuk a rendszergazda 21"-os monitora előtt kezelni a SUS-t. De mivel közismert, hogy a HTTP-n keresztüli forgalom abszolút nem biztonságos, ezért célszerű és lehetséges HTTPS-en keresztül használni a SUS kezelőfelületét. Magával a tanúsítvány létrehozásával és hozzárendelésével nem foglalkoznék, erről található részletes leírás ezeken a helyeken.
Ami viszont fontos, hogy a tanúsítványt a következő IIS virtuális mappákhoz kell hozzárendelni:

- Autoupdate\Administration *és* Autoupdate\Dictionaries
- Shared *és* Content\EULA

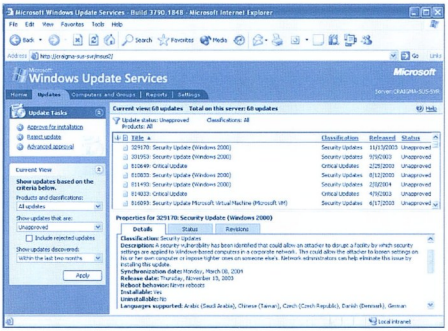
Értesítések

Ha jelentkezünk a Microsoft „*SUS e-mail notification*" szolgáltatására, akkor e-mailben azonnal informálnak bennünket arról, hogy dolgunk lesz, azaz szinkronizálnunk kell vagy legalábbis engedélyeznünk biztosan (bár van egy nem támogatott megoldás az automatikus engedélyezésre, kipróbáltam. működik, bátraknak részletek itt **[7]**), mert frissítések érkeznek/érkezni fognak a SUS szerverünkre. Erre a szolgáltatásra ezen a címen lehet jelentkezni **[8]**.

Összegzés (de még nincs vége)

A SUS egy remek és egyszerűen csatasorba állítható eszköz, ráadásul ingyenes (a licenszekkel sincs gondunk). Ha egyszer sikeresen beüzemeljük meg végképp kellemes, hiszen nincs vele utólag szinte semmi tennivaló. Viszont nem csodaszer, és amellett hogy van pár bosszantó hibája és hiánya, sok egyéb fontos műveletre nem alkalmas. Nem lehet vele az operációs rendszerhez eszközmeghajtókat vagy nem kritikus besorolású szoftverfrissítéseket sem telepíteni, és nem alkalmas Exchange, SQL vagy Office frissítések terítésére sem. De ne csüggedjünk, itt (lesz) a másik!

WUS (Windows Update Services)



Most körülbelül így néz ki a WUS

A helyzet az, hogy SUS 2 nem lesz, helyette viszont egy új nevű terméket kapunk, a WUS-t. 2004 tavaszától a bétatesztelők már használhatják az eddig elkészült verziókat, jómagam is kipróbáltam már tesztkörnyezetben és élesben egyaránt. Ahogy

már említettem, a Microsoft jelenlegi tervei szerint 2005 első negyedévében kész lesz az RTM, de arról is szó van, hogy a 2004-es év vége felé egy publikus bétát is kiadnak.
A terméken erősen látszik a fejlődés, több fájó ponton is javítottak, és több olyan új megoldás is belekerült, amelyek miatt tényleg egyszerűbb beüzemelni és sokrétűbben lehet használni is. A szolgáltatásait tekintve bizonyos szempontokból közeledik az SMS-hez, bár azért nyilván továbbra sem ennek kiváltása a cél.

Nézzünk egy felsorolást először a WUS és a SUS viszonyáról alap (és hangsúlyozom, tervezett) tulajdonságairól:

- Fut továbbra is Windows 2000/2003 Serveren *egyaránt*
- Nem „piszkít" bele a SUS életébe, mert a SUS továbbra is használható lesz a WU szerverekkel
- A WUS is kiszolgálhatja a SUS-t egy hierarchiában
- Lesz a SUS frissítéshez migrációs eszköz

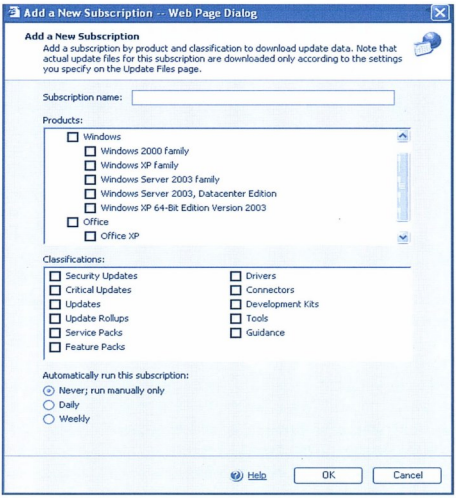
A WUS a következő termékeket frissítheti (természetesen az összes jövőbeni Microsoft terméken kívül):

- Windows 2000 SP3-tól felfelé mindegyik Windows
- Office XP SP2 és Office 2003
- SQL 2000 és MSDE 2000,
- Exchange Server 2003

A WUS platform/támogatási követelményei:

- Windows 2000 SP3 (Server: SP4) és későbbiek
- Windows XP RTM és későbbiek
- Windows Server 2003 és későbbiek
- Minden nyelvi változat (beleértve a MUI-kat is)

A működéssel kapcsolatban három fontos újdonságot mindenképpen kibontanék. Az első a telepítés és eltávolítás, azaz a WUS nemcsak telepíteni tud majd, hanem a megfelelő csomagokat el is távolítja a rendszerből. Ezt nyilván támogatnia kell majd a csomagoknak is és valószínűleg visszafelé nem is lesz kompatibilis, de azért kifejezetten szimpatikus ötlet. A második újdonság nagyobb ívű, hiszen ezzel vége az AD monopóliumnak, mert a WUS maga is képes lesz összegyűjteni a megcélzott számítógépeket és mi magunk tetszőleges csoportokba rakhatjuk majd ezeket a WUS felületén. Tehát a Helyi/Csoportházirend támogatás Active Directory környezetbe (*Client side lists*) megmarad, de kibővül a WUS-sal készíthető csoportokkal, AD nélküli környezetbe (*Server-side lists*). A csoportok kialakítása azért is fontos, mert lesz lehetőség beállítani azt, hogy bizonyos frissítéseket, csak bizonyos csoportokra telepítsen. A harmadik újdonsághoz passzol a következő kép, melyen látható, hogy a telepítendő frissítések leszedése ezentúl az ún. feliratkozásokon (Subscription) keresztül valósul meg. Ezekből természetesen többet is készíthetünk, és mindegyiken belül szűkíthetünk termékek szerint (azaz nem kell leszedni pl. a Windows 2000-es frissítéseket, ha nincs rá szükség) és (feledve a kritikus frissítések egyhangúságát), igencsak kibővített kategóriák állnak rendelkezésre. Az utóbbi két újdonság miatt kicsit alaposabban meg kell majd terveznünk a WUS működési körülményeit, de jóval többet is profitálunk majd ezeken a lehetőségeken keresztül.
Maga a letöltés is több fázisú, először csak a katalógus jön le, amely egy lista formájában látható, és amelyből választhatunk (akár többszörös kijelöléssel is), hogy melyeket szeretnénk letölteni. A frissítések (sorok a listában) mellett apró ikonokkal jelzi az állapotot (megjelenítve, letöltve, engedélyezve) a WUS, de lekérdezésekkel állapot, dátum és frissítési kategória szerint is megjeleníthetünk egyéni listákat.



Feliratkozás termékek ill. kategóriák szerint

Nagyon jó ötlet az is, hogy a WUS rendelkezik egy ún. „indító kereséssel", azaz képes előre megvizsgálni, hogy a mely javítások léteznek már a frissítendő kliensen. A frissítés teljesítését időkorláthoz (deadline) is köthetjük, azaz ha egy általunk megjelölt ideig nem települ az adott javítás, akkor az időkorlát eljövetelekor mindenképpen fog. A Csoportházirend opciói szintén erősen bővülnek, jelenleg tíznél tartunk, és olyan lehetőségeink lesznek, mint pl. a kliensek kapcsolódási periódusának hangolása vagy a nem admin jogú felhasználók értesítése.
Kifejezetten tetszetős megoldás még a WU kliens automatikus frissítésének megoldása központilag, valamint az egyelőre csak szkriptből elérhető ún. „*Big Red Button*" frissítési metódus is, de a különböző jelentések, visszajelzések megoldása is szenzációsan részletes lett. Lesz mit tanulnunk, az biztos.

GÁL TAMÁS
MCSE, MCSA, MVP (Software Distribution)
gtamas@tfszki.hu

A cikkben szereplő URL-ek:

[1] http://tinyurl.com/6b5xs
[2] http://tinyurl.com/5ihu
[3] http://tinyurl.com/54qql
[4] http://www.gatefold.co.uk/sus
[5] http://tinyurl.com/4hac7
[6] http://www.susserver.com
[7] http://tinyurl.com/5ruyj
[8] http://tinyurl.com/4ghyc

Rendszermonitorozás a .NET-ben II. rész

Teljesítményszámlálók

A .NET komponensek segítségével hozzáférhetünk a Windows teljesítményszámlálóihoz, megjeleníthetjük, vagy feldolgozhatjuk a kapott értékeket. Lehetőségünk van arra is, hogy saját, egyedi számlálókat hozzunk létre, amelyek gyűjtik az alkalmazásunk különféle teljesítménymutatóira vonatkozó adatokat.

A Windows teljesítményszámlálói

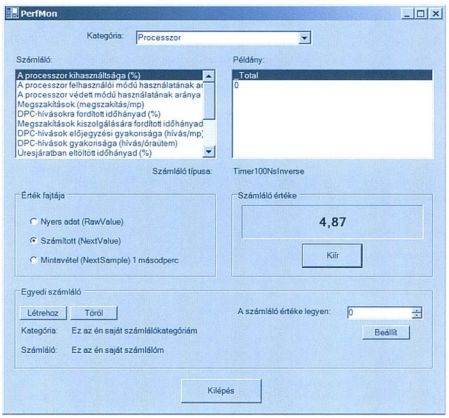
A Windows rendszerek teljesítményszámlálói (performance counters) a különböző rendszerkomponensek teljesítményadatainak gyűjtését végzik. A Windows számos előre gyártott teljesítményszámlálót tartalmaz, amelyek az operációs rendszerhez tartozó objektumok (hardver és szoftver) adataival foglalkoznak. Minden számláló a rendszer meghatározott funkciójához kapcsolódik, így találhatunk például a processzorra, a memóriára, a lemezegységekre, rendszerfolyamatokra és szálakra vonatkozó számlálókat is. Olyan számlálókkal is találkozhatunk, amelyek nem az operációs rendszerhez tartoznak, ezeket a rendszerre telepített különféle alkalmazások hozzák létre, hogy a felhasználók nyomon követhessék az adott alkalmazással kapcsolatos teljesítményadatokat.

A létező teljesítményszámlálókhoz való kapcsolódásra a .NET PerformanceCounter komponensét használhatjuk, amelynek segítségével alkalmazásunkat képessé tehetjük a számlálók adatainak megjelenítésére és feldolgozására. A rendszerhez, illetve más alkalmazásokhoz tartozó számlálók természetesen csak ReadOnly üzemmódban használhatók, de lehetőségünk van saját számlálók létrehozására is, amelyek értékét alkalmazásunk állíthatja be, és ha szükséges, meg is jelenítheti azt.

A cikkhez kapcsolódó mintaprogram **[1]** ezeket a műveleteket mutatja be, választhatunk a meglévő számlálók közül, különböző módokon megjeleníthetjük azok adatait, valamint létrehozhatunk saját számlálót, és beállíthatjuk annak értékét. A létrehozott számláló a többivel azonos módon használható, a beállított érték megjeleníthető például a rendszerhez tartozó System Monitor (perfmon.exe) programmal is.

Kategóriák, számlálók és példányok

A teljesítményszámlálók a számítógép különböző teljesítményobjektumainak (processzor, memória, lemezegység stb.) adatait gyűjtik. Egy kategóriához tartoznak az azonos teljesítményobjektumot figyelő számlálók. A .NET PerformanceCounter objektumának létrehozásakor elsőként azt a kategóriát kell megadnunk, amelyhez a csatolni kívánt számláló tartozik. A kategóriához tartozó számlálók az adott teljesítményobjektum különféle mutatóit képesek megjeleníteni. Bizonyos esetekben a kategóriákhoz az adott teljesítményobjektum több példánya is tartozhat (de legalább egynek tartoznia kell, hogy a kategória számlálói használhatóak legyenek). Például, ha a számítógépben több lemezegység van, ezek a „Fizikai lemez" kategória példányaiként jelennek meg (sőt van az összes lemezegységre vonatkozó „_total" nevű példány is). A kategória minden példányához használhatjuk a kategóriához tartozó valamennyi számlálót.



A teljesítményszámlálók értékeit megjelenítő mintaprogram

Természetesen vannak olyan kategóriák, amelyekből csak egyetlen példány létezhet, tipikusan ilyen például a „Rendszer" kategória. Olyan kategóriát is találhatunk, amelyhez akár száznál több példány is tartozik (például a „Végrehajtá-

si szál" kategória) Tehát röviden: a kategóriához tartozó számlálók, a kategóriához tartozó teljesítményobjektum példányok adatainak gyűjtését végzik.

Számlálótípusok

A számlálók típusa meghatározza azt, hogy a számláló milyen számítás eredményét adja vissza, ha a számított értéket kérjük el tőle (NextValue() metódus, lásd később). Számos típus létezik, amelyek mindegyike az alábbi öt csoport valamelyikébe sorolható. (A mintaprogram megjeleníti a kiválasztott számláló típusát is.)

- Átlagszámlálók (Average counters)
- Különbségszámlálók (Difference counters)
- Pillanatnyi érték számlálók (Instantaneous counters)
- Százalékszámlálók (Percentage counters)
- Gyakoriság számlálók (Rate counters)

Az összes lehetséges számlálótípust a PerformanceCounter Type enum tartalmazza.

Átlagszámlálók

Az ilyen típusú számlálók az utolsó két mintavétel közötti értékek átlagát adják vissza számított értékként. Két ilyen típus van, az AverageCount64, és az AverageTimer32. Minden ilyen számlálóhoz tartozik egy alapszámláló (AverageBase), amelynek értéke számításkor a nevezőbe kerül, vagyis például a műveletek teljes számát méri.
Tekintsünk egy konkrét példát, hogy könnyebben érthető legyen ez a számlálótípus. A Fizikai lemez kategória Átlagos átvitt adatmennyiség (bájt/átvitel) számlálója Average Count64 típusú. A számláló minden egyes lemezműveletkor az átvitt bájtok számával növekszik, a hozzá tartozó alapszámláló pedig eggyel, vagyis ez a lemezműveletek számát fogja tartalmazni. A számításkor használt képlet tehát a következő: $(N_1–N_0) / (B_1–B_0)$, ahol N_1 és N_0 a számláló két mintavételekor kapott értékek (vagyis különbségük a két mintavétel között átvitt bájtok összes száma), B_1 és B_0 pedig az alapszámlálótól kapott értékek (vagyis különbségük a két mintavétel közötti lemezműveletek száma). Hányadosuk tehát éppen az egy lemezműveletre jutó átlagos bájtszám, a két mintavétel között. Az AverageTimer32 számláló az adott művelet elvégzéséhez átlagosan szükséges időt adja vissza. Ilyen például a Fizikai lemez kategória Átlagos műveleti idő (mp/átvitel) számlálója.

Különbségszámlálók

Eredményül az utolsó és az utolsó előtti minta különbségét, tehát a mért érték változását adják vissza (vagy nullát, ha az eredmény negatív). Az ilyen számlálók típusa CounterDelta32, vagy CounterDelta64. Ide tartozik még az ElapsedTime számlálótípus, amelynek számított értéke (NextValue) a számláló létrehozása óta eltelt időt adja vissza másodpercekben. Jól felhasználható például egy adott folyamat futási hosszának mérésére.

Pillanatnyi érték számlálók

Ezek a számlálók az utolsó mérés eredményét adják vissza, számítást nem végeznek (NumberofItems32, NumberofItems64, NumberofItemsHEX32, NumberofItemsHEX64). Ilyen számláló például a Memória kategória Rendelkezésre álló memória (bájt) számlálója.

Százalékszámlálók

Az ebbe a típusba tartozó számlálók arra szolgálnak, hogy az adott komponens aktív, illetve inaktív állapotban töltött idejét adják vissza százalékos érték formájában.
Ide tartozik például a CounterTimer típus, amely visszaadja a figyelt erőforrás aktív állapotban töltött idejét, a teljes mintavételi időszak százalékában. A felhasznált képlet a következő: $(N_1–N_0) / (D_1–D_0)$, ahol N_1 és N_0 a számláló értékei a két mintavételi időpontban, D_1 és D_0 pedig a két mintavételhez tartozó időpont (a számláló ezt is tárolja).
A CounterTimerInverse típusú számlálók az erőforrás inaktív állapotban töltött idejét adják vissza a teljes mintavételi időszak százalékában. A Timer100NS és Timer100NsInverse típusú számlálók, pontosan megegyeznek az előzőkkel, csak az időt 100ns-os egységekben mérik.
Timer100NsInverse típusú például, a Processzor kategória Processzor kihasználtsága (%) nevű számlálója.
A százalékszámlálók speciális fajtái a több teljesítményobjektumot figyelő számlálók (Multi-Counters). Ezek a százalékszámlálókkal azonos módon viselkednek, a különbség abban áll, hogy MultiBase típusú alapszámláló tartozik hozzájuk, és mivel több teljesítményobjektum aktív idejét mérik (amelyek egymással párhuzamosan is aktív állapotban lehetnek) 100 %-nál nagyobb értéket is visszaadhatnak.
Képzeljük el például, hogy egy több processzoros rendszerben olyan számlálót szeretnénk, amely több folyamat által aktív állapotban eltöltött időt ad vissza, a teljes eltelt idő százalékában. A számláló (amelynek típusa CounterMultiTimer, CounterMultiTimerInverse, CounterMultiTimer100Ns, vagy CounterMultiTimer100NsInverse lehet), méri a folyamatok által aktívan töltött időt. A visszaadott érték az összes aktív időszak, és az eltelt idő hányadosa, osztva a megfigyelt komponensek számával (ezt tartalmazza a MultiBase típusú alapszámláló), tehát az összes komponensre vonatkozó átlagos aktív idő.

Gyakoriság számlálók

Ezek a számlálók valamilyen esemény vagy művelet másodpercenkénti átlagos gyakoriságának mérésére szolgálnak. Ilyenek például a RateOfCountsPerSecond32 és a RateOfCountsPerSecond64 típusok. A visszaadott érték a két mintavétel közötti különbség és az eltelt idő hányadosa. A gyakoriság számlálók speciális változatai a mintavételes számlálók (Sample Counters). A legfontosabb különbség az, hogy a mintavételes számlálók esetében az eltelt időt (ami a nevezőbe kerül), mi magunk (vagyis a programunk) határozhatjuk meg, míg az alapesetben ez csak a két mintavétel között eltelt idő lehet. Az eltelt idő meghatározása a SampleBase típusú alapszámláló beállításával történik.
Ilyen módon tehát olyan számlálót hozhatunk létre, amely visszaadja a másodpercenkénti átlagos műveletek számát, de nem a teljes mintavételi időtartamra, hanem például csak az adott komponens aktív állapotban eltöltött idejére vonatkozóan.

Performance Counter komponensek létrehozása

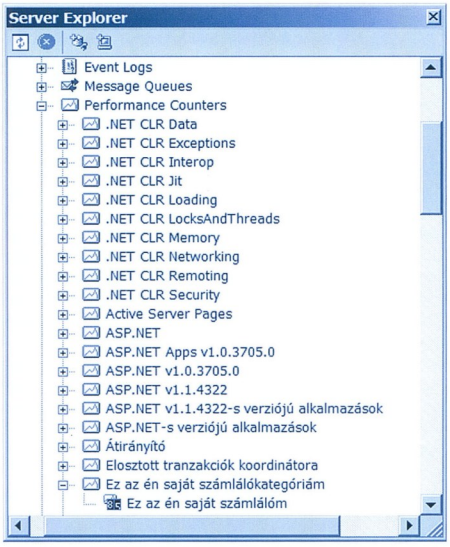
Programjainkból a System.Diagnostic névtérben található PerformanceCounter komponens segítségével férhetünk hozzá a rendszer teljesítményszámlálóihoz. Az objektum létrehozása után (vagy már a konstruktor paraméteréként) meg kell adnunk a számláló kategóriáját (vagyis a megfigyelni kí-

ván teljesítményobjektumot), a számláló, és a példány nevét. A konstruktorban megadott paraméterekkel csak egypéldányos kategória számlálóját hozhatjuk létre. Alapértelmezés szerint az objektum ReadOnly tulajdonsága is be van állítva, a rendszerhez tartozó számlálók esetén nem is változtathatjuk meg ezt az értéket.
Ezzel készen is van a kapcsolódás, kiolvashatjuk, a számláló által tárolt (vagy számított) adatot. Az alábbi példában a „Processor" objektum „_Total" példányához tartozó „%Processor Time" számláló értékét kérjük el.

```
using System.Diagnostic;
PerformanceCounter szamlalo = new
↳ PerformanceCounter();
szamlalo.CategoryName = „Processor";
szamlalo.CounterName = „% Processor Time";
szamlalo.InstanceName = „_Total";
long ertek = szamlalo.RawValue;
```

Teljesítményszámláló létrehozása a Visual Studio-val

Visual Studio-val még egyszerűbb dolgunk van, a Server Explorer-ben megtalálhatjuk az adott számítógép teljesítményszámlálóit, amelyek közül a megfelelőt kiválasztva, egyszerűen ráhúzhatjuk azt projektünkre.
Nagyjából az előzővel megegyező kód generálódik, a MachineName tulajdonság tartalmazza annak a gépnek a nevét, amelyen a kiválasztott számláló található.



Teljesítményszámlálók a Server Explorer-ben

Kategóriák, példányok és számlálók listázása

A következőkben áttekintjük, hogy milyen metódusok segítségével állíthatjuk elő a rendszerben létező kategóriák, példányok és számlálók listáját.

A számlálókategóriákat a PerformanceCounterCategory osztály GetCategories() statikus metódusa adja vissza, PerformanceCounterCategory objektumokból álló tömb képében. Az alábbi metódus a tömb elemeinek CategoryName tulajdonságával tölti fel a Categories nevű listadobozt.

```
using System.Diagnostic;
private void GetCategoriesList() {
   PerformanceCounterCategory[] Cat;
   this.Categories.Items.Clear();
   Cat =PerformanceCounterCategory.GetCategories();
   for (int i = 0; i < Cat.Length; i++) {
      this.Categories.Items.Add(Cat[i].
      ↳ CategoryName);
   }
   this.Categories.SelectedIndex=0;
}
```

A kategória példányainak lekérdezéséhez már nem használhatunk statikus metódust: létre kell hoznunk a kiválasztott kategóriához tartozó PerformanceCounterCategory osztályú objektumot. A konstruktor paramétereként a kategória nevét kell megadnunk. A példányok listáját az objektum GetInstanceNames() metódusa adja vissza sztringekből álló tömb formájában. Ha csak egy példány létezik, üres tömböt kapunk vissza, ezt az esetet külön kell kezelnünk. Több példány esetén, a tömb elemeit hozzáadjuk az Instances nevű listadobozhoz.

```
private void GetInstancesList(String CategoryName){
   String[] Inst;
   try {
      PerformanceCounterCategory Cat =
      ↳new PerformanceCounterCategory(CategoryName);
      this.Instances.Items.Clear();
      Inst = Cat.GetInstanceNames();
      if (Inst.Length == 0)
         this.Instances.Items.Add(„Csak egy példány
         ↳ létezik");
      else
         for (int i = 0; i < Inst.Length; i++)
         ↳{this.Instances.Items.Add(Inst[i].
         ↳ToString());}
      this.Instances.SelectedIndex=0;
      }
   catch (Exception e){MessageBox.Show(e.Message);}
}
```

A számlálók lekérdezéséhez a kategória és a példány nevére is szükségünk van. A kategória nevét a PerformanceCounterCategory konstruktora kapja, a példány nevét pedig, az így létrehozott objektum GetCounters() metódusának kell paraméterül adnunk. A GetCounters() metódus PerformanceCounter objektumokból álló tömböt ad vissza, a Counters listadobozba a tömb elemeinek CounterName tulajdonságát töltjük.

```
private void GetCountersList(String CategoryName,
↳ string InstanceName){
   PerformanceCounter[] Count;
   try {
      PerformanceCounterCategory Cat = new
      ↳PerformanceCounterCategory(CategoryName);
      this.Counters.Items.Clear();
      Count = Cat.GetCounters(InstanceName);
      for (int i = 0; i < Count.Length; i++) {
      this.Counters.Items.Add(Count[i].
      ↳CounterName);}
```

```
        this.Counters.SelectedIndex=0;
    }
    catch (Exception e){MessageBox.Show(e.Message);}
}
```

A számlálók által visszaadott érték

Miután mindhárom listából kiválasztottuk a megfelelő elemet, létrehozhatjuk a teljesítményszámlálóhoz csatolt Performance Counter objektumot, és megjeleníthetjük annak értékét. A számláló értékének visszaadására egy tulajdonság és két metódus szolgál, amelyek a számláló által tárolt adatot három különböző módon jelenítik meg:

- Nyers adat (RawValue) – A RawValue tulajdonság a számláló pillanatnyi, feldolgozatlan értékét adja vissza. Hogy ez jó-e nekünk, vagy nem, az a számláló típusától függ. Ha a számláló valamilyen mennyiség pillanatnyi értékének visszaadására szolgál (átvitt bájtok teljes száma, folyamatok, szálak száma stb.), akkor nem is lehet másra szükség. A nyers adat kiolvasása igen rövid idő alatt lezajlik, mivel semmiféle számításra nincs szükség. Bonyolultabb típusok esetén azonban (átlag, százalék stb.) a RawValue tulajdonság által visszaadott érték önmagában gyakorlatilag semmire sem használható.
- Számított érték (NextValue) – A legtöbb számláló a mért adatokon különféle számításokat is végez (hogy pontosan milyet, az a számláló típusától függ). A számítás eredményét a NextValue() metódus adja vissza. Ez a metódus már minden számlálótípus esetén értelmes eredményt ad, de ha a hívások pillanatában a mért érték jelentősen eltér a szokásostól, akkor félrevezető eredményeket is kaphatunk. A számítás úgy történik, hogy a metódus a következőkben szereplő NextSample() hívás segítségével elkéri a számláló adatait, majd a CounterSample osztály Calculate metódusának segítségével összehasonlítja az aktuális mintát az előző futásakor elmentett CounterSample objektummal. A számításhoz tehát a metódus az előző hívásakor kapott értéket is felhasználja, így az első NextValue() hívás mindig nullát ad vissza. A NextValue() metódus kódjának segítségével könnyen megérthetjük a NextValue() és a NextSample() közötti kapcsolatot és különbséget.

```
public float NextValue() {
   CounterSample sample1 = this.NextSample();
   float single1 = 0f;
   single1 = CounterSample.Calculate
   ↳(this.oldSample, sample1);
   this.oldSample = sample1;
   return single1;
}
```

- Mintavétel (NextSample) – A NextSample() metódus visszatérési értékeként kapott CounterSample objektum mezői a számláló pillanatnyi értékének minden fontos jellemzőjét tartalmazzák. A lekért vagy elmentett minták alapján különféle számításokat végezhetünk, vagy felhasználhatjuk a CounterSample osztály statikus Calculate() metódusát, amely két CounterSample objektumot kap paraméterül, és így az általunk megadott minták alapján végzi el ugyanazt a számítást, amely a NextValue() eredményét is adja. A NextSample() metódus segítségével akár két különböző számlálótól (csak, ha a típusuk azonos) kapott mintákat is összevethetünk.

A mintaprogram következő metódusa, a felhasználó választása szerint a háromféle érték egyikét jeleníti meg. (A NextValue() csak a második hívásra ad nullától különböző értéket.)

```
private void Kiir_Click(object sender,
↳System.EventArgs e) {
   try {
      PC.CategoryName=Categories.
      ↳SelectedItem.ToString();
      PC.CounterName=Counters.SelectedItem.
      ↳ToString();
      if (Instances.Items.Count > 1)
         PC.InstanceName=Instances.SelectedItem.
         ↳ToString();
      else PC.InstanceName="_";
      if (Raw.Checked)
         display.Text=PC.RawValue.ToString("f");
      if (Calc.Checked)
         display.Text=PC.NextValue().ToString("f");
      if (Sample.Checked)
         /display.Text=GetSampleValue();
      CounterType.Text=PC.CounterType.
      ↳ToString();
      display.Visible=true;
   }
   catch (Exception ex){MessageBox.
   ↳Show(ex.Message);}
}
```

A mintavételt a következő metódus végzi, méghozzá úgy, hogy az első mintavétel után egy másodperccel következik a második, a Calculate() metódus az így kapott objektumokat hasonlítja össze. A PC nevű PerformanceCounter objektumot már korábban létrehoztuk.

```
private string GetSampleValue(){
   CounterSample sample;
   float result;
   sample = PC.NextSample();
   System.Threading.Thread.Sleep(1000);
   result = CounterSample.Calculate(sample,
↳PC.NextSample());
   return result.ToString("f");
}
```

Saját teljesítményszámláló létrehozása

A .NET lehetőséget ad saját számlálókategóriák és számlálók létrehozására is. A létrehozott számláló adatai a registrybe kerülnek, ami két fontos következménnyel jár.

- A számláló létrehozásához rendszergazda jogosultság szükséges (vagy legalábbis írási jog a megfelelő registry területekhez).
- A program bezárásával az új számláló nem tűnik el a rendszerből (a System Monitor-ból például továbbra is elérhető), ha meg akarjuk szüntetni, a megfelelő metódus meghívásával nekünk magunknak kell ezt megtennünk (lásd később).

A teljesítményszámláló létrehozása két részből áll: elsőként létre kell hoznunk a rendszerszintű kategóriát, és benne a számlálókat, majd a hozzá csatlakozó PerformanceCounter

objektumot, amellyel beállíthatjuk az új számláló értékét. Általában ezt a két feladatot két önálló programrész végzi, a számlálók létrehozását például az alkalmazás telepítőprogramja, az értékek frissítését pedig az alkalmazás maga.
A mintaprogram alábbi metódusa új kategóriát (ha még nem létezik ezen a néven kategória), és azon belül egy új NumberOfItems64 típusú számlálót hoz létre. Elsőként egy CounterCreationData objektumot hozunk létre (ha több számlálót szeretnénk, akkor többet), és beállítjuk a számláló tulajdonságait. Az így beállított objektumokat a CounterCreation DataCollection-höz kell hozzáadnunk, ami a Performance CounterCategory osztály statikus Create() metódusának harmadik paramétere lesz. Első paraméterként az új kategória nevét, másodikként pedig, a leírását kell megadnunk. Ezután még létrehozzuk a kategória egyetlen példányát, és beállítjuk a számláló kezdőértékét.

```
private void CreateCustomCounter(string
↳CategoryName, string CounterName){
   try {
   if(!PerformanceCounterCategory.
   ↳Exists(CategoryName)) {
      CounterCreationData sajat = new
      ↳CounterCreationData();
      sajat.CounterName = CounterName;
      sajat.CounterHelp = „.NET PerfMon
      ↳mintaszámláló";
      sajat.CounterType =
      ↳PerformanceCounterType.NumberOfItems64;
      CounterCreationDataCollection ccdc = new
      ↳CounterCreationDataCollection();
      ccdc.Add(sajat);
      PerformanceCounterCategory.Create(CategoryName
      ↳,".NET PerfMon mintakategória",ccdc);
      PerformanceCounter Sajatszamlalo = new

↳PerformanceCounter(CategoryName,CounterName,
      ↳"„,false);
      Sajatszamlalo.RawValue=0;
      MessageBox.Show(„A számláló létrejött. Neve:
      ↳"+CategoryName,"PerfMon");
   }
   else
      MessageBox.Show(„A számláló márlétezik!",
      ↳"PerfMon",MessageBoxButtons.OK,
      ↳MessageBoxIcon.Warning);
   }
   catch (Exception e){MessageBox.Show(e.Message);}
}
```

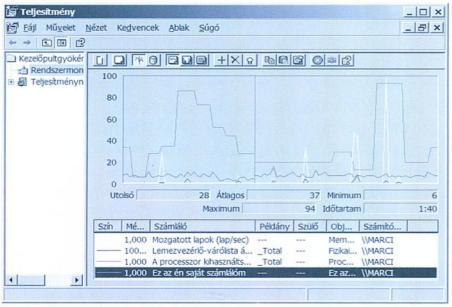
Fontos megjegyezni, hogy nincsen arra lehetőség, hogy már létező kategóriához új számlálót adjunk hozzá (vagy töröljünk belőle). A létrehozás és törlés legkisebb egysége a kategória, teljes tartalmával együtt.

Teljesítményszámláló értékének beállítása

Hogy beállíthassuk a számláló értékét, a hozzá csatlakozó PerformanceCounter komponens ReadOnly tulajdonságát false-ra kell állítanunk. Az alábbi kódrészletben ezt már a konstruktor negyedik paraméterében megtesszük, ezután a RawValue tulajdonság értékét már nem csak lekérni, hanem tetszés szerint beállítani is tudjuk.

```
SetCustomCounterValue(string ↳CategoryName,
string CounterName){
   try {
   if(PerformanceCounterCategory.Exists
   ↳(CategoryName)) {
      PerformanceCounter Sajatszamlalo = new
   ↳PerformanceCounter(CategoryName,CounterName,
      ↳"„, false);
      Sajatszamlalo.RawValue=Decimal.ToInt64
      ↳(CounterValue.Value);
   }
   else
      MessageBox.Show(„Nem létezik a számláló!",
      ↳"PerfMon",MessageBoxButtons.OK,
      ↳MessageBoxIcon.Warning);
   }
   catch (Exception
e){MessageBox.Show(e.Message);}
}
```

A beállított számlálóérték ezután a System Monitor programmal is megjeleníthető.



Saját számláló megjelenítése a System Monitorban

Számláló törlése

A számlálókategóriák (és a bennük lévő összes számláló) törlésére a PerformanceCounterCategory osztály statikus Delete() metódusa szolgál. Az alábbi kódrészletben először ellenőrizzük, hogy létezik-e a törölni kívánt kategória, majd a Delete() metódus paramétereként megadjuk a kategória nevét.

```
private void DeleteCustomCounter(string
↳CategoryName){
   try {
   if(PerformanceCounterCategory.Exists
   ↳(CategoryName)){
PerformanceCounterCategory.Delete(CategoryName);
   MessageBox.Show(„A számláló
   ↳törölve!","PerfMon");
   }
   else MessageBox.Show(„Nincs ilyen számláló",
   ↳"PerfMon",MessageBoxButtons.OK,
   ↳MessageBoxIcon.Warning);
   }
catch (Exception
e){MessageBox.Show(e.Message);}
}
```


Szerényi László
szerenyi.l@met.hu

A cikkben szereplő URL:

[1] http://store.netacademia.net/mshu/OTHER/technet_code/PerfMon.zip

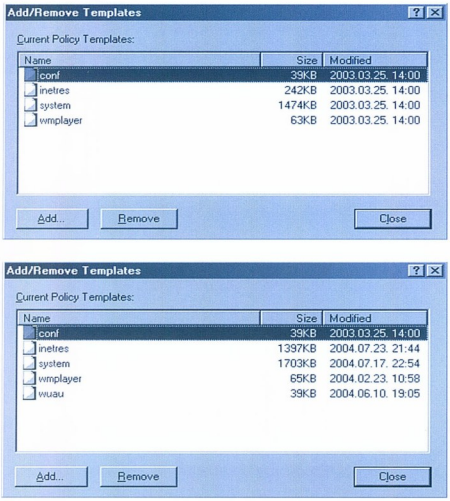
Windows XP SP2

Csoportházirend újdonságok

Az új szervizcsomag révén az XP sokat változott, mert a szándékosan szigorú megoldások beépítése mellett új lehetőségek is rendelkezésre állnak. Ezek a változások akkor érvényesülnek sikeresen tartományi környezetben, ha az üzemeltetést segítő támogatással is rendelkeznek. Szemezgessünk tehát: milyen változások történtek a Csoportházirendben?

609

Szám szerint ennyi új beállításunk lesz a Csoportházirendben, ha „ráhúzzuk" az SP2-es sablonokat. Eléggé hihetetlen érték ez, különösen, ha abból indulunk ki, hogy a Windows 2000 Server RTM változatában is kb. ennyi volt - összesen. Megtehetjük azt is, hogy a Windows Server 2003 gyári sablonjainak a lemezen elfoglalt méretével hasonlítjuk össze, akkor az arány kb. 1,75 MB / 3 MB, azaz valamivel több, mint 3 MB helyet foglal el a Sysvol megosztásban az új sablonokkal felülírt, kiegészített sablonkönyvtár. Ha jól megnézzük a lenti képeket, kiemelkedik az inetres.adm sablon, amely kb. hatszorosára nőtt. Persze rögtön érthetővé válik ez a növekedés, ha tudjuk, hogy ebből a sablonból táplálkozik maga az Internet Explorer is.



A Windows Server 2003-as és az XP SP2-vel frissített sablonok méretei

A méretnövekedéshez még egy – fontosnak is számítható – megjegyzést tennék. Ha figyelmesen megtekintjük a Sysvol megosztást, azaz az itt található mappákat:

```
%SYSTEMROOT%\sysvol\tartománynév\Policies\
```

akkor észrevehetjük, hogy több is van, többféle méretben, akár komolynak is vehető helyfoglalással, ami esetlegesen kihathat a Sysvol megosztás replikációjára is. Viszont használhatjuk a Csoportházirend két opcióját a kordában tartásra. Ezek pedig a következőek és az alábbi helyen találhatóak:

- *„Always use local ADM files for Group Policy Editor"*
- *„Turn off automatic update of ADM files".*

```
Computer Configuration\Administrative Templates\
System\Group Policy
```

Ha az elsőt bekapcsoljuk, akkor a helyi Administrative Templates állományok nem másolódnak be automatikusan a Sysvol megosztásba, ha a másodikat is, akkor a %Systemroot%\inf mappában található sablonok tartalma alapján mutatja a rendszer a Csoportházirend beállításokat. Mindkét parancs hordoz magában figyelemreméltó veszélyeket, ezért mielőtt alkalmazzuk ezeket, olvassuk el a fontos információkat, **[1]** a mellékhatásokról illetve a két beállítás variálásáról.
Ahhoz, hogy visszatérhessünk a lényegre, azaz az újításokra, bővítések megtekintésére a Csoportházirend sablonokkal kapcsolatban, még el kell hárítanunk egy az előző számban **[2]** már részletesen ismertetett problémát, amely az új, SP2-es sablonokkal és az ezekkel nem kompatibilis szerkesztőkkel kapcsolatos.

Windows Firewall

```
Computer Configuration\Administrative Templates\
Network\Network Connections\Windows Firewall
```

Mivel az új tűzfal alapból bekapcsolt állapotban van, ez azt jelenti, hogy egy tartományon belül (is) minden hálózati forgalmat, alkalmazást és szolgáltatást blokkol, ezért duplán ér-

demes odafigyelnünk rá. (*Kis kitérő: a jelenleg még béta állapotban lévő Windows Server 2003 SP1-ben viszont kb. ugyanez a tűzfal, frissítés esetén ki lesz kapcsolva, kivéve az új telepítést (slipstream), de ennél is csak addig, amíg automatikusan letölti a szükséges frissítéseket.*) A ki- és bekapcsolástól kezdve a port kivételekig az összes beállítása kezelhető a Csoportházirendből, ám van egy érdekes újdonsága is, a két profil beállításának lehetősége. Rendelkezésünkre áll egy tartományi profil és egy standard, amely a tartományba nem kötött számítógépekre hat (szintén az összes WF beállítással), azaz pl. arra a laptopra, amelyet esténként hazaviszünk. Ezt a lehetőséget nyilván úgy kell használnunk, hogy az elvileg központilag, erősen védett tartományunkban többet engedünk meg (főleg, mert sokszor muszáj is), míg az otthoni, általában sebezhetőbb körülmények közötti használatra egy agresszívebb beállítást alkalmazunk.

Tesszük ezt azért is, hogy előzetesen elkerüljük azt a közismert szituációt, hogy egy-egy esetleges problémát (vírust, férget, ads programot, stb.) behoz a tartományba a felhasználó. *Újabb kis kitérő: ha megvalósul a VPN karantén mintájára a „LAN karantén" is (van erre már kezdeményezés, ez lesz/lehet a Windows Server 2003 R2-be kerülő NAP, azaz Network Access Protection* ***[3]****), akkor ez a problémahalmaz is csökkeni fog.*

Felmerülhet az a kérdés, hogyan dől el, hogy most tartományban van az adott laptop (mert mondjuk nem kapcsolom ki, vagy hibernálom mielőtt hazaviszem) vagy nem? A következők az analízis szempontjai általában gép bekapcsolásakor illetve a pl. a feléledésekor:

- Tartományi tag egyáltalán a gép? Ha nem: Standard Profil.
- Ha a legutoljára kapott csoportházirend frissítéshez tartozó DNS utótag (suffix) nem egyezik meg a jelenleg is aktív hálózati kapcsolatok közül egyikkel sem, akkor: Standard profil.
- Ha a legutoljára kapott csoportházirend frissítéshez tartozó DNS utótag (suffix) megegyezik a jelenleg is aktív hálózati kapcsolatok közül bármelyiknél szereplő utótaggal _és_ ez a kapcsolat nem betárcsázós ill. VPN akkor: Domain profil.

Van viszont egyetlen, profiloktól független kényelmi beállítás, amely alapértelmezés szerint ki van kapcsolva („*Allow authenticated IPSec bypass*"). Ha bekapcsoljuk, akkor engedélyezzük az IPSec forgalmat a tűzfal hatásainak kikerülésével, de persze a tűzfal továbbra is odahat ezeken a gépeken is, bármilyen más forgalomra. Ez az opció logikus, hiszen ha egy jól működő IPSec infrastruktúránk van, akkor valóban nincs szükség ezt a forgalmat fékezni a tűzfal vizsgálataival. Érdekes még az engedélyezése, hiszen egy felettébb speciális módon kell megadnunk az ebbe a körbe tartozó számítógépeket ill. számítógép-csoportokat, mégpedig az ún. *Security Descriptor Definition Language* (SDDL) sztringekkel. Az alábbi példa egyetlen, a zárójelben definiált csoport megadását eredményezi, ha a végére bejegyezzük persze a csoport SID-jét is.

```
O:DAG:DAD:(A;;RCGW;;;SID)
```

Helyszűke illetve offtopic volta miatt az SDDL-ről a következő oldalakon tájékozódhatunk bővebben **[4]** **[5]**.

WF és ICF

Olyan környezetben, ahol egyaránt rendelkezünk az XP RTM, SP1 és SP2-es verziójú kliensekkel, a kétfajta tűzfallal kapcsolatban figyelembe kell vennünk néhány körülményt. A nem SP2-es gépek esetén az egyetlen beállítás, amivel a Csoportházirenden keresztül befolyásolhatjuk az ICF-et, az a „*Prohibit use of Internet Connection Firewall on your DNS domain network*" opció, ami a következő helyen található:

```
Computer Configuration/Administrative Templates/
Network/Network Connections
```

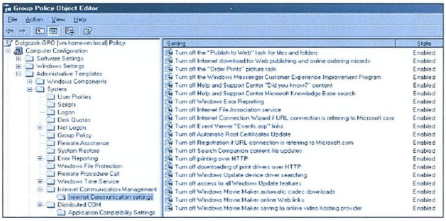
Ha ezzel letiltjuk az ICF-et, és történetesen ez a GPO hatással van az SP2-es kliensekre is (mert mondjuk azonos OU-ban vannak), akkor a következő két szituáció képzelhető el:

1. Ha nem konfiguráljuk a WF alapértelmezett beállításait, akkor a WF is le lesz tiltva.
2. Ha már hozzányúltunk WF beállításaihoz (azaz engedélyeztük a „*Protect all network connections setting*" opciót), akkor üzemszerűen működni is fog.

Internet Communication Management

```
Administrative Templates\System\Internet
Communication Management
```

Jónéhány beállítás került ebbe a teljesen új ágba, amely mind a Computer mind a User szakaszban megtalálható (tegyük azért hozzá, hogy ezek mind-mind csak az SP2-vel kompatibilisek). A névnek megfelelően, ezek a beállítások a számítógépek és a felhasználók számára a Windows komponensekkel kapcsolatos internetes szolgáltatások igénybevételét ill. közlését tilthatják meg.



Bekapcsoltuk a főkapcsolót...

A képen nem látszik, de ennek a szakasznak a gyökerében van egy külön opció („*Restrict Internet communication*"), amely gyakorlatilag egy főkapcsoló. Amíg ezt nem kapcsoljuk be, a többi beállítás sem érvényesülhet. De legyünk óvatosak, mert ha beélesítjük, akkor ezzel az egy lépéssel egyúttal az összes felsorolt korlátozást engedélyeztük. Azaz, ebben az esetben, ha szeretnénk mégis kivételt tenni valamelyik beállítással, akkor annak hatását le kell majd tiltanunk. Nézzünk meg ezek közül a tiltások egyet-kettőt:

Controlling Printing over HTTP

A http-n keresztüli (akár internetes, akár intranetes) nyomtatás ellenőrzését kézbevehetjük innentől, azaz két lehetőségünk is van erre:

- „*Turn off printing over http*", azaz a nyomtatás letiltása,
- „*Turn off downloading of print drivers over http*", azaz a nyomtató meghajtóprogramok letöltésének megtiltása.

Turn off access to all Windows Update features
Az összes Windows Update-tel kapcsolatos szolgáltatást letilthatjuk, azaz blokkolható a hozzáférés a WU weboldalakhoz, akár a Start menüben található ikonon, akár az IE Tools menüjén keresztül. Ezen kívül az Automatic Updates kliens működésének és a Device Manager-ből a driverek Windows Update-n keresztüli frissítésének is vége szakadhat, ha beállítjuk.

Online Print, Web Publishing, Add Network Place Wizards
A Windows Explorerben rendelkezésünkre áll néhány speciális varázsló, amelyek különböző online szolgáltatásokat engedélyeznek, miközben különböző internetes helyekre kapcsolódnak a háttérben. Amikor elindítunk egy ilyen varázslót, a működéséhez szükséges URL-eket általában a registry-ből szerzi meg, vagy pedig a Microsoft weboldalairól. Valószínűnek tartom, hogy ezen szolgáltatások egyik közös tulajdonsága az, hogy egy paranoiás rendszergazda szívesen eltávolítaná ezeket a felhasználók elől. Innentől megteheti, külön-külön csak a számítógépekre vagy a felhasználókra is érvényesítve.

Automatic Updates

```
Computer Configuration\Administrative Templates\
Windows Components\Windows Update
```

Az SP2 sablonjaival bekerülnek a Csoportházirendbe a SUS szerver és az AU kliens működéséhez szükséges sablonok is (wuau.adm). Ezen kívül két új opció is megjelenik, ami viszont csak az SP2-vel ellátott számítógépekre hatásos. Ezek a „*Install Updates and Shut Down*" művelet körülményeire vonatkoznak, ami a frissítések telepítését jelenti, közvetlenül a gép leállítása előtt. Ez a parancs a Leállítás menüben jelenhet meg, és a „*Do not adjust default option in 'Install Updates and Shut Down' option in Shut Down Windows dialog box*" opció értékétől függ az hogy ez lesz-e az alapértelmezett ebben a menüben vagy sem. A másik beállítás a „*Do not display 'Install Updates and Shut Down' option in Shut Down Windows dialog box*", amely a menüpont láthatóságát korlátozza.

Terminal Services

A TS szemszögéből sem maradt érintetlenül a Csoportházirend. A következő két helyen:

```
Computer ill. User Configuration\Administrative
Templates\ Windows Components\Terminal Services
Client
```

egyaránt megtilthatjuk a biztonságosság érdekében a felhasználóknak, hogy a kliensen lementsék a jelszavukat („*Do not allow password to be saved*").

User Profiles

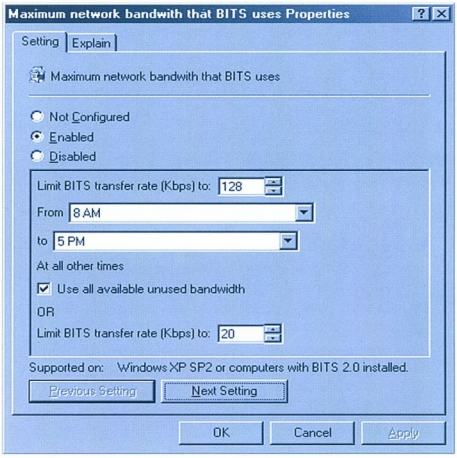
```
Computer Configuration\Administrative Templates\
System\User Profiles
```

Alapértelmezés szerint, ha törlünk egy vándorló profilt, akkor az összes beállítás, pl. a szoftvertelepítésekkel kapcsolatosak is elszállnak. Azonban ha a „*Leave Windows Installer and Group Policy Software Installation Data*" opciót engedélyezzük, akkor nem lesz szükség a felhasználó következő belépésekor a hozzá rendelt alkalmazások telepítésének megismétlésére. Ha mégis az a célunk, hogy teljes körűen eltávolítsuk a profilt, akkor lokális rendszergazdaként belépve kell törölnünk a registry és a filerendszer profilhoz kapcsolódó részeit.

Hálózat

```
Computer Configuration\Administrative Templates\
Network\Background Intelligent Transfer Service
```

- BITS: A Background Intelligent Transfer Service egy kicsit érdemtelenül elhanyagolt terület, hiszen eddig nem lehetett a Csoportházirendből központilag szabályozni, sőt az adott gépen még mindig csak a kicsit nehézkes bitsadmin.exe az egyetlen (parancssori, GUI-n futó nincs is) eszköz erre a feladatra. Az SP2-vel kicsit jobb a helyzet, hiszen bekerült két új sablon is. Egyik ezek közül a „*Maximum network bandwith that BITS*", amellyel szükséges esetben limitálni lehet a BITS 2.0 (az SP2-ben már ez a verzió van) által használható sávszélességet. De ennél többet rejt ez az opció, hiszen intervallumot és konkrét sávszélességet is beállíthatunk, sőt azt is, hogy a fennmaradó időben mennyivel repesszenek a BITS által vezérelt letöltések.



BITS optimalizálás

A másik opció („*Timeout (days) for inactive jobs*") segítségével az általunk beállított nap értékig még életben marad a letöltési kérés, csak ezután törlődik.

GÁL TAMÁS
MCSE, MCSA, MVP (Software Distribution)
gtamas@tjszki.hu

A cikkben szereplő URL-ek:

[1] http://tinyurl.com/5lbzr
[2] http://tinyurl.com/5zk9u
[3] http://tinyurl.com/4n7oo
[4] http://tinyurl.com/6sbv6
[5] http://tinyurl.com/5l7wh

ASP.NET 2.0 (Whidbey)

Mi várható a 2005. évi ASP.NET-ben?

III. rész: Personalization és Membership API – testreszabható weblapokhoz

Miért vásárolok az „amazonon"? Azért, mert mindig ajánl nekem újabb és újabb könyveket, amelyek tényleg érdekesek a számomra. Figyeli miket vásároltam, mások miket vásárolnak, figyeli a szokásaimat és ezek alapján nagy valószínűséggel megtippeli, mi érdekelhet. A weboldal teljesen az én ízlésemnek megfelelő adatokat szolgáltat nekem, és ugyanezt teszi másoknak is. A titok kulcsa: Personalization, azaz személyes ízlésre szabás. A sorozat mostani részében megnézzük, hogyan lehet ezt megoldani az ASP.NET 2.0 eszközeivel.

Az állapottárolás problematikája

A web- és asztali alkalmazások közötti egyik legnagyobb működésbeli különbség az állapotkezelés módja. Egy normál asztali alkalmazás, mint pl. a WinWord, amíg fut, bármilyen beállítást, információt tud tárolni a memóriában. Ezzel szemben egy webalkalmazásban a weblap mögötti osztály egy példánya csak a másodperc tört részéig létezik, aztán megsemmisül. Emiatt mezőkben, tagváltozókban nem tárolhatunk tartós információkat, akár csak néhány perces időtartamra sem.

Természetesen minden webfejlesztési keretrendszer biztosít egy általában Sessionnek nevezett tároló alkalmatosságot, ami pár percig képes tárolni a webes felhasználók adatait. A Session azonban rövid idejű tárolásra van kitalálva, amit nem tudok eléggé hangsúlyozni! Láttam már olyan céget, ahol felvették a Session Timeout értékét 2 napra, mert így nem kellett újra belépni a felhasználóknak. Természetesen az adatbáziskapcsolati objektumok is Sessionben voltak letárolva, így legalább volt dolga a rengeteg RAM-nak az SQL Serverben.

A Session nem hosszú idejű adattárolásra való, pont. De akkor hogyan emlékszik az Amazon vagy a tudástár **[1]** rám egy hónap múlva? Cookie segítségével. Ok, de a kiszolgálón nem elég felismerni a böngésző által küldött Cookie-t, valahol le is kell tárolni két látogatás közben a számomra hasznos adatokat. Akár hónapokig is. Erre találták ki az adatbázisokat. Csakhogy mi a programunkban általában objektumokban gondolkodunk, míg az adatbázisok táblákban. Érdekel engem, hogyan tárolódik le a megőrzendő memóriabeli struktúrám az adatbázisban? Ha igen, akkor érdemes elmélyedni a Persistence Frameworkök működésében, érdekes kaland **[2]**. Ha nem, akkor a kedves olvasó szeretni fogja az ASP.NET 2.0 Personalizationt.

Personalization – Profile funkció

A Personalization keretrendszer célja, hogy a webalkalmazás látogatói részére tartósan fontos adatokat *típusosan* és a *háttéradatbázis elfedésével* tárolja el.

Típusos, azaz nem olyan, mint pl. a Session, ahol minden kivett adatot vissza kell konvertálni a tényleges típussá, az elfedés pedig azt takarja, hogy ha nem akarunk vesződni a háttéradatbázis sémájának kialakításával és a típusok elmentésével-visszatöltésével, akkor a Personalization leveheti ezeket a terheket a vállunkról.

Konkrétabban, a Personalization User Profile szolgáltatása segítségével tetszőleges számú jellemzőt tárolhatunk el a felhasználóhoz rendelten, még anonymous felhasználók esetén is.

A tárolandó jellemzőket a web.configban írhatjuk elő:

```
<system.web>
  <profile>
    <properties>
      <add name="BgColor"
        type="System.Drawing.Color"
        allowAnonymous="true"/>
      <add name="VisitorName"
        defaultValue="ismeretlen vándor"
        allowAnonymous="true"/>
    </properties>
  </profile>
</system.web>
```

Azaz minden látogatóhoz tárolni fogjuk a lap háttérszínének értékét Color típussal, és a látogató nevét, az alapértelmezett string típussal. Nem akarjuk megkövetelni a felhasználóktól, hogy regisztrálják magukat nálunk és mindig belépjenek, ezért engedélyeztük az allowAnonymous="true"-val, hogy a

Personalization API magától egy GUID-ot tartalmazó Cookiet küldjön az ismeretlen felhasználónak, így legközelebb elő tudja venni a számára letárolt beállításokat. A Profile API komplex típusokat is tud tárolni, tömböket, listákat vagy saját típusokat is. Ekkor érdemes elgondolkodni hogyan tároljuk le őket a háttéradatbázisban. Ezt az add elemen belül a serialize-As attribútummal szabályozhatjuk (alap a ProviderSpecific):

```
serializeAs=[„String|Xml|Binary|ProviderSpecific“]
```

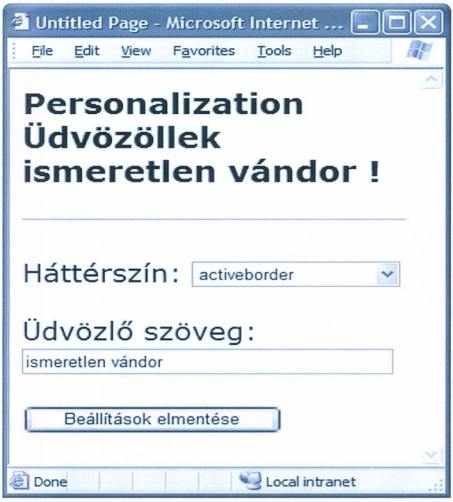
Mivel az ismeretlen látogatók adatainak tárolása jelentős adatbáziskölts égg el járhat, ezért alapban az anonymous personalization nem engedélyezett, de a web.configban a system.web elem alatt könnyen bekapcsolható:

```
<anonymousIdentification enabled="true“ />
```

A Profile adatokat a HttpContext Profile jellemzőjén keresztül érhetjük el, amelyet kényelmi okokból a Page osztályra is kivezettek Profile néven. Ezen keresztül a web.configban definiált adatokat *típusosan* érhetjük el!
Azaz esetünkben

```
visitorMsg.Text = Profile.VisitorName;
Color c = Profile.BgColor;
```

hívásokkal olvashatjuk ki a letárolt (vagy alapértelmezett) adatokat. A VisitorName string típusú, a BgColor pedig Color típusú, nem kell semmiféle konverziót elvégeznünk! Az ASP.NET kódgenerálással típusos Profile osztályt hoz létre a konfigban megadott adatok alapján. Ezért ez sokkal kényelmesebb, mint az 1.1-es keretrendszerben unásig ismételt kasztolás, konverzió.
Ezek után a felhasználó szeretné a saját személyiségére szabni az oldalt.



A Profile funkciót tesztelő lapunk, testreszabás előtt

Kiválaszt egy háttérszínt és beírja a nevét. A gombnyomásra nekünk el kell tároltatni az adatokat a Profile API-n keresztül, illetve vizuálisan azonnal jelezni kell neki az új beállításokat.

```
void SaveSettings()
{
  Profile.VisitorName = userNameTextBox.Text;
  Profile.BgColor = Color.FromName(
    BgColorList.SelectedItem.Value);
  SetUIControls();
}
```

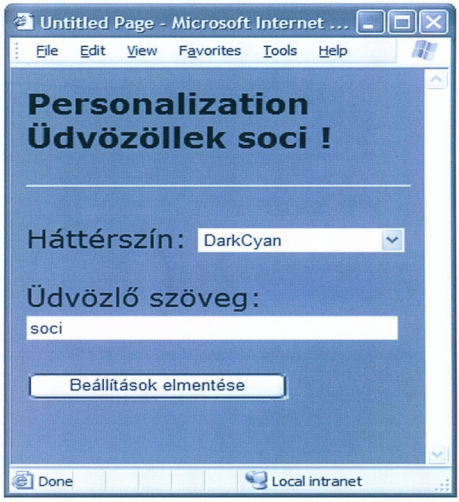
Látható, hogy most is típusos értékeket rakunk el a Profile-ba. A felhasználói felület állítása már nem tartozik szorosan a témánkhoz, de a teljesség kedvéért bemutatom annak kódját is:

```
void SetUIControls()
{
  userNameTextBox.Text = visitorMsg.Text =
    Profile.VisitorName;
  StringCollection colors = GetColorNames();

  BgColorList.DataSource = colors;
  BgColorList.DataBind();

  Color c = Profile.BgColor;

  if (!c.IsEmpty)
  {
    bodri.Attributes[„bgcolor“] =
      c.ToKnownColor().ToString();
    BgColorList.SelectedIndex =
      colors.IndexOf(c.Name);
  }
}
```



A testreszabott oldal

A bodri változó a body elem, annak állítjuk a bgcolor attribútumát:

```
<body runat="server“ id="bodri“>
```

A Profile tárolása

Szép, hogy ennyire automatikus a Profile kezelése, de azért valahol csak letárolódik az információ, amit időnként menteni (bakupolni) kellene, és egyéb karbantartásokat végezni rajta, úgyhogy nézzünk utána, milyen adatbázist használ a Profile API. A Profile általában valamilyen relációs adatbázisban kerül tárolásra. Az adatbázisfüggőség elkerülésére az ASP.NET Profile a ProfileProvider absztrakt alaposztály mint interfészen keresztül tárolja le és éri el a profile információkat. Az interfész így néz ki:

```
public abstract class ProfileProvider
{
  public abstract int DeleteInactiveProfiles(...);
  public abstract int DeleteProfiles(...);
  public abstract ProfileInfoCollection
    FindInactiveProfilesByUserName();
  public abstract ProfileInfoCollection
    FindProfilesByUserName(...);
  public abstract ProfileInfoCollection
    GetAllInactiveProfiles(...);
  public abstract ProfileInfoCollection
    GetAllProfiles(...);
    public abstract int
  GetNumberOfInactiveProfiles(...);
}
```

Az interfész egy klasszikus adatelérő rétegnek néz ki, ezt implementálva valamilyen konkrét adatbázishoz máris részese lehetünk a Profiler API-nak. A Microsoft két implementációt ad számunkra: AccessProfileProvider Access adatbázishoz és SqlProfileProvider SQL Server háttérhez. Ha ezek nem felelnek meg, például Oracle adatbázis támogatásra volna szükség, akkor a fenti interfészt implementálni nem nagy ördöngösség. Az implementációt a web.config segítségével csatolhatjuk be a Profile API vérkeringésébe:

```
<profile defaultProvider="MyProfileProvider" />
```

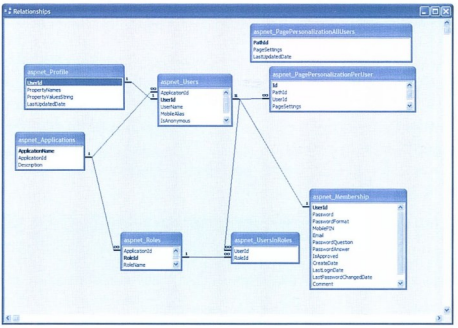
A defaultProvider-t természetesen pontosan specifikálni kell. Például az AspNetSqlProvider (SQL Serverhez) leírása így szerepel a machine.configban:

```
<profile enabled="true"
  defaultProvider="AspNetAccessProvider">
  <providers>
    <add name="AspNetSqlProvider"
      type="System.Web.Profile.SqlProfileProvider,
      System.Web, Version=2.0.3600.0,
      Culture=neutral, PublicKeyToken=..."
      connectionStringName="LocalSqlServer"
      applicationName="/"
      description="Stores and retrieves profile
      data from the local Microsoft SQL Server
      database" />
    <add name="AspNetAccessProvider" ...
  </providers>
</profile>
```

Látható, hogy a ProfileProvider leszármazott assembly qualified neve szerepel a type attribútumban, így a Profile tárolásához vagy betöltéséhez az ASP.NET be tudja tölteni a kívánatos implementációt. Mint korábban látható volt, alapértelmezettként az Access szolgáltató van megadva, vélhetően azért, mert annak használata igényli a legkisebb bemosakodást. Például ha nem is rakunk az alkalmazás mögé semmilyen Access .mdb fájlt, a provider az első használat során létrehoz egy ASPNetDB.mdb adatbázist a DATA nevű alkönyvtárban (feltéve, ha az ASPNET felhasználónak van joga hozzá). Érdekes látni az Access adatbázis reneszánszát, hisz úgy tűnt az MSDE miatt már lassan kihal(asztják), de úgy néz ki az ASP.NET csapat még szereti. Nincs is azzal semmi gond, ha néhány felhasználót kell kiszolgálni, biztos nem olyan nyűgös a telepítése, mint egy MSDE 2000-é. (Az ennek megfelelő SQL 2005 Express telepítési szempontból talán egy könnyebben megülhető ló lesz).

(Dohogás... *A cikket a 2004 nyári kiadású Whidbey Beta 1 alapján írom. A talán november táján kijövő Beta2 esetén már új könyvtárnevek lesznek a speciális célokra, mint az előbb a Data könyvtár. A Data Application_Data lesz, a bin Application_Assemblies, stb. Szeretnek gépelni az ASP.NET fejlesztők, de én nem. Értem én, nem akarják, hogy egy meglévő alkalmazás, ami használja pl. a Data könyvtárnevet, bajban legyen. Addig megtanulok gépelni, vagy ők rájönnek, hogy egyszerűbb lenne az élet, ha maradnának a jó kis rövid nevek, és akinek nem tetszik, az beállíthat magának valami mást. ...dohogás vége.)*

Érdekességképpen mellékelem a Personalization (valamint a következőkben tárgyalandó Membership és Role API) által használt adatbázis nagyvonalú struktúráját.



Personalization, Membership és Role Manager háttéradatbázis szerkezet

Membership API – felépítés

Eddig programunkban a Personalization saját maga azonosította az anonymous felhasználókat Cookie alapján. Sok esetben azonban szeretnénk, ha a felhasználók regisztrálnák magukat a webalkalmazásunkban, így később beléphetnek hozzánk. Ezután bármely számítógépről belépve ugyanazt a környezetet tárhatjuk eléjük. Ehhez viszont nekünk kell megmondanunk a Personalization modulnak, kinek az adatait kell elmenteni és betölteni, valamint nekünk kell tárolni a felhasználó nevét, jelszavát, stb. Ezt már több ezer ASP.NET fejlesztő megírta, itt volt az ideje, hogy maga a platform is adjon hozzá segítséget. Jöhet a Membership API.

A Membership célja a webes felhasználók adatainak kezelése. Egyik nem titkolt célja persze a Personalizationnel való együttműködés, így a Membership által azonosított felhasználó lapjait szabhatjuk testre, azaz Membership célja az authentikáció (hitelesítés), az összes többi adat a Personali-

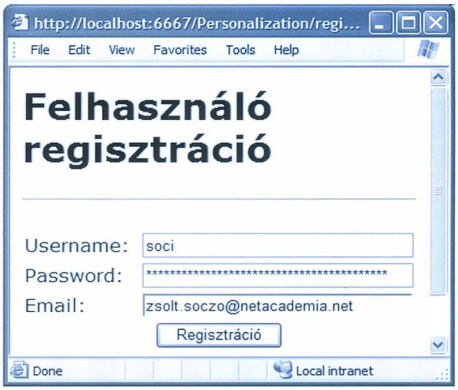
zation feladata. A Membership is a Personalizationnél látott Provider modellre épül, azaz létrehoztak egy absztrakt alaposztályt mint interfészt, ezt implementálják a konkrét adatbázisra szakosodott szolgáltatók, amelyek jellemzőit és típusát a web.configban adhatjuk meg. Esetünkben minden Membership szolgáltató ősе a MembershipProvider osztály. Milyen tipikus funkciókat várunk el az API-tól? Felhasználó regisztrációt, beléptetést és jelszóemlékeztetőt. Személyes adatok kezelése (címek, telefonszámok és hasonlók)? Nem! Ez már a Personalization része, nem a Membershipé. Fókuszáljunk most erre!

Az API funkcióit a MembershipProvider oszály felületén keresztül tekinthetjük át (minden tag publikus és absztrakt, csak a tömörség kedvéért ezek nem szerepelnek a kódban). Látható, hogy az előbb lefektetett funkciónál jóval többet kapunk:

```
public abstract class MembershipProvider {
  bool ChangePassword(string name,
    string oldPassword, string newPassword);
  bool ChangePasswordQuestionAndAnswer(
    string name, string password,
    string newPasswordQuestion,
    string newPasswordAnswer);
  MembershipUser CreateUser(string username,
    string password, string email,
    string passwordQuestion,
    string passwordAnswer,
    bool isApproved,
    out MembershipCreateStatus status);
  bool DeleteUser(string name,
    bool deleteAllRelatedData);
  MembershipUserCollection FindUsersByEmail(
    string emailToMatch, int pageIndex,
    int pageSize, out int totalRecords);
  MembershipUserCollection FindUsersByName(
    string usernameToMatch, int pageIndex,
    int pageSize, out int totalRecords);
  MembershipUserCollection GetAllUsers(
    int pageIndex, int pageSize,
    out int totalRecords);
  int GetNumberOfUsersOnline();
  string GetPassword(string name, string answer);
  MembershipUser GetUser(
    string name, bool userIsOnline);
  string GetUserNameByEmail(string email);
  string ResetPassword(string name,
    string answer);
  void UpdateUser(MembershipUser user);
  bool ValidateUser(string name, string password);
  string ApplicationName { get; set; }
  bool EnablePasswordReset { get; }
  bool EnablePasswordRetrieval { get; }
  bool RequiresQuestionAndAnswer { get; }
}
```

Membership – felhasználó regisztráció

A kiválasztott Membership szolgáltatóval együttműködő legegyszerűbb regisztrációs lap valahogy így festene:



Az email a jelszó-emlékeztető miatt fontos. Gombnyomásra a következő kód menti el a felhasználó adatait a Membership API segítségével:

```
void CreateUser_Click(object sender, EventArgs e)
{
  Membership.CreateUser(
   Username.Text, Password.Text, Email.Text);
}
```

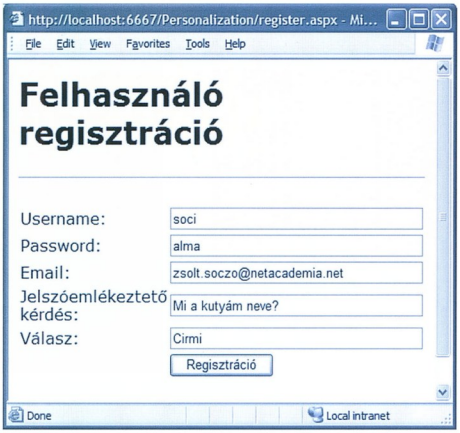
Kódunk azonban hibával ér véget:

```
System.Web.Security.MembershipCreateUserException:
The password-question supplied is invalid.
```

A probléma oka a következő config-részlet kiemelt sora:

```
<membership defaultProvider="AspNetAccessProvider"
  <providers>
    <add name="AspNetAccessProvider"
     type="System.Web.Security.
          AccessMembershipProvider,
     System.Web, Version=2.0.3600.0,
 Culture neutral,
     PublicKeyToken=b03f5f7f11d50a3a"
     connectionStringName="AccessFileName"
     enablePasswordRetrieval="false"
     enablePasswordReset="true"
     requiresQuestionAndAnswer="true"
     applicationName="/"
     requiresUniqueEmail="false"
     passwordFormat="Hashed"
     description="Stores and retrieves membership
     data from the local Microsoft Access
     database file „
    />
  </providers>
</membership>
```

Azaz alapesetben csak akkor tudunk létrehozni felhasználót, ha megadunk egy jelszóemlékeztető mondatot és az arra adott választ (mi a kutyám neve, mazsola). Tegyük ezt, ha már egyszer tudja a rendszer, használjuk ki!



Felhasználó regisztráció jelszóemlékeztetővel

```
Membership.CreateUser(username.Text,
% password.Text, email.Text,
question.Text, answer.Text,
true, out status);
```

Így már sikerrel regisztrálhatom magam.

Membership API – Felhasználó beléptetés

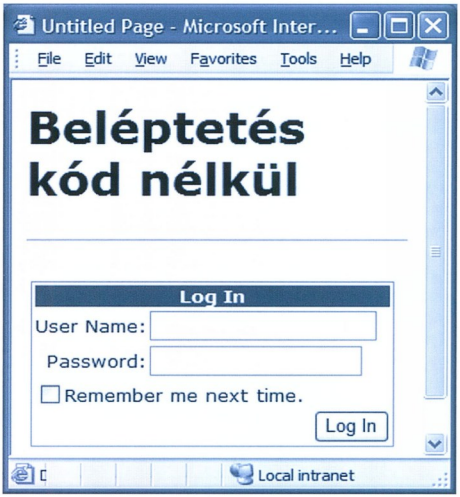
A felhasználó ellenőrzéséhez szokás szerint bekérjük annak felhasználói nevét és jelszavát, majd megnézzük az adatbázisban szerepel-e. Esetünkben az adatbázist a Membership fedi el, így azzal kell kommunikálnunk. Ha az ellenőrzés sikeres, akkor használhatjuk az ASP.NET 1.1-ből már ismert Forms hitelesítést a beléptetéshez:

```
if (Membership.ValidateUser(
    Username.Text, Password.Text))
  FormsAuthentication.RedirectFromLoginPage(
    Username.Text, false);
```

Nem nagy kihívás, igaz-e?
De ennél kevesebb kóddal is megúszhatjuk, pontosan nulla sorral. Az új Login vezérlő elintéz helyettünk mindent:

```
<asp:login id="Login1" runat="server" />
```

Semmilyen kódot nem kell hozzá írni, mivel a Membership API és a FormsAuthentication modul segítségével elintézi azt, amit az előbb én kézzel tettem meg. Ha a felhasználó belépett, akkor Forms hitelesítés a HttpContext.User.Identity.Name-be berakja a korábban látott RedirectFromLoginPage első paraméterében átadott felhasználói nevet. A Personalization a HttpContext.User jellemző alapján dönti el, hogy valakit neki kell anonymous módon követni, vagy egy általunk azonosított felhasználó adatait kell tárolni. Azaz a HttpContext.User közös pont a két API között, így a Personalization automatikusan épít a Membership adataira, ha a FormsAuthentication vagy akár egy saját authentication modulunk azonosította a látogatót.



A Login vezérlő – no code

Még egy megoldandó probléma azért maradt. Tegyük fel, hogy a látogató regisztráció nélkül, anonymousként már beállított magának néhány vizuális jellemzőt a lapon, vagy például vásárolt termékeket a vásárlói kosarába, amelyet mi a Profile API-val letároltunk. Ezek után fizetni szeretne, ehhez viszont már regisztrálni kell magát. Regisztráció után viszont elveszik a vásárlói kosara, mert beléptetés után a Personalization modul már más felhasználóként érzékeli barátunkat. Ha nem mentjük át az adatait anonymous korából, tuti máshol fog vásárolni, mert újra nem fogja összekattogni a kosarát.
Azaz a feladvány az anonymous adatok átmentése a regisztrált felhasználó Profile-jába. Ezt közvetlenül az authentikáció után, még a következő lap betöltése előtt kell megtennünk. A global.asax-ben kapunk erre a célra egy visszahívást a Profile_MigrateAnonymous eseményben, melyben átemeljük az anon felhasználó beállításait a belépett felhasználóéba:

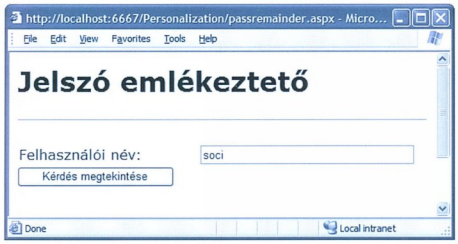
```
void Profile_MigrateAnonymous(Object sender,
  ProfileMigrateEventArgs pe) {
  //Az anonymous profile lekérése
  HttpProfile anonProfile =
    Profile.GetProfile(pe.AnonymousId);

  //Másoljuk ami van
  if (anonProfile.BgColor !=
    System.Drawing.Color.Empty) {
    Profile.BgColor = anonProfile.BgColor;
  }
  if (Profile.VisitorName != null) {
    Profile.VisitorName = anonProfile.VisitorName;
  }
}
```

Membership – jelszóemlékeztető

Ha a jelszó nem visszafejthető módon van tárolva (ez az alapértelmezett és ajánlott), akkor lehetőséget kell adni a felhasználóknak, hogy a regisztrációkor megadott kérdésre helyesen válaszolva új jelszót kapjon.

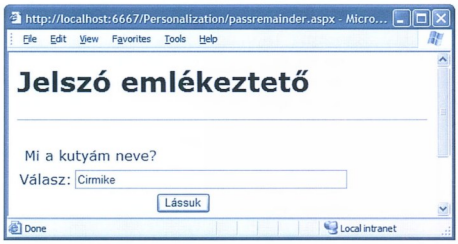
Első lépésben bekérjük a nevét:



A gombra kattintva lekérjük és megjelenítjük a felhasználó trükkös kérdését (a tömörség kedvéért nincsenek ellenőrzések):

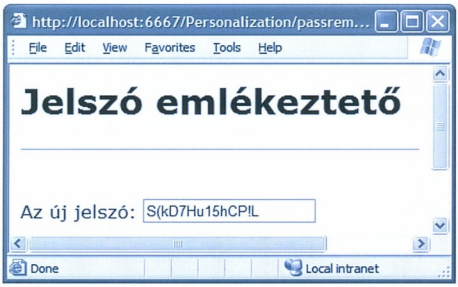
```
void questionButton_Click(object sender,
  EventArgs e) {
  passQuestion.Text =
    Membership.GetUser(
    userName.Text).PasswordQuestion; ...
}
```

Második lépésben válaszolnia kell a kérdésre:

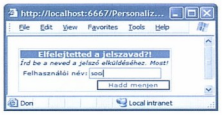


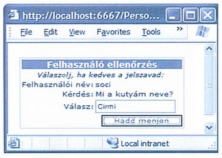
A választ az illetékes Membership Provider ResetPassword metódusával dolgozzuk fel, ami új jelszót generál a felhasználónak:

```
void goRecover_Click(object sender,
  System.EventArgs e) {
  newPassword.Text =
    Membership.Provider.ResetPassword(
    userName.Text, passAnswer.Text);
}
```



Elég erős jelszót kaptunk, ennek megváltoztatásához is írhatnánk felhasználói felületet, amely a Membership.Provider.ChangePassword metódussal meg tudja változtatni a jelszót a régi ismeretében.
Természetesen az egész procedúrát csak azért mutattam meg, hogy megérthessük a jelszavak módosításának hátterét. Valójában ezt a feladatot is meg lehet oldani egy fia sor kód nélkül, a PasswordRecovery vezérlővel:





Ez a vezérlő levélben küldi el a jelszót, amihez a feladót meg kell adni a web.configban:

```
<smtpMail
serverName="localhost"
serverPort="25"
from="passmaster@netacademia.net" />
```

Erre a PasswordRecovery már boldogan küldi emailben a jelszavunkat:

Zárszó

Helyhiány miatt nem mutattam be a Role Manager használatát, amellyel a felhasználókat szerepkörökbe rendezhetjük, és testre szabott tartalmat szolgáltathatunk számukra. A téma iránt bővebben érdeklődőknek ajánlom a **[3]** linket, a példakódok a **[4]** címen találhatók.

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] www.netacademia.net/tudastar/
[2] ambysoft.com/persistenceLayer.html
[3] netacademia.net/training/3201
[4] netacademia.net/tudastar/articlepage.aspx?upid=4108

Tippek & trükkök

Fókuszban a Windows TCP/IP

Egy-egy „trükk" megismerése sokat segíthet napi munkánkban. Ezúttal a Windows TCP/IP témaköréhez adunk tippeket, és folytatjuk az SQL Server 2000-ről szóló tippek korábbi számunkban megkezdett sorozatát.

Az RRAS nélkül szeretném használni az IP forwarding funkciót, de nem találom meg a Windows Server 2003-ban...

A Windows NT 4.0-ben ez még egyszerű volt, az „Advanced TCP/IP" tulajdonságok között megtalálhattuk az „Enable IP Routing" jelölőnégyzetet. Azóta ez kissé macerásabb (ennek nyilván oka van), a több hálózati kártya közötti korlátlan átjáráshoz a regisztrációs adatbázisba kell belenyúlnunk, mégpedig ide:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet
Services\Tcpip\Parameters
```

Itt keressük meg az IPEnableRouter kulcsot, és írjuk át az értékét 1-re. Sajnos az élesítéshez muszáj újraindítani a gépet.

DHCP-szerverrel osztom ki a címeket, de egészen más (169.254.x.x) cím került a kliensre. Miért és honnan jön ez a cím?

Ez az APIPA protokoll automatikus használata miatt van. Ha a DHCP valamilyen okból nem működik, és persze manuálisan sem állítottunk be TCP/IP adatokat, akkor az APIPA (Automatic Private IP Addressing) szolgáltatás szerint a 169.254.x.x-es tartományból kapjuk a TCP/IP konfigurációt. Ez a „B" osztályú IP-tartomány egy a 3330 számú RFC-ben deklarált címtartományok közül, amely arra a célra is jó, hogy a különböző Windows operációs rendszerek (így a Windows Server 2003 is) akkor is automatikusan megtalálják egymást a hálózatban, ha nincs DHCP-szerver. Mivel alapértelmezés szerint a TCP/IP adatok automatikus érkezésére „áll be" a telepítés után minden Windows operációs rendszer, ezért vagy a DCHP-szerver által, vagy ha nincs DHCP, akkor az APIPA segítségével tudunk kommunikálni a klienssel. Amennyiben van DHCP-szolgáltatás, és mégis ilyen címet kapunk, akkor a DHCP-vel vagy a fizikai hálózattal kapcsolatos hibajelenség áll fenn.

Le lehet tiltani az APIPA-t végleg?

Igen. Ha ezt szeretnénk, járjunk el a következő lépések szerint a regisztrációs adatbázisban:

1. Látogassuk meg a következő kulcsot:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet
↳ Services\Tcpip\Parameters\Interfaces
```

2. Válasszuk ki a megfelelő interfészt.
3. Edit menü > New – DWORD value
4. A bejegyzés neve legyen: IPAutoconfigurationEnabled
5. Az értékét pedig állítsuk 0-ra.

Ha történetesen egy menetben az összes hálózati kártyán le akarjuk tiltani ezt a szolgáltatást, akkor hozzuk létre a következő bejegyzést, és az értékét állítsuk nullára:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSe
↳ Services\Tcpip\Parameters\IPAutoconfiguration
↳ Enabled
```

Ha lehúzom a hálózati kábelt a laptopomról (akár csak 3 másodpercre is), akkor IP-cím híján leáll minden szolgáltatás és alkalmazás, ami a hálózattal kapcsolatos. Nem lehetne ezt az automatizmust kiküszöbölni?

Lehetséges. De nézzük meg, miről van szó. Ez a funkció a Windows 2000 Server óta létezik, úgy hívják „media sense", azaz „kapcsolatérzékelés". Ha az operációs rendszer nem látja a „kábelt", akkor tiltással reagál. Az persze nem igaz, hogy minden, ami hálózattal kapcsolatos, leáll, mert csak az adott interfésszel kapcsolatos protokollok/szolgáltatások halnak meg, és például egy másik hálózati kártya vagy egy loopback adapter (127.0.0.1) ilyenkor is működik vígan tovább. Ha viszont mi akkor is szeretnénk például IP-címet használni, amikor nincs csatlakozás, akkor állítsuk a kapcsolatérzékelés tiltását (a példa csak a TCP/IP protokoll „életben hagyására" vonatkozik).

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet
↳ Services\Tcpip\Parameters
```

1. Edit menü > New – DWORD value
2. A bejegyzés neve legyen DisableDHCPMediaSense.
3. Az értéke pedig 1.
4. Újraindítás.

Ez a beállítás akkor is hasznos lehet, ha állandó hálózat nélküli, ám hálózati kártyát tartalmazó gépben szeretnénk például virtuális gépeket hálózati kapcsolattal használni (VMware/VPC).

Mi az az egyetlen tulajdonság amely a Microsoft Client for Networks tulajdonságai között szerepel változatlanul már a Windows NT4 Server óta?

Nos, ezen a panelen az RPC szerviz névfeloldási típusát választhatjuk ki. Az alapértelmezett (és alapos ok nélkül nem is célszerű ezt megváltoztatni) Microsoft Locator szolgáltatás tehát szintén egy névfeloldó szolgáltatás, amely az RPC szerviznek segít megtalálni és feloldani a különböző hálózati objektumok és szerverszolgáltatások nevét. Pontosabban az adott szerverprogramot vagy -szolgáltatást működtető gépen (ami lehet akár egy XP is, nemcsak a szerverek) futó Locator szolgáltatáshoz forduló RPC klienseknek az eszköz logikai nevéért cserébe az ismert hálózati nevét adja oda. Mert például egy printszerverrel kapcsolatban a kliens csak az előbbit tudja, de csak az utóbbival jut valamire. A Windows 2000 óta ez a szolgáltatás állandó „adatbázist" tart fenn ezekből a hozzárendelésekből, valamint támogatja az ACL-eket is. Még a hősidőkben az NT4 Server (jóval pazarlóbb és oktondibb módon) a memóriában tartotta ezeket a bejegyzéseket, ezért minden újraindítás után muszáj volt újraépítenie, azaz szórt üzenetekkel minden alkalommal utánanézni ezeknek az adatoknak. A másik opció ezen a panelen a DCE Cell Directory Service, amely kiválasztása és a NSID (NSI Daemon **[2]**) átjáró számítógép címének beírása után az RPC kérések névfeloldása átirányítódik a kijelölt, RPC kiszolgálóként működő gépre (pl. egy DEC alapúra).

Hogyan tudnám gyorsan megállapítani egy kliens MAC-címét a szerverről?

Nagyon egyszerűen: adjunk ki egy „ping" parancsot a kliens IP címére, majd az „arp -a" paranccsal listázzuk ki az Address Resolution Protocol (ARP) gyorsítótárat. Ebben rögvest megtaláljuk a kérdéses gép IP-címe mellett a távoli gép MAC-címét is, amit volt szíves mellékesen elküldeni még a ping-re válaszolás előtt. De igyekezzünk az adatok megtekintésével :D, mert a gyorsítótár tartalma dinamikus, és az időbélyeg miatt 2 perc után kikerül belőle a bejegyzés.

Szeretnék egy gyors ellenőrzést a TCP/IP konfigurációval kapcsolatban. Van ilyen eszköz a Windows Server 2003-ban?

Igen van, ez a „netstat" parancssori program, amellyel a protokollstatisztikát és a a TCP/IP-kapcsolatok különböző tulajdonságait lehet kilistázni. Több kellemes paramétere is van, pl. a „netstat -e" a hálózati statisztikát mutatja meg (a „netstat -s" ugyanezt protokollok szerint), míg a „netstat -r" pedig az útvonal-választási táblát és a hálózati kapcsolatokat listázza ki. A „netstat -o" hatására a portok állapotát és a folyamatok azonosítóját mutatja meg.

Mivel tudnám egyszerűen megoldani a multi TCP/IP beállítást?

Annak ellenére, hogy a Windows XP/2003 már ismeri az alternatív konfigurációt (de csak 1 fix és 1 DHCP-s kapcsolatra), továbbra is lehet gondunk a TCP/IP paraméterek állítgatásával például akkor, ha 2 fix IP-s hálózat között hordozgatjuk a laptopunkat. Van erre a problémára profi, fizetős szoftveres megoldás, pl. a NetSwitcher **[2]**, de van nekünk azért egy „csodaszerünk" is beépítve (Windows 2000-től felfelé), úgy hívják: netsh, azaz Netshell. Ezzel a parancssori eszközzel kapcsolatban úgy szól a fáma, hogy szinte csak kávét nem főz, de arra is betanítható. Az nslookup-hoz hasonlóan van interaktív (netsh>) és direkt paranccsori üzemmódja is van, valamint interaktív módban létezik egy ún. kötegelt mód (offline vagy a set mode offline paranccsal indítható), amikor a kiadott parancsok felhalmozhatók, majd egy kötegben hajtódnak végre a commit parancs kiadásakor. Az összegyűlt parancsok ilyenkor a flush paranccsal törölhetők is persze.
A TCP/IP paraméterek diagnosztikája és konfigurálása mellett használható többek között a DHCP, RAS, IPSec, PortProxy, IPv6, Routing, Network Bridge, RPC komponensekhez is. Teljesen korrekt rövidítési módszere van, minden parancs első két-három betűje is megfelelő (csak győzzük megtanulni), például:

```
sh ip int = show ip interface
```

A TCP/IP konfigurációváltás előkészítése és menete során először le kell mentenünk az aktuális beállításainkat egy szövegállományba, a következő paranccsal:

```
netsh -c interface dump > munkahely.txt
```

Ezután állítsuk át a paramétereket, majd az új (mondjuk, az otthoni) beállítást is mentsük le hasonló módon (otthon.txt). A váltáshoz pedig a

```
netsh -f munkahely.txt
```

parancsot kell használnunk, amivel egy következő üzemmódra is fény derül, ugyanis az „-f" hatására a szövegállományba beírt netsh parancsok futatthatók szkriptként. De még ez előtt, és miután elkészítettük a .txt állományt, nézzünk is bele rögvest, ugyanis hiba esetén a program nem szól egy árva szót sem, hanem csak szép csendben beleírja az adott állományba a hibaüzenetet.
Egyébiránt valóban lehetne fejleszteni a NetShell tudását (talán a kávéfőzésre is), hiszen ez a tudás gyakorlatilag a rendszerben lévő, a programot kiszolgáló .dll állományok (a „-c" kapcsoló után adjuk meg ha szükséges, lásd fent az interface parancsot) mennyiségén múlik.
A bővítéssel/törléssel és a segítő dll-ek listájának megjelenítésével kapcsolatos parancsok a következők:

```
netsh add/delete/show helper
```

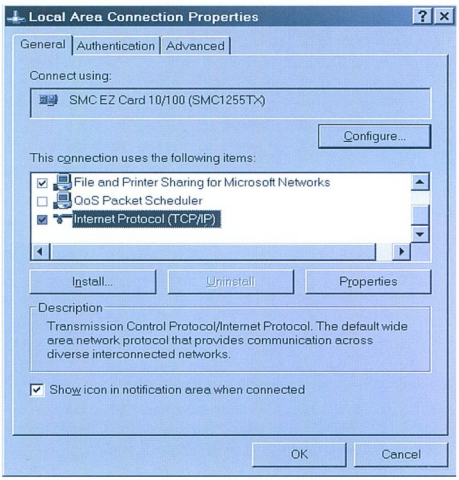
Még egy esetben jöhet jól ez a parancs, mégpedig biztonsági mentésként, hiszen ha esetleg lényegesen meg akarjuk változtatni az esetleg több interfészt is tartalmazó TCP/IP konfigurációnkat, akkor praktikus előtte a dump paranccsal lementeni.

Hogyan tudnám alaphelyzetbe állítani a TCP/IP protokollt? A „leszedem/felrakom" módszer nem jó, mert szürke az Uninstall gomb.

Ez bizony így van, előfordulhat, hogy a TCP/IP protokollt nem lehet eltávolítani, mert a TCP/IP verem az operációs rendszer alapvető fontosságú része. Viszont itt is segíthet a netsh! Ezt

a segédprogramot használva a TCP/IP verem az eredeti (az operációs rendszer telepítésekor érvényes) állapotba állítható vissza. A parancs használata a következő:

```
netsh interface ip reset tcpipreset.txt
```



Ennek hatására újraépülnek a TCP/IP verem által használt rendszerleíró kulcsok, ami gyakorlatilag ugyanazzal jár, mintha újraraktuk volna a protokollt. Viszont minden esetben szükség van a naplóállomány megadására, amelybe (ugyanúgy, mint a konfiguráció mentéskor) a netsh a parancsműveleteket írja be.

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

ÚJRA AZ SQL SERVER 2000-RŐL

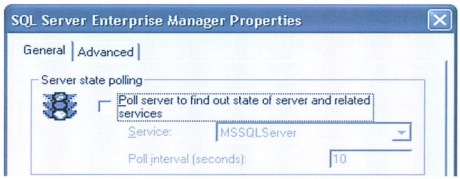
DATALENGTH() és a LEN() függvények használata

A *LEN()* függvény nem számolja a paraméterben átadott karakteres kifejezés végén lévő szóközöket, ami bizonyos esetekben félrevezető lehet. Ha ez kritikus, célszerű a *DATALENGTH()* használata, vagy az *RTRIM()*-el való kombinációja. Továbbá a *LEN()* függvény nem működik *TEXT, IMAGE* adattípusokra, szemben a *DATALENGTH()*-el. Arra figyelni kell, hogy unicode típusok *(NCHAR, NVARCHAR, NTEXT)* esetén duplán számol (byte szintű), ilyen esetben osztani kell kettővel.

```
SELECT LEN('Alma '), DATALENGTH('Alma ')
Eredmény: 4, 5
```

A felesleges hálózati forgalom csökkentése

Ha az *Enterprise Manager (EM) Tools → Options* menüjében a beregisztrált *SQL*-szervereink folyamatos státuszlekérdezését (*server polling*) letiltjuk, nagyban csökkentjük a felesleges hálózati forgalom indukálását. Ennek akkor van jelentősége, ha a hálózatban több SQL Server is üzemel, és ezek több gép *EM*-ében is be vannak regisztrálva. Ez egy nagyobb fejlesztői csoportban – helyi telepítésű *SQL Server Developer* verziókat feltételezve – tetemes mértékű is lehet. A másik eset, amikor a gépünkön üzemelő *EM*-ben csak átmenetileg elérhető (pl. *RAS* kapcsolat létesítése után) szerverek vannak nagy számban beregisztrálva. Az EM futtatása alatt ezek folyamatosan *ping*-elve vannak, hogy a fastruktúrában az adott szerver mellett megjelenő ikon tájékoztasson az adott szerver futási állapotáról (*started/stopped*).



1. ábra: Az Enterprise Manager-be regisztrált szerverek státuslekérdezésének letiltása

Táblareferencia-sorrend

Gyakran szükségünk lenne egy olyan táblalistára, amely meghatározza, milyen sorrendben tudjuk az adatokat átemelni (*INSERT*-álni) egy másik adatforrásból anélkül, hogy *Foreign Key* referenciát sértenénk vele.

```
CREATE FUNCTION RefCount (@tableName SYSNAME)
RETURNS INT AS
BEGIN
  DECLARE @objid INT, @refcnt INT
  SET @objid = OBJECT_ID(@tableName)
  SET @refcnt = 0
  DECLARE @temprefs TABLE (objid INT NOT NULL)
  INSERT INTO @temprefs
  SELECT DISTINCT rkeyid FROM dbo.sysreferences
  WHERE fkeyid = @objid
  WHILE (@@rowcount > 0)
    INSERT INTO @temprefs
    SELECT DISTINCT rkeyid FROM dbo.sysreferences
    WHERE fkeyid IN (SELECT objid FROM @temprefs)
          AND rkeyid NOT IN (SELECT objid
                             FROM @temprefs)
  SELECT @refcnt = COUNT(*) FROM @temprefs
  RETURN @refcnt
END
GO
SELECT TABLE_NAME, dbo.RefCount(TABLE_NAME) AS Ref
FROM INFORMATION_SCHEMA.TABLES
WHERE TABLE_TYPE = 'BASE TABLE'
ORDER BY RefCount
```


MÁTHÉ ZOLTÁN
mathez@bsi.hu
MCSD, SQL Server MVP

A cikkben szereplő URL-ek:

[1] http://www.lions.odu.edu/docs/dce/nsid.html
[2] http://www.netswitcher.com

Microsoft CRM v 1.2 – 3. rész

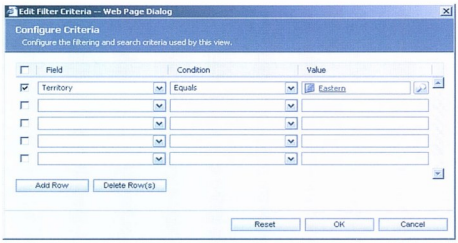
A Microsoft CRM testreszabásának lehetőségei

Mielőtt még fejlesztésen kezdenénk gondolkodni, vegyük sorra, miket szabhatunk testre a CRM beépített eszközeivel. Ezeket a területeket tekintjük át átfogóan cikksorozatunk harmadik részében.

A Microsoft CRM rendszert próbálták úgy megtervezni, hogy mind a különböző cégprofilokhoz, mind pedig a cégek, vállalkozások méretéhez igazodva is megfelelően lehessen használni. Bár számtalan mező, eset és opció érhető el, sok esetben szükségünk lehet saját, egyedi hierarchiára, menükre, adatbázisra, stb. Ezek megvalósítására két lehetőség kínálkozik: az egyik a CRM beépített testreszabási eszközei, a másik pedig a közvetlen alkalmazásfejlesztés a CRM-hez a Visual Studio .NET és .NET alapú technológiák segítségével. Jelen cikkünkben, első körben, a beépített testreszabási lehetőségeket tekintjük át dióhéjban.

Nézetek testreszabása

A listanézeteket, illetve az előnézeteket könnyen testreszabhatjuk, új oszlopokat vihetünk fel, illetve keresési és rendezési, szűrési feltételeket is megadhatunk, hogy mit is szeretnénk látni a listán. Létre tudunk hozni például egy olyan listanézetet a *Partnereknél (Accounts)*, ahol csak a keleti területhez tartozó megkereséseket látjuk.

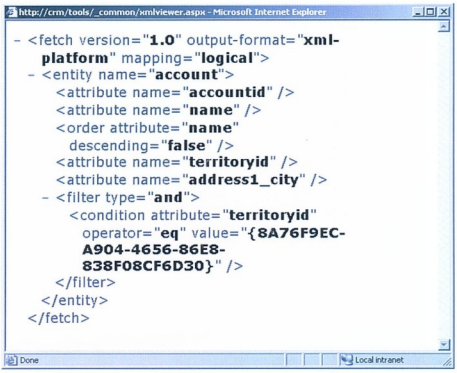


Szűrőbeállítások a listanézetnél

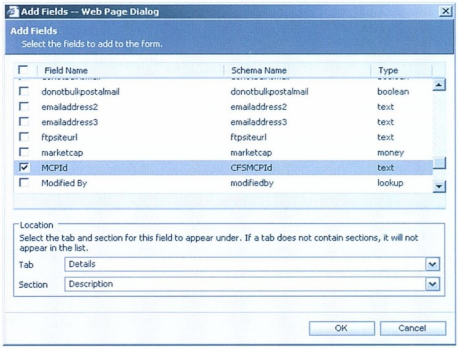
Lehetőségünk van arra, hogy egy elkészített nézet publikus, azaz mindenki számára hozzáférhető legyen, vagy korlátozhatjuk a nézetek használatát felhasználói csoportokra (team).

Űrlapok testreszabása

Az űrlapok esetében már jóval több a lehetőségünk, mint a nézeteknél. Nemcsak az űrlap „kinézetét" tudjuk megváltoztatni, hanem természetesen a tartalmát is, azaz például új mezőket vehetünk fel. Nézzünk erre is egy példát! A partnerinformációk között látni szeretnénk egy *MCPId* mezőt. Természetesen ezt még előbb létre kell hoznunk a *Deployment Manager* segítségével, itt most arra van lehetőségünk, hogy az új mezőnk láthatóvá és használhatóvá váljon az űrlapon.



Listanézetünk beállításai XML-ben

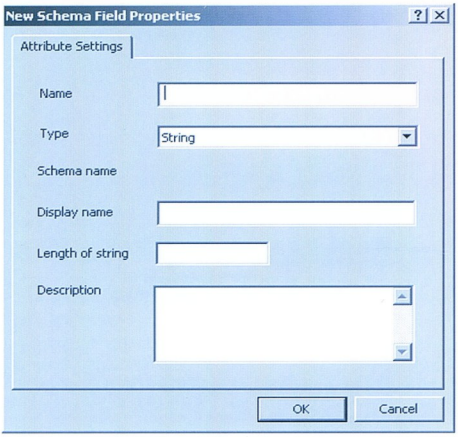


Új mező hozzáadása

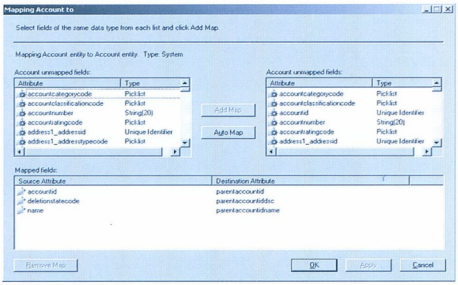
Ahogy az ábrán láthatjuk, a mezők különböző adattípusúak lehetnek. A *picklist (legördülő lista)*, illetve a *boolean (választható, igaz/hamis)* típusú mezők „rábírhatók" arra, hogy az űrlapon lévő más mezők értékére is hatással legyenek, de ehhez némi JavaScript programozás szükséges, melyet a HTML alapú lapokban könnyen megoldhatunk.

Deployment Manager – Schema Manager

Ezzel az eszközzel lehetőségünk nyílik módosítani az adatbázis sémát: új mezőket, illetve új megfeleltetéseket (mapping) vehetünk fel. Új táblákat sajnos nem tudunk hozzáadni adatbázisunkhoz, illetve fejlesztés nélkül csak olyan táblák között hozhatunk létre új megfeleltetéseket, ahol már elve vannak is. A megfeleltetések egyirányúak!



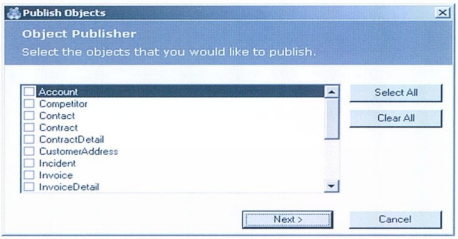
Új mező felvétele



Megfeleltetések szerkesztése

A testreszabások közzététele, export, import

Mielőtt a testreszabásokat használni tudnánk, azokat publikálni kell a kijelölt CRM szerveren. Erre a *Deployment Manager Customization Transport Manager* nevű eszköze szolgál. Lehetőségünk van importálni és exportálni is munkánkat. Nem szükséges közzétenni a nézetek módosításait, azok maguktól is „élesednek".



Testreszabott objektumok közzététele

Munkafolyamat automatizálás (CRM Workflow)

Egy példán keresztül röviden ismertetjük a CRM munkafolyamat (workflow) működését és a munkafolyamat készítés folyamatát. Első lépésként megadjuk, milyen rekordhoz készül a munkafolyamat. A CRM üzleti objektumait CRM rekordnak hívjuk. CRM rekord például a partner, a megrendelés, a számla, a termék, a tevékenység és a megjegyzés. A rekordok mezőkből állnak, mint például a név, termékkód, ár. Ezután megmondjuk, hogy a munkafolyamat milyen típusú, azaz mi indítja el a működését:

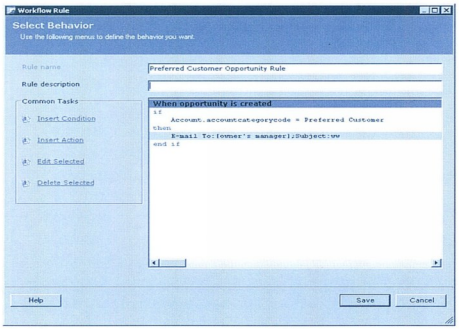
- *Manual* – azaz a felhasználó rendeli a rekordhoz
- *Assign* – akkor lép működésbe, ha a rekordot átadjuk egy másik felhasználónak
- *Create* – a rekord létrehozásakor lép működésbe
- *Change Status* – a rekord állapotának változásakor, például passzív állapotról aktívra, vagy zártról nyitottra váltáskor lép működésbe.

A következő lehetőségünk feltételeket vizsgálni, illetve feltételekre várni *(Check conditions, Wait for conditions)*. Megvizsgálhatjuk, hogy az adott rekordhoz kapcsolódik-e tevékenység *(Check activities)*, vagy a rekordon belül, illetve más, kapcsolódó rekordon belül van lehetőségünk valamely mező értékét vizsgálni *(Check object condition)*. Várhatunk feltételek teljesülésére tevékenységekkel kapcsolatban *(Wait activity condition)*, illetve a rekordok mezőivel kapcsolatban *(Wait object condition)*. Lehetőségünk van bizonyos idő leteltére is várni *(Wait for timer)*. A vizsgált feltételek alapján pedig műveleteket tudunk végrehajtani:

- *Create activity* – tevékenység létrehozása
- *Create note:* megjegyzés fűzése a rekordhoz
- *Send email:* CRM email küldés (a címzett lehet konkrét felhasználó, team, account, contact, vagy ún. logikai felhasználó, például a felhasználóhoz rendelt manager vagy team)
- *Update object:* a rekord egy vagy több mezőjének értékét adhatjuk meg
- *Change Status:* megváltoztatjuk a rekord állapotát
- *Assign object:* a rekordot más felhasználóhoz (vagy magunkhoz) rendelhetjük
- *Post URL:* egy ASP vagy ASPX lapra küldhetjük el a kiválasztott mezők értékeit XML formátumban.

Nézzünk egy egyszerű példát a munkafolyamat használatára: célunk, hogy abban az esetben, ha egy kiemelt partnerrel kapcsolatban rögzítünk lehetőséget, arról automatikusan email üzenetet kapjon a lehetőséget rögzítő felhasználó menedzsere:

- Típus: *Create*
- Feltétel vizsgálat: *check condition → object condition → Account.accountcategorycode = Preferred Customer*
- Művelet: *E-mali To:[owner's manager'], Subject: "A new opportunity has been created"*



Munkafolyamat automatizálás – Workflow Manager

Alap eladási folyamat (Standard Sales Process)

A CRM Standard verziójában nincs külön workflow, itt a Standard Sales Process áll rendelkezésre kereskedelmi folyamataink automatizálására, így az üzletkötőknek nem kell „kitalálniuk, hogy mi legyen a következő lépés".

Ha hozzákapcsoljuk egy lehetőséghez, a rendszer létrehozza azokat a kapcsolt tevékenységeket, amelyeket el kell végeznünk a lehetőséggel kapcsolatban. Emellett hozzárendel a lehetőséghez egy állapotkódot, először a *01 - Initial meeting* nevűt. Amennyiben a létrehozott tevékenységeket elvégeztük, automatikusan a soron következő állapotkódot (jelen esetben *02 - Sales Discovery)* és az ahhoz kapcsolódó tevékenységlistát kapjuk, egészen addig, míg el nem jutunk a *07 – Won* állapothoz. Természetesen a *standard sales process* is testreszabható a vállalat igényeinek, szokásainak megfelelően.

A következő részben alaposabban is megnézzük a testreszabási módszereket és eszközöket, gyakorlati felhasználást, és azt, hogy lehet felhasználni ezekhez a Microsoft CRM SDK és a .NET programozás nyújtotta lehetőségeket.

Azok részére, akik alaposan meg szeretnék ismerni a CRM rendszer testreszabási lehetőségeit ajánljuk a SZÁMALK Továbbképzés 8032-es, *Microsoft CRM 1.2 Customization* hivatalos Microsoft tanfolyamát.

Kovács László
(MCSE+S, MCT, CRM vezető oktató)
kovacsl@sbuilders.hu
Kovács Zoltán
(MCSE, MCDBA, MCSD, MCT)
kovacsz@sbuilders.hu
A szerzők a Számalk Oktatási Rt. Továbbképzés hivatalos Microsoft oktatói.

Magyarországon is elérhető a token alapú azonosítási technológia

A Microsoft Magyarország és az RSA Security bejelentette, hogy Magyarországon is elérhetővé vált az RSA SecurID token alapú azonosítási technológia, amely még nagyobb biztonságot nyújt a Windows operációs rendszerrel dolgozó vállalatok számára.

A Microsoft és az RSA Security 2004 februárjában jelentette be, hogy együttműködésre lépnek egymással, melynek keretében a Microsoft két faktoros azonosítási technológiával vértezi fel Windows rendszereit. A token alapú felhasználóazonosításra a Microsoft Windowsba épített támogatás áll majd rendelkezésre, mely a felhasználói név és jelszó mellett lehetővé teszi az RSA SecurID megoldása által biztosított, egyszer használatos azonosítóval történő hozzáférést a hálózati és lokális erőforrásokhoz, úgy online mint offline üzemmód esetén. Októbertől Magyarországon is kapható a Microsoft által támogatott és javasolt biztonsági megoldás. A megújult technológia fontos előnye, hogy az most már a Windows 2003 szervertermékek által is támogatottá és kompatibilissé vált.

Az RSA és a Microsoft közös fejlesztésének lényege, hogy az RSA által fejlesztett SecurID token megoldás segítségével a felhasználó adott időintervallumonként – alapból 60 másodpercenként – egyedi, változó kód birtokába jut. Összeállítva a felhasználó által ismert, a tokenhez tartozó PIN kódot és az előbbi ugrókódot, előáll az úgynevezett „passcode", amelyet jelszó helyett a megszokott Windows bejelentkező képernyőn adhatunk meg. Ezzel a címtárszolgáltatással szinkronizált módon folyamatosan változó jelszóval az értékes vállalati információk hatékonyabb védelmére nyílik lehetőség, hiszen a felhasználó mindössze saját PIN kódját kell hogy megjegyezze, mégis folyamatosan a vállalat által előírt hosszúságú és mégis változó kóddal azonosíthatja magát. A felhasználók kényelmét szolgálandó a tokenek többféle formátumban kerülnek forgalomba, úgy mint USB kulcs, smart card, számológép vagy kulcstartó, de létezik szoftveres változat is.

A Microsoft Megbízható Számítástechnika (Trustworthy Computing) nevű kezdeményezésének égisze alatt létrejövő együttműködés nagy előrelépést jelent a vállalat biztonsági erőfeszítései terén, az új azonosítás-technológia által ugyanis megnő a biztonság az online személyazonosság igazolása révén.

Ami a hivatalos Microsoft tanfolyamokból kimaradt...

Microsoft Office 2003 – FrontPage

Folytatjuk a „csináld magad" megoldásokat a Microsoft Office 2003 eszközeinek segítségével. Rovatunk mostani cikkében egy, a FrontPage 2003 által elkészíthető megoldás kerül nagyító alá. A FrontPage 2003 egyike az egyre népszerűbb kis- és középvállalatok részére kínált intranetes „csináld magad" megoldások eszközeinek.

A FrontPage már jó ideje része a Microsoft Office termékcsaládnak. Néha ugyan háttérbe szorult kissé, de a 2003-as csomagban újra előtérbe került és teljes fényében pompázik. Ma már szinte minden az integrált megoldásokról és a webes technológiáról szól. Mindezért érdemes (és talán érdekes is) újra nagyító alá venni a FrontPage-et, mint eszközt. Többen súgták a fülembe, hogy a Word-del is lehet igényes webes oldalakat készíteni, a profiknak meg ott van a VisualStudo.NET. Mégis maradnék a FrontPage 2003 eszköznél, ami hidat emel a „hobby programozó" és a professzionális megoldás között.

A feladat

Szakértői minőségemben az elmúlt évek alatt úton-útfélen beleakadtam abba a kérdésbe, hogy a céges internetes, intranetes oldalakat professzionális módon miként lehet több nyelven elkészíteni. Ez egy profi fejlesztő számára lefutott kérdés, de... Sokszor az adott cég vezetése nem áldoz költséget arra, hogy profi „webdesigner" céget bízzon meg. Talán sajnálja rá a pénzt, vagy talán egyszerűen azt gondolja: azért van az informatikus, hogy ezt is megoldja a többi bütykölnivaló mellett.

Sokszor láttam azt a megoldást sajnálatos módon (nagyobb cégeknél is), hogy egyszerűen minden oldalt kétszer készíttettek el, és manuálisan lefordították azokat. Persze ilyenkor mosolygunk magunkban, hogy amatőrök , pedig a valóság egyáltalán nem vicces. Ha nemcsak kettő, hanem több nyelvről is szó van, és viszonylag gyakran frissített oldalakról beszélünk, akkor ez igen emberes feladat annak, akinek a főnöke továbbpasszolta.

Tehát csináljunk egy olyan webes oldalt, amelyen a szövegek akár több nyelven is megjeleníthetők. Persze ha már itt vagyunk, ne érjük be ennyivel, és bolondítsuk ezt meg úgy, hogy a weblapunk figyelemmel legyen arra is, hogy a felhasználó milyen nyelven van bejelentkezve. Ez egy apró, de fontos csavar, hisz az internetes oldalak nagy többségénél nekünk magunknak kell kiválasztani a kívánt nyelvet. Csak ritkán látni olyan megoldást, ahol a manuális felülbírálási lehetőség mellett, automatikusan vált a nyelv a megfelelőre. Mindjárt feltételezzük is, hogy ezen cég mögött profi fejlesztő csapat áll. Pedig ez nem egy olyan nagy ördöngősség, és igen sok elismerést zsebelhetünk be vele. Arról nem is beszélve, hogy egyre több hazai cégnél kerül előtérbe a dinamikus nyelvezetű intranet, ahol nem árt, ha nem kell egérrel állítgatni napról-napra a saját készítésű oldalakhoz tartozó nyelvi környezeteket.

Szedjük össze a hozzávalókat

Egyrészről ugye azt mondanom sem kell, hogy a megoldás egy része az XML struktúrában leledzik szerencsére. Szerencsére, mert az XML sémák segítségével jól átlátható, és ezért jól kezelhető (javítható, frissíthető akár Word-ben is) hosszútávon a nyelvesítési folyamat. Másrészről pedig kézenfekvő ugye, hogy a FrontPage 2003 alkalmazást fogjuk használni a *.html, *.aspx (*.asp) oldalak elkészítéséhez. Először készítsünk egy teljesen egyszerű webes oldalt (minta.aspx). Megjegyzem, a minta igen bugyuta lesz, mert talán így sallangok nélkül könnyen érthető a lényeg:

```
<html>
<head>
<meta http-equiv="Content-Language" content="hu">
<meta http-equiv="Content-Type"
↳ content="text/html; charset=windows-1250">
<title>minta.aspx</title>
</head>
<body>
<p>Az én nevem Farkas Viktor</p>
</body>
</html>
```

Remélem senkit nem riasztottam el a kiindulási minta egyszerűségével. Lesz ez még csúnyább is.
Kerítsük elő a global.asax fájlunkat is (csak emlékeztetőül: a global.asax egy szerver oldali script fájl, ami a web alkalmazásunk indulásakor lefut, és amiben alkalmazás szintű eseményeket, objektumokat lehet definiálni). Magát a nyelvet itt célszerű meghatározni az automatizmusokhoz. Ezért definiáljunk egy változót neki, mondjuk „lang" néven. Háromféle prioritást állítunk fel (kicsit bővítve faladatunkat magánszorgalomból).

1. Amennyiben az URL címzésnél közvetlenül utasítjuk az oldalt egy adott nyelvre (pl.: http://minta.aspx&lang=hu), akkor ez a kérés (utasítás) mentődjön el egy cookie-ba is egyúttal. Mindezt azért, hogy legközelebb figyelemmel legyen az oldal ezen kérésre. Ez abból a szempontból lehet hasznos, ha mégis eltérő nyelven szeretnénk megjeleníteni a minta.aspx oldalt a Windows környezetétől (kibúvó kiskapukra mindig szükségünk lehet).
2. Amennyiben nincs ilyen közvetlen utasítás az URL címzésnél, akkor vizsgálja meg a mi kis megoldásunk, hogy van e már cookie mentve nyelvi beállítással. Ilyenkor alkalmazza automatikusan ezt a beállítást.
3. Végezetül, ha se közvetlen utasítás, se cookie nincs még mentve, akkor szemfüles módon vizsgálódjon a megoldásunk, és állapítsa meg kliens oldalon az IE böngésző nyelvi beállítását. Ha ezt sikeresen elkapta, mentse le szintén cookie-ba.

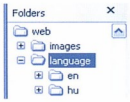
Szóval van eddig egy bugyuta minta.aspx oldalunk (ami inkább html, mint aspx, mert semmi nincs benne eddig amitől aspx lenne), és egy „lang" változónk. A Microsoft a nyelvek azonosítására 2-2 karakteres (összességében a kötőjellel együtt öt karakteres) névteret használ. Az első kettő az ország kódja szeretne lenni (pl: hu; en). A második kettő pedig a kultúra azonosítója (pl.: en-us; en-uk). Nomármost ezzel lesz még némi bajunk, mármint hogy mikor kettő, mikor öt a karakterek száma. De ezt a bibit előnyünkre fordítjuk kicsit később. Az eltéréseket leginkább az operációs rendszerek verziói közötti különbségek adják. Például a Win98-on más eredményt ad vissza a kérés, ha a saját (eredeti) IE fut rajta, mint amikor Upgrade-elt a böngésző mondjuk 6.1-re.

A golobal.asax

Meg kell tehát határoznunk a „lang" paraméter értékét a fentiek figyelembevételével a global.asax fájlban:

```
if len(Request(„lang")) > 0
  then 'ha van értéke a lang változónak
  Session(„lang") =
  Request(„lang")
  Response.Cookies(„MINTA")(„lang") =
  Request(„lang")
  Response.Cookies(„MINTA").Expires =
  DateAdd(„m",1,Now())
  ElseIf Request.Cookies(„MINTA")(„lang") <> „"
    then 'a cookie érték átadása a lang változónak
    Session(„lang") =
    Request.Cookies(„MINTA")(„lang")
    Else
    Session(„lang") = Request.ServerVariables
    („HTTP_ACCEPT_LANGUAGE")'IE felismerése
End if
```

Jól látható a hármas tagolódás a feltétel szerkezetében, ami meghatározza a „lang" változónk értékét. Azt is megfigyelhetjük, hogy a változó értéke a session-ra lesz érvényes, tehát minden oldalra, amit megnyitunk a böngészőben mindaddig, míg ki nem lépünk belőle. Továbbá az is kivehető a fenti script részletből, hogy a mentendő cookie „MINTA" névre fog hallgatni.
Meg kell még határoznunk a nyelvi (xml) fájlok elhelyezkedését, elérési útvonalát. Az áttekinthetőség érdekében (amennyiben több oldalt nyelvesítünk) célszerű követni a nyelvi könyvtáron belül az eredeti könyvtárstruktúrát, és azon belül is az .asp fájlok elnevezéseit, mindezt úgy, hogy jelöljünk ki egy gyökér könyvtárat amihez képest kezelhetjük a relatív eléréseket. Legyen ez esetünkben a „language\" könyvtár.



Definiáljunk egy új változót „langdir" és tároljuk benne a gyökerünket.

```
langdir = Request.ServerVariables
(„APPL_PHYSICAL_PATH") & „language\"
```

Ezután a gyökér könyvtárhoz adjuk hozzá a „lang" változóban őrzött értéket, és ellenőrzésképpen vizsgáljuk meg, hogy létezik-e egyáltalán ilyen könyvtár. Nemcsak azért kell ezt megtennünk, mert azt tanultuk az iskolában, hogy mindent ellenőrizni kell, hanem azért is, mert ez esetben kezelhetjük a kétkarakteres és ötkarakteres válaszul kapott eredmények közötti eltéréseket.

```
If Not fs.FolderExists
(langdir & Session(„lang")) Then
  filename = split(Session(„lang"),"-")
  Session(„lang") = filename(0)
  If Not fs.FolderExists
  (langdir & Session(„lang")) Then
    Session(„lang") = left(Session(„lang"),2)
    If Not fs.FolderExists
    (langdir & Session(„lang")) Then
      Session(„lang") =
      config(„lang") 'Fallback kiolvasása
    End if
  End if
End If
```

A script részlet először is megvizsgálja, létezik-e a kívánt nyelvi könyvtár (pl.: language\hu).
Ha nem, akkor megvizsgálja a kötőjel (-) előtti két karakteres egyezőséget.
Ha így sem jut pozitív eredményre a script, akkor levágja az első két karaktert „lang" tartalmából, és ezt veszi alapul. Ez azért jó, mert így lehetőség nyílik olyan egyezőségek kezelésére is, ahol mondjuk teljes nevek vannak kezelve (pl.: Hungary – hu; English – en). No ez az a pont, ahol a bibit előnyünkre fordíthatjuk, mert nem kell pontosan tudnunk mondjuk az URL-ben deklarálni kívánt nyelv két- vagy ötkarakteres azonosítóját, elég ha csak a közelbe lövünk (pl.: hu=Hungary; hun).
A legvégén, ha egyik módon sem jutunk értelmes eredményre, ami az eddigiek alapján nehezen elképzelhető, akkor a „fallback" segítségével a „config" fájlhoz nyúlunk segítségért. A config fájl egy segéd fájl, amiben az ilyen esetekre gondol-

va közvetlenül definiálhatunk alapértelmezett elérési utakat (pl.: lang = hu). A „config" fájl használata nem kötelező érvényű, de nagyobb rendszereknél ajánlott, de ezen cikkben erre nem térek ki részletesebben.
Tulajdonképpen a global.asax fájlunk nyüstölését be is fejeztük. Persze a nyelvesíteni kívánt oldalunk adott nyelvi tartalmát egy xml objektumba létre kell hozni, melyet a legegyszerűbben úgy tudunk megtenni, ha a nyelvesítés gyökerébe (azaz a language\ könyvtárba) létrehozunk egy „instruct.aspx" fájlt, melynek ez légyen a dolga.

Az instruct.aspx

Szóval itt is script-elünk kicsit, de nem komplikáltabban, mint eddig. Tehát az instruct.aspx feladata az lesz, hogy az aktuális nyelvesíteni kívánt fájlnak megfelelő nyelvi tartalmat elkészíti egy xml objektumba, és betölti azt a memóriába. Az előző oldalon már kitértem arra, hogy érdemes követni a nyelvesítéseken belül az aspx fájlok neveit az xml fájlok elnevezésénél. Ahhoz, hogy tudjuk mely fájlok szükségesek éppen számunkra a nyelvesítéshez, szükségünk van egy változóra mely ezt az információt tartalmazza az instruct.aspx számára. Ez a változó legyen a „who". Mivel ennek egy átadott (kapott) értéknek kell lennie ezért ezt máshol definiáljuk, most csak jegyezzük fel, hogy van egy elvarratlan szálunk. Amennyiben tehát nem egy nagy nyelvi xml fájlunk van, hanem oldalanként külön-külön, úgy a memóriába mindig csak azon részlet töltődik be a nyelvesítéshez, melyre éppen szükségünk van. Ez nemcsak spórol a memóriával, hanem lényegesen gyorsítja is a működési procedúrákat.
Visszatérve oldalirányú gondolataimtól, szükséges egy függvénydefiníció a megfelelő nyelvi tartalmak lekérdezéséhez:

```
Function getLangString(strName)
Dim xmlNode
   Set xmlNode = cLangDoc.selectSingleNode
        („mintalang/" & strName)
```

Ez a mini script részlet önmagában nem elégséges a kívánt eredményhez, csupán helyszűke miatt ragadtam ki ezt a részletet, mely a lényegét mutatja a feladatnak. Mint minden ilyen esetben, most is kiválló barátom (alias Tocsi) keze-nyoma található a letölthető zip fájlban (a zip fájl tartalmazza természetesen az instruct.aspx-et is teljes terjedelmében), mely az alábbi címen érhető el:
http://www.iqjb.hu/UserFiles/tn_frontpage.zip.
Ugye érthető, hogy az „xmlNode" tartalmazza nyelvesített tartalmat (pl.: Az én nevem: Farkas Viktor), melyet a strName segítségével határozunk meg. Itt csupán az a kérdés, hogy honnan kerül adat az „strName" változóba. Erre visszatérünk, mint második elvarratlan szálra. Ha tehát eddig jól figyeltünk, akkor arra van most szükség, hogy az xml struktúrában hol található meg a kívánt nyelvi megfelelés. Tehát elérkeztünk az xml fájlunk felépítéséhez.

Az xml fájl

Mint már többször (tehát most is sokadszorra) említetem, a nyelvi xml fájlok kövessék lehetőség szerint az aspx fájlok neveit. Ebből következően a mi kis bugyuta mintánk esetében a nyelvi fájlok a következők:
..\language\hu\minta.xml
..\language\en\minta.xml

Az xml fájlokban mindig használjunk egy gyökér névteret. Ezt ugyan csak most említem (jobb később mint soha), de az instruct.aspx-ben már hivatkozom rá a fentebbi script részletben. Ez pediglen a „mintalang". Mivel körbe hivatkozás van a feladatban szereplő fájlok között, elkerülhetetlen volt, hogy egy ilyen előhivatkozást megtegyek az elvarratlan szálak mellett. A végén fény derül ennek is a szerepére, de ne menjünk nagyon előre. Nézzük inkább az xml fájlok felépítését. Elsőként a „hu" könyvtár alatti minta.xml-t, majd az „en" könyvtár alatti xml-t:

```
<?xml version="1.0" encoding="utf-8"?>
<mintalang caption="Magyar">
    <default>
        <text>Az én nevem Farkas Viktor</text>
    </default>
</mintalang>

<?xml version="1.0" encoding="utf-8"?>
<mintalang caption="English">
    <default>
        <text>My name is Viktor Farkas</text>
    </default>
</mintalang>
```

Azt hiszem magukért beszélnek a minták. A „mintalang" is értelmeződött. Nincs más hátra, mit megmutatni az eredeti minta.aspx fájlunknak, hogy innen vegye a feliratokat.

A minta.aspx

Visszatértünk az eredeti kiindulási állapotunkhoz. Jó nagy vargabetűt tettünk, de nem hiába. No meg nem is volt ez olyan komplikált, mint elsőre látszik, úgyhogy letörölhetjük izzadság cseppjeinket homlokunkról. Mindenekelőtt van két elvarratlan szálunk, melyek közül az első ugye a „who" változónk mibenléte. A „who" változónak az a szerepe, hogy tudja melyik xml fájlban található az oldalhoz tartozó nyelvesítő struktúra. Tehát ezt mindenképpen definiálnunk kell a minta.aspx fájlunk elején:

```
<%const who="minta.xml"%>
```

Amennyiben egy alkönyvtárban található a cél xml fájl, úgy meg kell adni az alkönyvtárt is úgy, mintha az adott nyelvi könyvtár lenne a gyökér (pl.: language/hu/ = gyökér elérési út). Ezt követően include-olni kell a instruct.aspx fájlt, hogy elkészüljenek a megfelelő xml objektumok a memóriába.

```
<!— #include file=
„../language/instruct.aspx" —>
```

Végezetül pedig ki kell cserélni a nyelvesítendő részt az általunk meghívható függvényre.

```
<%=getLangString(„default/text")%>
```

Rakjuk össze a részleteket, és nézzük meg, hogyan is néz ki minta.aspx fájlunk, a cserélgetések után egyben:

```
<%@ Language=VBScript %>
<%const who="minta.xml"%>
<!— #include file=
„../language/instruct.aspx" —>
<html>
```

```
<head>
<meta http-equiv="Content-Language" content="hu">
<meta http-equiv="Content-Type"
content="text/html; charset=windows-1250">
<title>minta</title>
</head>
<body>
<p><%=getLangString("default/text")%></p>
</body>
</html>
```

Jól látható a getLangString függvény meghívásánál a második elvarratlan szálunk megoldása. Azaz megadjuk a paraméter értékét (esetünkben: default/text) mely a instruct.aspx-nek átadva értékül szolgál a strName változónak. Innen tudja a függvény, hogy az strNode változóba melyik xlm elemet kell felvennie a minta.xml fájlból.

Mire jutottunk

Elértük, hogy bugyuta mintafeladatunk két nyelven is elérhető. Továbbá azt is, hogy automatikusan kiválasztja a böngészőnk nyelvi beállításai alapján az általunk (mint kliens által) használt nyelvjárást. Sőt, le is menti cookie-ba, hogy a legközelebbi visszatérésnél eleve ezzel a nyelvvel induljon az oldal. Figyelnünk kell azon klienseken, ahol az SP2 már telepítve van, mert a cookie-k kezelésére ráült a szúrós szemű figyelés. Persze hagytunk magunknak kiskaput, és közvetlenül is megadhatjuk az URL megadásánál a kívánt nyelvet. Továbbá szintén a jól kezelhetőség érdekében elértük, hogy xml fájlokban tároljuk a nyelvesítéshez tartozó információkat, ezért az esetleges helyesírási javításoknál nem kell hozzányúlni az eredeti fájljainkhoz. Sőt bármikor újabb nyelveket hozhatunk létre anélkül, hogy módosítani kéne az eredeti oldalainkon. Mindezt persze egy notepad segítségével is elkészíthettük volna, és eddig nem is igen esett szó magáról a FrontPage2003-ról. Mint a cikk elején is írtam, a FrontPage 2003 hidat emel a „hobby felhasználó" és a professzionális megoldások között. A FrontPage 2003 alkalmazás remekül kezeli az aspx fájlokat, mutatja az include-olt összefüggéseket, kezeli az xml struktúrákat. Tehát egyben átláthatóan lehet vele egyszerűen készíteni – a mintafeladatunkban szereplő megoldásnál lényegesen összetettebb oldalakat, melyek ténylegesen professzionális megjelenést képesek nyújtani azoknak is, akik maguk készítik internetes, intranetes oldalaikat. Amiről eddig írtam, az a FrontPage 2003 segítségével könnyen alakítható akár egy SharePoint kijelzővé, így téve spéci oldalainkat a portálunk integrált részévé. De erről a következő számban, ahol Exchange 2003 „admin" eszközöket fogunk készíteni FrontPage segítségével a Portál rendszerünk számára (természetesen több nyelven is kezelhetően). Aki ennél többet szeretne tudni a témáról, az keresse fel bátran a John Bryce Oktatóközpont munkatársait...

FARKAS VIKTOR
IQSOFT - John Bryce Oktatóközpont
farkasv@KSZKI.OBUDA.hu
MCSE, MCT, HP-ASE

A cikkben szereplő URL-ek:

[1] http://www.iqjb.hu/UserFiles/tn_frontpage.zip

Dr. Watson

ISA Server 2004 áttekintés

Dr. Watson most nem kevesebbre vállalkozik, mint hogy néhány oldalban összefoglalja az ISA Server számos újdonságát. Ez azért számít kihívásnak, mert az ISA 2000-nek halvány nyoma sem maradt az új termékben. Új a felhasználói felület, a tűzfalkoncepció, a szabályok kiértékelése, a hálózatok kezelése, a naplózás és még sok minden más. Ahogy elnéztem, egyedül a jelentéskészítés hasonlít az elődöt megoldására.

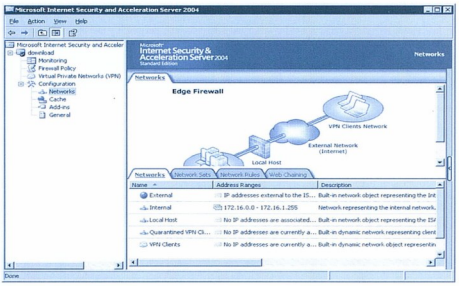
LAT-olgatunk?

A legeslegfontosabb újdonság az ISA 2000-hez képest, hogy nincs többé LAT (Local Area Table). Azaz nincs olyan kiemelt hálózat, amely felé az ISA teljesen nyitott, ahonnan szinte meg sem lehet állapítani, hogy tűzfallal van-e dolgunk, mert teljesen védtelen, meztelen. Épp az imént ütköztem bele ebbe a „problémába": az egyik tantermi számítógépünkről próbáltam Terminal (RDP) ablakot nyitni, de rá kellett jönnöm, hogy a tantermi IP-címekről és gépekről ez nem fog menni, mert – nem véletlenül – ezt a hozzáférést előzőleg nem engedélyeztem. Telepítéskor ugyan feljön egy LAT-bekérő-szerű párbeszédpanel, ahova a belső címeket begépelhetjük, de ez pusztán a hálózatok közötti alapértelmezett kapcsolat beállítására szolgál. Nem „tépi le" az ISA egyik oldaláról a páncélt.

Az ISA 2004 tehát lezárva születik, de ez most már nemcsak a külső hálózatra igaz, hanem mindegyikre.

Hálózatok

Telepítés után az első dolgunk a hálózatok megtekintése legyen. Megfigyelhetjük, hogy tűzfalunk öt hálózatot ismer (ha két hálózati kártyája van):



Az ISA 2004 hálózatai

Ezek közül kettőt tartok igen figyelemre méltónak: a Local Hostot és a Quarantined VPN Clients hálózatot. Az első ugyanis megtöri azt az átkot, amitől a korábbi változat szenvedett, nevezetesen hogy az ISA Server nem lehet kliense (ügyfele) önmagának. De! Lehet! Nem kell többé téglákat kiütni a tűzfalból ahhoz, hogy az ISA Serverről is elérjünk mondjuk egy POP3-as levelesládát. A Local Host „hálózaton" azok a TCP/IP-alapú programok találhatók, amelyek magán a tűzfalon futnak (SBS-vásárlók mondják utánam: hurrá!).

A Quarantine (karantén) hálózat pedig azokat a VPN-ügyfeleket foglalja magába, akik (még) nem tettek eleget azoknak a feltételeknek, amelyek feljogosítanák őket a zárka elhagyására. A ketrecből csupán (általunk megadott IP-szűrővel) korlátozott forgalmat kezdeményezhetnek a belső hálózat felé.

A webproxy szolgáltatás „lecsúszott" a kiszolgáló szintjéről (ISA 2000: Outgoing Web Requests) a hálózatok szintjére. Mindegyik hálózaton egyenként be lehet állítani, hogy fogad-e (és ha igen, melyik portján?) web proxy és tűzfalügyfeleket.

Most menjünk „két füllel jobbra". A Network Rules lapon látható, hogy mi művel az egyes hálózatokból kiinduló, és más hálózatokba tartó forgalommal az ISA. Mint ahogy LAT sincs, a NAT (Network Address Translation, belső IP-címek átfordítása publikus címmé) sem kötelező többé. Van ugyan egy alapértelmezett szabály a belső hálózat és a külvilág közötti NAT-olásra, de ez immár nincs kőbe vésve, ha úgy tetszik, szabadon megváltoztatható:

Hálózatok közötti IP-forgalom meghatározása. NAT vagy Route?

Erről a fenti ábráról az is leolvasható, hogy a választék mindössze ennyi: NAT (címfordítás), vagy Route (hagyományos útválasztás) történjen?

Furfangos kérdés: vajon az ISA Serveren futó alkalmazások NAT vagy Route módszerrel találnak ki az internetre?

Szabályok

Menjünk a kezelőfelületen a Firewall Policy (tűzfalszabályok) ágra! Itt egyetlen, jól láthatóan tiltó házirendet találunk, melynek láttán azt a téves következtetést vonhatnánk le, hogy ez az ISA süket és vak:

Alapértelmezett felhasználói tűzfalszabály

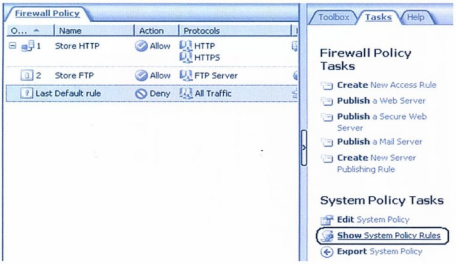
Pedig erről nincs szó. Például zökkenőmentesen eléri a belső hálózat Windowsos kiszolgálóit a 445-ös porton, noha visszirányban ez nem igaz, ő nem válaszol a fájlmegosztási kérelmekre. Nem érhető el egyetlen normális webhely sem, bezzeg a www.microsoft.com! (Minden szentnek maga felé hajlik a keze...) Hol bujkál ez a sok szabály ebben az egy tiltásban? Nem ebben bujkál, hanem a Rendszerszabályok között!

Rendszerszabályok (System Policy)

Mindenkinek azt javaslom, hogy az ISA 2004 felderítésénél különös hangsúlyt fektessen a Rendszerszabályok átfésülésére, mert:

1. lehet, hogy valami olyasmi is elérhető az ISA Serverről, vagy visszirányban, amire álmunkban sem gondoltunk
2. lehet, hogy valamilyen hozzáférést kár „megvarázsolnunk", mert van már rá (üres vagy tiltott) Rendszerházirend, amit két kattintással átalakíthatunk
3. ha létrehozunk egy ugyanolyan házirendet, mint ami rendszerszinten is rendelkezésre áll, elindulunk a káosz felé vezető rögös úton
4. úgysem sikerül létrehoznia a megfelelő szabályt. Vagy meg tudná valaki mondani, hogyan kell engedélyezni a Computer Management MMC snap-in hálózati kapcsolatát?

Tehát: Rendszerházirend megtekintése:

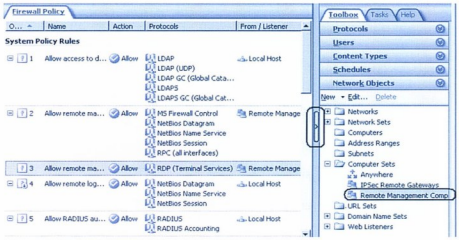


A Rendszerházirendek megjelenítése

Amint idekattintunk, a szabályok száma 30-ra kúszik fel! Ezek többsége arra való, hogy az ISA Server távfelügyeletét meg lehessen oldani, vagy hogy elérje az Active Directoryt stb. Példaként nézzünk egy fontos feladatot: be szeretnénk terminálni (RDP) az ISA Serverre. Ehhez akár rittyenthetünk egy saját szabályt is, de jobb választásunk tűnik a hármas számú szabály felhasználása, amely máris engedi az RDP protokollt bizonyos Remote Management Computers és az ISA Server között:

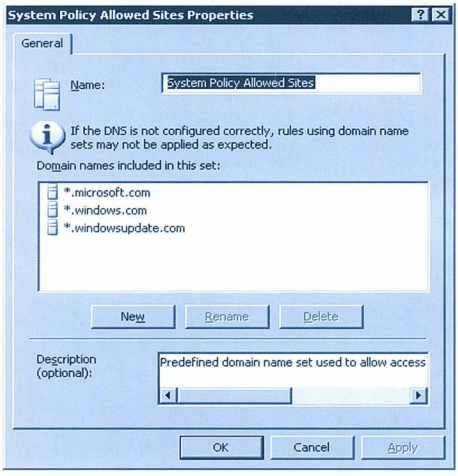
Távfelügyeletet engedéyező szabály

De kik tartoznak a Remote Management Computers csoportba? Rántsuk csak elő a konzol jobb oldaláról a szerszámkészletet (Toolbox)!



A szerszámosláda egyik értékes eleme

Itt találjuk ezt a bizonyos csoportot, mégpedig üresen. Hát rakjuk meg számítógépekkel! (Csak ésszel!)
Most nézzük meg, miért érjük el a www.microsoft.com-ot? Én egy http szabályra gyanakszom...
A 17. szabály pont http-ről szól: a LocalHost (az ISA Server maga) elérheti a System Policy Allowed Setbe tartozó webcímeket:



A szent keze, amely maga felé hajlik...

Hát így állunk. Mentségükre legyen mondva, hogy az utolsó cím elérése tényleg fontos, mert onnan lehet letölteni a biztonsági javításokat.
A hátralévő 28 Rendszerházirendet ISA Server tanfolyamon nézzük meg részletesen, hisz egy egész tech.net magazint megtöltene, ha ezeken most végigmennénk.

Felhasználói házirendek

Ha vállalatunknál felmerül az igény a www.microsoft.com-on kívül eső webcímek meglátogatására is (kötve hiszem! ☺), kénytelenek leszünk saját házirendet létrehozni. Szerencsére a szabályvarázsló elég egyértelmű. Amit azonban az ISA 2000-

szerelmeseknek figyelembe kell venniük, hogy minden hozzáférés pusztán **szabály**, és nem Site and Content Rule, Access Rule vagy Packet Filter, és nem Web Publishing, de nem is Server Publishing. A megkülönböztetés nem létezik többé. A szabály az szabály és kész. Minden hozzáférés egy közös listában látszik, nem kell ötfelé kavarogni a fában, hogy megtudjuk, mit is csinál egy adott ISA.
De hogy a régi motorosok is élvezkedhessenek valamin, ötféle varázsló viszont van. Az egyik kienged, a másik beenged, a harmadik a levelezést engedélyezi, a negyedik megsütötte, az ötödik pedig, az icike-picike, mind megette!

Mikor lép életbe?

A szabályok, hálózatok és egyebek megváltoztatása után az ISA Server nem áll át azonnal az új beállítások használatára. Ehelyett megjelenik egy figyelmeztető csík, hogy tényleg véglegesíteni szeretnénk-e mindazt, amit csináltunk. Ennek a módszernek nagy előnye, hogy seregnyi változtatást egy lépésben tudunk véglegesíteni. Kisebb terheltségű webhelyeken ez nyilván nem számít, de nálunk, ahol még éjszaka is 50-60 egyidejű Internetes felhasználó lóg a gépen (nappal pedig több száz), bizony nem mindegy, hogy hányszor szakad meg a kapcsolat a szolgáltatások átparaméterezése, esetleg újraindítása miatt.

ISA-szabályok elfogadása vagy elvetése

Export-import

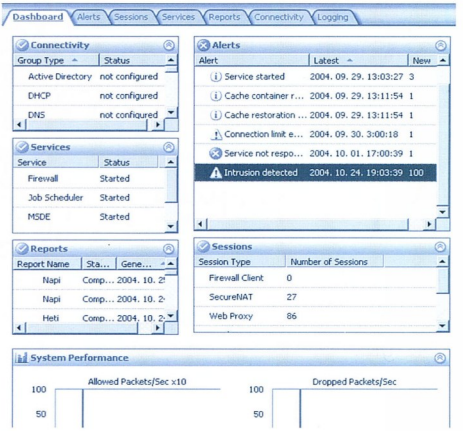
Ha készen vagyunk a szabályainkkal, XML-fájlba menthetjük, ami kiváló archiválási lehetőség, egyszersmind lehetővé teszi, hogy egyik ISA Serverről beállításainkat átvigyük egy másikra. Az export-import a fában majdnem mindenhol előcsalogatható az egér jobb gombjával, ezáltal akár egy-két beállítást is átvihetünk, nem kell mindjárt egy teljes szervert klónozni.

Monitorozás

Az úgynevezett Dashboard (műszerfal) segítségével egy pillanat alatt át lehet tekinteni az ISA Serveren futó folyamatokat. Példaként álljon itt a mi szerverünk éjszaka fél 9-kor lekapott műszerfala. Első ránézésre feltűnik az Intrusion Detected (betörési kísérlet történt) üzenet. Ez az egy sor valójában vagy 826 korábbi IP HalfScan Attack összefoglalása egyetlen üzenetbe. Ha akarom, kibontom, ha úgy óhajtom, csoportosan törlöm. Azután van ott egy Connection Limit Exceeded üzenet, amiről sejtelmem sincs, hogy mi lehet…
Ezen a késői órán 98 egyidejű „júzerünk" van, ez viszont mind külső, mert a belső hálózaton egyes egyedül én vagyok ébren.

Napló

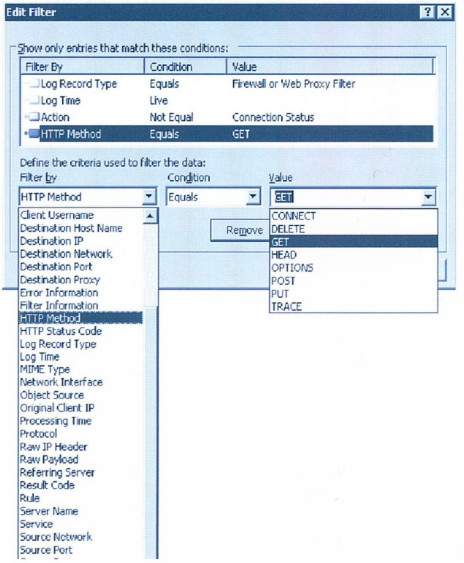
Az események naplózása teljesen megújult, telepítő felpakol nekünk egy MSDE adatbáziskezelőt, abba kerülnek a logok. Mivel mostantól egy SQL Server kezeli a feljegyzéseket, könnyű volt egy remek új szolgáltatást hozzárittyenteni: a logon menet közbeni (online) megtekintését! Ahogy a költő mondaná:



Az ISA 2004 műszerfala

A Monitoring ágon ülve kattintsunk a Logging fülre.

Itt keressük meg a Start Query linket a jobb oldalon, és máris elindul az élő adás. Ebben mind a Firewall, mind a WebProxy szolgáltatáshoz érkező kérések és reakciók elolvashatók. Ha pedig megtaláljuk a lekérdezés szerkesztése lehetőséget, újabb hétnyi mazsoláznivalóhoz jutunk:



Föti Marcell
MCSE, MCT, MZ/X
marcellf@netacademia.net

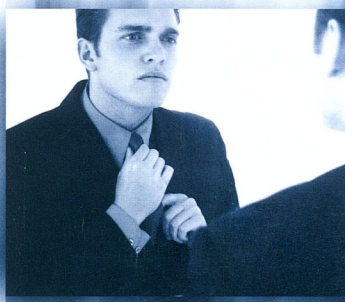
netACADEMIA

A LEGJOBBAKAT TANÍTJUK

Arccal az MCSA felé!

Akciónk keretében a Microsoft Windows 2003 rendszeradminisztrátor minősítés (MCSA) megszerzéséhez szükséges 3 + 1 tanfolyamot megrendelők részére cégünk a vizsgákat ingyenesen biztosítja.

Az MCSA minősítés előnyei:



- Az Ön szakmai felkészültségének elismerése az iparágon belül.
- Közvetlen hozzáférés az MCP tagok weboldalán keresztül a Microsoft legfrissebb technikai és termékinformációihoz.
- Az adott minősítést igazoló logók elérhetősége és használata (szabályozott kereteken belül).
- Ingyenes vagy kedvezményes meghívások itthoni és külföldi konferenciákra, tanulmányutakra, rendezvényekre.
- Ingyenes hozzáférés a Microsoft MCP Online Magazine-ra és egyéb minősítés-specifikus számítástechnikai kiadványokhoz.
- A kedvezményes előfizetés a Windows & .NET Magazinokra.
- Illetve néhány közvetett előny (például karrierlehetőségek itthon és külföldön).

Az MCSA-képesítés a következő feladatok elvégzésére készít fel:

- Microsoft Windows Server 2003 környezetek menedzselése és hibaelhárítása.
- Active Directory kialakítása és üzemeltetése.
- Group Policy Objektumok létrehozása és konfigurálása. Felhasználói környezet kezelése csoportos házirenddel.
- Erőforrások hozzáférésének kezelése.
- A kiszolgáló teljesítményének monitorozása.
- Katasztrófa előtti állapot visszaállításának kezelése (Disaster Recovery).
- Szoftverek karbantartása Software Update Services használatával.
- TCP/IP alapú hálózati környezetek kialakítása.
- Biztonságos hálózat kialakítása.
- Munkaállomások telepítése és konfigurálása.
- Az új ISA Server 2004 konfigurálása és üzemeltetése.

A részvétel feltételei:

- A résztvevőknek egy éven belül el kell végezniük mind a négy MCSA tanfolyamot.
- Az ingyenes vizsgabónt mindig a következő tanfolyam megrendelésekor adjuk ki (tehát az elsőt a második tanfolyam megrendelése után, a 2. bónt a 3. tanfolyam után stb.).
- **Jelentkezési határidő: 2005. február 25.**

Az összeállítás csak egy a lehetőségek közül.
A teljes kínálatról tájékozódjon honlapunkon!

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®