tech.net

WORKING WITH WINDOWS

III. évfolyam
11. szám

Ára: 1344 Ft

A TARTALOMBÓL

A Single Sign On, mint biztonsági rés
4. oldal

Tanusítványok és magánkulcsok a Windowsban
6. oldal

Intelligens kulcstartók
11. oldal

ISSN 15865185

9 771586 518005 11

A NetACADEMIA és a NETLOCK Az Első Hitelesítés Szolgáltató közös szervezésében

Szerezzen átfogó tudást az elektronikus aláírás és titkosítás vállalati szintű alkalmazásában

Technikai blokk:

- Mi az a kriptográfia?
- Hogyan tudunk nyílt hálózaton kommunikálni úgy, hogy azonosítani tudjuk kommunikációs partnerünket?
- Meg tudunk bizonyosodni üzeneteink **sértetlenségéről?**
- Hogyan működik a **titkosítás**? Mi a **kulcspár**, a **tanúsítvány**, mi a szerepe az intelligens kulcstároló **eszközöknek**?

További témakörök:

- szimmetrikus algoritmusok, nyílt kulcsú **RSA**, hash, a **X.509** tanúsítványok szerepe és **használata**,
- hitelesítésszolgáltató, a hierarchia felépítése,
- kulcstároló eszközök: **intelligens kártya** és **USB Token**,
- bejelentkezés tanúsítvánnyal (Single Sign On), SSL, S/MIME, HTTPS,
- **virtuális magánhálózatok**.

Jogi blokk:

- az elektronikus aláírási törvény célja, következményei, a végrehajtási rendeletek.
- Jogkövetkezmények, és a szükséges feltételek megléte.

Ajándék!

Hallgatóink a tanfolyamon használt kriptográfiai eszközöket a tanfolyam után megtarthatják, és - a mellékelt NetLock tanúsítványokkal - hiteles elektronikus aláírást és nyílt kulcsú titkosítást használhatnak!

Hozza el a főnökét! **Jelentkezési határidő: 2002. november 29.**

Ha cégüktől **két fő** jelentkezik a PKI-workshopra, mindketten ingyen részt vehetnek Mikulás-konferenciánkon!

(Mikulás-konferenciánkról további információk az első borító hirdetésében vagy a http://www.netacademia.net/mikulas címen!)

A tanfolyam időpontjai	Hossz [nap]	Bővebb információ és jelentkezés	Ár [Ft]	Jelentkezés
2002. december 10.	2	http://www.netacademia.net/workshop/PKI	140,000	... fő
2003. január 14.	**2**	**http://www.netacademia.net/workshop/PKI**	**140,000**	**... fő**

Részletekről érdeklődjön a 06/1/472-1214-es telefonszámon vagy az info@netacademia.net e-mail címen!

A tanfolyam helyszíne: Budapest 1062, Andrássy út 62.

PKI különszám?

Szerkesztőség
Főszerkesztő: Fóti Marcell
marcellf@netacademia.net
Főszerkesztő-helyettes: Fülöp Miklós
mick@netacademia.net
A szerkesztőség címe:
1062 Budapest, Andrássy út 62.
Tel.: 472-1214
technet@netacademia.net
Nyilvános levelezési lista:
tech.net@technetklub.hu

Kiadja és terjeszti a
NetAcademia Kft.
Terjesztési, előfizetési információ:
Tel.: 472-1214
terjesztes@netacademia.net
Megjelenik havonta, ára 1.344 Ft

Előfizethető megrendelőlevélben a szerkesztőségnél:
1062 Budapest, Andrássy út 62.
Fax.: 472-1215
http://technet.netacademia.net/subs

Hirdetésfelvétel: Szívós Éva
Tel.: 472-1214
Fax.: 472-1215
info@netacademia.net
Grafikai tervezés, kivitelezés:
Gregor László
Nyomdai előkészítés:
NetAcademia Kft.

Nyomda:
Hieron Kft.
2120 Dunakeszi, Tamás A. u. 11/a
Felelős vezető: Török Andrea

ISSN 1586-5185

Egész őszöm a PKI *(Public Key Infrastructure, nyílt kulcsú titkosítási módszerekre épülő infrastruktúra)* jegyében telt. Kezdődött egy országjáró körúttal, ahol – hogy, hogy nem – a Windows beépített biztonsági lehetőségeit, azon belül a nyílt kulcsú infrastruktúrával megvalósítható dolgokat adtuk elő a publikumnak. Majd folytatódott az egyik októberi tech.net konferenciával, ahol az Exchange-alapú levelezés biztonsága volt a téma – és a dolog megint a PKI-ba torkollott *(S/MIME, HTTPS stb.)*. Ezzel párhuzamosan Fülöp Miklós kollégám cikksorozatban emlékezett meg az RRAS virtuális magánhálózati képességeiről és az IPSecről. Csodák csodája, a PKI itt is felbukkant.

Ezek után azon kezdtem gondolkodni, nem érdemelne-e meg ez a téma egy picivel több figyelmet lapunk részéről, és arra a meglátásra jutottam, hogy de. Kiadtam az utasítást: novemberben arccal a PKI felé! És mi történt? Csak jöttek a cikkek, csak jöttek és jöttek, mígnem a tartalom több mint fele ezzel a témakörrel telt meg.

Van itt minden: CA-hierarchia cikk, kulcstárolás hardvereszközön és a Windows Protected Store-ban, PKCS-szabványok, IIS hitelesítés tanúsítványokkal, igaz történetek a Single Sign On kapcsán stb. Ennek ellenére nem tudtuk teljeskörűen kivesézni a témakört. Hiányzik például a .NET Serverbe épített CA újdonságainak elemzése, a .NET Assemblyk aláírásának témaköre és – erőforrás híján – most sem vesézzük ki teljes körűen a tanúsítványigénylés menetét, változatait, a Smart Card logont stb.

Reméljük, ezzel a kiadvánnyal hozzájárulunk a PKI bevezetését meggátoló legfőbb ellenérvek megválaszolásához, és egész kicsi hazánk ismét közelebb lép a fejlett országok technikai szintjéhez.

Valljuk be egymásnak: nem könnyű fogást találni a PKI-n. Íme néhány észérv, ami elgondolkodtatja, és sok esetben el is riasztja a potenciális kuncsaftokat:

· Tervezés nélkül nem érdemes bevezetni, mert később bizonyos változtatások nem hajthatók végre. Például az önhitelesített kulcsok utólagos hitelesítésére nincs mód.

· A kulcsok elvesztése a titkosított dokumentumok elvesztésével egyenlő. Melyik a nagyobb kockázat? Ha néha ellopnak egy-egy doksit, vagy ha mindet elveszítjük a kulcs elvesztése miatt?

· A fentiek miatt a kulcsok biztonságos tárolására mindenképpen külön eszköz *(Smart Card vagy USB token)* használata javasolt. Ha a merevlemezen tároljuk, nem tudjuk magunkkal vinni, és ráadásul a megsemmisülés esélye sem egy a végtelenhez.

· Már megint meg kell tanítani valamit a felhasználóknak.

Mit mondok erre én, a kívülálló? Azt, hogy induljunk el kicsiben! Vizsgáljuk meg, kinek válna ténylegesen hasznára a PKI, valamint kik azok, akiknek ismerniük kell a rendszert. Észre fogjuk venni, hogy egy-két embernek azonnal szüksége lenne digitális aláírás kibocsátási lehetőségre *(ügyfélszolgálat, titkárság)*, egy *(-két)* ember szeretne titkosított leveleket küldeni az Internetre - és itt vagyunk mi magunk rendszergazdák, akiket a dolog szakmai szempontból izgat. A pilot négy főt érint.

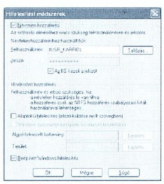
A következő gond: honnan szerezzünk X.509 tanúsítványt? Aki ismeri a Windows 2000 szolgáltatásait, tudja, hogy van benne egy Certificate Server, amely ingyen ad tanúsítványt. Na de milyet? Belső, azaz távoli partnerek által el nem ismert tanúsítványokat! A CertSrv beillesztése a világszerte elismert hitelesítési szervezetek láncába nem túl drága, és nem is túl bonyolult. Ám a hitelesítő szerver előírásszerű fenntartása már nehézkes dolog. De minek nekem CertSrv ahhoz a hat tanúsítványhoz, amelyre valóban szükségünk van? Pár ezer forintért tökéletesen elfogadott tanúsítványokhoz juthatunk!

Mihez kellhet mégis a Windows-féle Certificate Server? A Windows-bejelentkezés PKI-alapokra helyezéséhez. Ehhez elég a belső tanúsítványkiszolgáló által adott, a külvilág által el nem fogadott tanúsítvány is, hiszen a tartományi bejelentkezés belső művelet *(ellentétben mondjuk az S/MIME-mal)*.

Vágjunk bele!

Fóti Marcell
marcellf@netacademia.net

2002. 11. szám

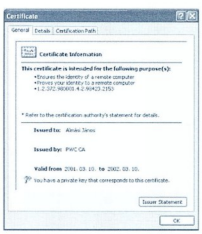


A Single Sign On, mint biztonsági rés 4. oldal

A hálózatok elterjedésének hőskorában minden szolgáltatásgyártó saját bejelentkeztetési módszert, ezzel önálló név-jelszó párost erőszakolt a felhasználókra. Három-négy ilyen rendszer párhuzamos használata már elképzelhetetlen a jelszavak feljegyzése nélkül; a legjobb, ha a monitorra ragasztgatjuk, úgy mindig kéznél vannak...

Tanúsítványok és magánkulcsok a Windowsban 6. oldal

Saját magánkulcsaink, valamint az általunk megbízhatónak minősített tanúsítványok biztonságos tárolása kritikus fontosságú. Amennyiben más is hozzáfér magánkulcsunkhoz, visszafejtheti a számunkra titkosított üzeneteket, vagy aláírhat a nevünkben. Ha rajtunk kívül bárki képes tanúsítvány elhelyezésére a megbízható tanúsítványtárunkban, azzal olyan aláírások elfogadására vehet rá minket, amit egyébként visszautasítanánk. Méghozzá úgy, hogy ez számunkra észrevétlen marad...



Intelligens kulcstartók – Az elektronikus aláíráshoz szükséges kódokat tároló miniszámítógépek 11. oldal

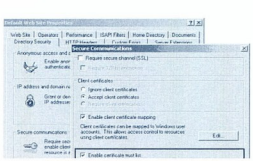
Miért fontosak az elektronikus aláírás készítésénél az úgynevezett kulcsok? Mik ezek pontosan és miért fontos a magánkulcs megfelelő védelme? Hogyan lehet a magánkulcsokat megfelelő módon kezelni és tárolni ahhoz, hogy biztonsági integritásuk megmaradjon? A következőkben bemutatjuk azon végfelhasználói eszközöket, amelyek segítségével biztonságos körülmények között, megbízhatóan tárolhatóak azon ún. aláírás-létrehozó adatok, amelyeket az elektronikus aláírások készítésénél alkalmazunk.

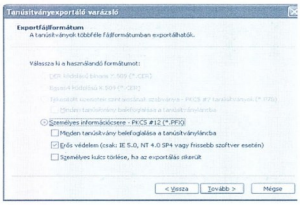
Hitelesítés-szolgáltatói hierarchiák–Tanúsítványláncok, tanúsítvány ellenőrzés bonyolult esetben 14. oldal

Az elektronikus aláíráshoz szükséges tanúsítványok kibocsátását mint a világ több más országában, hazánkban is piaci alapokon működő hitelesítés-szolgáltatók *(HSZ)* végzik. Ezen megbízható harmadik személyek rendelik össze az aláírás ellenőrző adatát jelentő nyilvános kulcsot és az entitás adatait a tanúsítványban, amit végül a hitelesség és megváltoztathatatlanság érdekében maguk is aláírnak. De vajon hogyan kapcsolódnak egymáshoz maguk a szolgáltatók és az általuk hitelesített tanúsítványok?

Tartományi felhasználók webes azonosítása a PKI segítségével 16. oldal

A Windows 2000 webkiszolgálója, az IIS 5.0 képes a hozzá érkező felhasználók tanúsítványalapú azonosítására is. A kiszolgáló elfogadhatja, illetve kérheti a felhasználóktól a tanúsítvány bemutatását, ami kiegészítheti, vagy kiváltathatja a „hagyományos" *(jelszavas)* felhasználóazonosítási módokat.



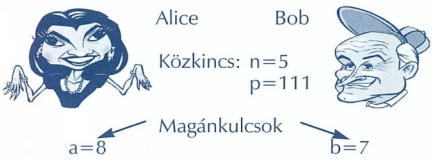


PKCS, X.509, RFC, ISO: mi micsoda a PKI-szabványok világában? 20. oldal

Talán a nyílt kulcsú algoritmusok után a második legnagyobb logikai útvesztő a technológiával kapcsolatba hozható szabványdzsungel megértése. Borzasztó látni, és tudni, hogy egy PKCS 10-es tanúsítványkérelemre egy RCF XXXX-be csomagolt X.509 -es választ kapunk, melyben a publikus kulcs az AIN.1-nek megfelelően tárolódik.

RRAS és PKI élesben 22. oldal

A tech.net magazin korábbi számaiban is több cikk jelent meg az RRAS eszközről, az IPSec, VPN és PKI technológiákról. Mindegyik cikk kiváló összefoglalója volt az adott témának. Most mindezen eszközök együttes alkalmazását szeretném megmutatni egy projekt megvalósításának folyamatán keresztül.



Titkos kulcscsere nyílt adatokkal: a Diffie-Hellman algoritmus (PKCS #3) 26. oldal

Mint azt a tech.net olvasói „elviselik", időről időre megjelenik egy-egy kriptográfiai algoritmus leírása. A mostani számunk szintén ilyen. A világ legelterjedtebb kulcscserélő algoritmusát, a Diffie-Hellman eljárást ismertetem.

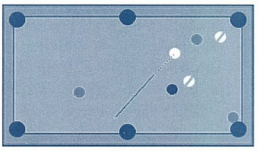
Távoli segítségnyújtás 28. oldal

Három évvel ezelőtt szorgos munka után a kollégáimmal sikerült egy teljesen automatizált Windows NT Workstation 4.0 telepítő környezetet létrehozni.Az operációs rendszert olyan kiegészítőkkel láttuk el, amelyek a termék megjelenése pillanatában még nem léteztek. Ilyen a WBEM, a WHS, a Time Service, és a VNC. Ezek közül a Windows 2000 majdnem mindent beépítve tartalmaz, A Windows XP azonban feltette a pontot az i-re, a távsegítség (Remote Assistance) szolgáltatás pótolta az utolsó hiányosságot is.

Farkasokkal táncoló XI.- Cluster a gyakorlatban 32. oldal

A fürtnapló elemzését talán nem mindenki találja izgalmasnak. Ez érthető, ellenben az olvasóközönség ezen rétegének mára sem tudok semmi jót ígérni. Akik viszont felvették Sherlock „rendszergazda" Holmes szerepét, dörzsölhetik a markukat *(vagy elégedetten pipázhatnak)*, mert mindjárt egy rejtély megfejtésével kezdjük. Pontosabban folytatjuk, hiszen ott tartottunk, hogy a nehezen összeszedett quorum.log-ot éppen töröltük.

Az UML – 2. rész 37. oldal



Az előző részben igyekeztem szemléletesen bemutatni az UML felépítését, nézeteit és diagramjait. Arról azonban nem szóltam, hogy az első ötleteléstől hogyan, mi módon juthatunk el a kész termékig. Ezúttal igyekszem pótolni ezt a hiányt, és megválaszolni a felmerült kérdéseket.

Net Academia MesterQrzusok 2003. tavaszi programja 42. oldal

MesterQrzus rendezvényünkön bepillantást nyerhetnek a NetAcademia oktatási tevékenységébe. Jöjjön el, legyen jelen legújabb cikkeink, workshopjaink, előadásaink születésénél!
A tavaszi előadások vendégelőadói értékes ajándékokkal is meglepik a kedves résztvevőket!

A Single Sign On, mint biztonsági rés

A hálózatok elterjedésének hőskorában minden szolgáltatásgyártó saját bejelentkeztetési módszert, ezzel önálló név-jelszó párost erőszakolt a felhasználókra. Három-négy ilyen rendszer párhuzamos használata már elképzelhetetlen a jelszavak feljegyzése nélkül; a legjobb, ha a monitorra ragasztgatjuk, úgy mindig kéznél vannak...

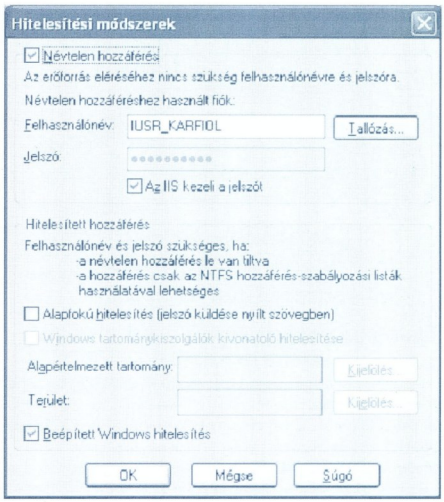
Ezen a biztonságilag igencsak áldatlan helyzeten segítenek a Single Sign On *(SSO)*, vagyis egyjelszavas rendszerek. De ne higgyük, hogy ettől a vállalati hálózatok biztonságosabbak lettek. Csak annyi történt, hogy a sárga cetlikről a hálózati forgalomba kerültek a mindenki által olvasható jelszavak. Miért?
Minden cég rendelkezik szellemi tőkével. Ha mást nem is takar ez a fogalom, csak a szerződésállományt, akkor is jelentős értékről van szó. Ma már az a szabály érvényesül, hogy ha egy cégnek csak egy szikrányi szellemi tőkéje is van, azt biztosan számítógépen *(is)* őrzi.
Az SSO, vagyis a hitelesítési adatok egységesítése és közös ellenőrzése határozottan jó ötletnek tűnik. Az egyedi, kétes biztonságú "címtárak" (néha csak egy textfájl) és algoritmusok (XOR :-) helyébe egy neves gyártó bizonyítottan biztonságos megoldása kerül. A Windows-világban ilyen az Active Directory - Kerberos páros. Amiről megfeledkeztünk, az az autentikációs szolgáltatáshoz csatlakozó többi alkalmazás biztonsága. Igenis, megfeledkeztünk! Önök is! Ha ugyanis minden felhasználónak egyetlen neve és jelszava van, nyilvánvaló, hogy minden esetben ezt fogja használni. Más választása ugyanis nincs.

1. SSO-esettanulmány

Kőmíves Kelemen, a Falatrax Kft. üzletkötője rendszeresen fél napokat tölt a megrendelőknél, mert együtt dolgozzák ki az építkezés részleteit. Ha hirtelen kell neki egy dokumentum, az anyacégtől emailben elküldik neki. Ilyenkor Kelemen megkér valakit, hogy engedje oda a gépéhez, amíg ő Outlook Web Accessel elolvassa a levelet. Egyszer csak a partnercég egyre tájékozottabbnak tűnik Kelemen levelezésével kapcsolatban. Mi több, egy hónap múlva az összes futó szerződést mondvacsinált okokkal, azonnali hatállyal felbontják, és az árakat a nyereségesség alsó határáig lenyomva újraköttetik Kelemennel. Szegény Kőmíves Kelemen! A fizetése függött a nyereségességtől!
Mi történhetett?
A partnercég rendszergazdája a tűzfalon HTTP-logot készít. Az informatikus beállítottságú főnök hetente átnézi a logokat, hogy megállapítsa, a ki-be menő forgalom hány százaléka pornó. Ha gyanús az URL, megnyomja a "Detailed info..." gombot, illetve ha nagyon kíváncsi, ellátogat az adott címre. Így bukkant a protokollnaplóban az OWA URL-re, és Kelemen nevére, jelszavára. Próba, szerencse. Itt a hacker, hol a hacker!

IIS autentikációs szintek - melyik ujjamat harapjam?

Adjunk tanácsot a Falatrax rendszergazdájának, hogy befoltozhassa az OWA körül tátongó biztonsági rést, képes legyen más jelszótitkosítást is nyújtani az ügyfeleknek. Soroljuk fel az IIS hitelesítési lehetőségeit, hadd válasszon közülük!



Az IIS felhasználóazonosítási módszerei

1. **Névtelen** ***(anonymous).*** Ez az azonosítás jó lenne abból a szempontból, hogy név és jelszó nem kerül a kábelre, de sajnos névtelenül nem tudunk csatlakozni a saját postafiókunkhoz sem, így le kell tennünk erről a módszerről.
2. **Alapfokú** ***(basic).*** Ez a hagyományos titkosítatlan, Clear Text jelszóküldés. Nemcsak a hálózati forgalom monitorozásával, hanem mint láttuk, részletesebb tűzfalnaplóval is össze lehet gyűjteni a jelszavakat. Ez sem jó módszer.
3. **Kivonatoló** ***(digest).*** Ez a szabványos azonosítási módszer már nem a titkosítatlan jelszót, hanem csak egy abból készített hasht visz át a hálózaton. Gyönyörű megoldás lenne, ha ez a hash megegyezne az Active Directoryban használttal. De nem egyezik meg. Így sajnos csak akkor működik, ha az AD-ban a felhasználónál engedélyezzük a visszafejthető jelszótárolást. Én nem tennék ilyet. Ez a módszer sem jó tehát.
4. **Beépített Windows** ***(Integrated Windows).*** Végre valami ütős! NTLM, illetve Kerberos! Feltörhetetlen! És használhatatlan. Nincs tűzfal, amit úgy állítanának be, hogy az NTLM átmehessen rajta. A Kerberos egyes helyeken még csak-csak átjut, de nem tartománytag gépekről egyszerűen nem lehet használni. Ez a módszer sem választható.

Négyből semmi. Nem rossz arány ugye? Az egyetlen kiút a kelepcéből az lenne, ha nem az autentikációt tuningolnánk,

hanem a gépek közötti teljes adatforgalmat titkosítanánk. Vagy lenne egy ötödik azonosítási módszer. Van is! PKI-alapú! Erről néhány lappal hátrébb, az IIS és az X.509 tanúsítványok összegyűrását taglaló cikkünkben olvashatnak. Nehéz ügy...

Single Sign On, mint biztonsági rés?

Nagyon szomorú, de igaz: az SSO miatt nem egy akármilyen név+jelszó került napvilágra, hanem Kelemen tartományi jelszava! Ha ezzel valaki bejuthatna a Falatrax hálózatára...! És a krimi folytatódik:

A szerződések újratárgyalásához további információkra volt szükség. A partnercég megbízta az e-Kém *(ejtsd: i-kém, azaz ipari kém)* nevű céget, akik úgynevezett Social Engineering módszerrel (és Kelemen jelszavával) gyakorlatilag a Falatrax teljes adatmennyiségét határidőre leszállították. *(CD-re mentve.)* Hátborzongató, de igaz: az e-Kém módszerei közé tartozik a próbaidőztetés. A kiszemelt áldozatcég meghirdetett álláshelyeire jelentkeznek az ügynökök, és ott próbaidős felvételt nyernek. Lesz loginjük (még ha gyenge is), számítógépük stb. A mi esetünkben kapásból rendelkezésre állt egy erősebb jelszó is - Kelemené. A többit találják ki.

2. SSO-esettanulmány avagy Az Ember, Aki Képes EFS-t Olvasni

A Falatrax Kft. vezetőjének, Fontos Gyulának van egy laptopja, ezzel jár tárgyalni, mert így minden dokumentum mindig kéznél van. Szakértői tanácsára a fontosabb adatokat egy EFS-sel titkosított mappában tárolja, mert így biztos lehet benne, hogy még ha el is lopják a gépet, akkor sem olvashatják el illetéktelenek a fájlokat. Nemrégiben annál a cégnél járt, akivel párhuzamosan pályáznak egy nagyobb tenderen. Miután ott járt, a másik cég új pályázati anyagot nyújtott be, melyről a tenderbontáskor kiderült, hogy 1%-kal olcsóbb árat adott mindenre, mint a Falatrax.

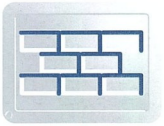
Gyula esküszik, hogy egy pillanatra sem hagyta magára bejelentkezve a laptopot. Valakik valahogy mégis elolvasták a palyazat.xls-t. Nyilván nem Gyula gépéről, mert ott titkosítva van minden. Vagy mégis? Kérdezzük ki Gyulát, hogy is zajlott az a nap?

> „Többször megszakítottuk a munkát, kávészünet, ez-az. De olyankor mindig kijelentkeztem, ahogy a rendszergazda tanította. Ha igaza van a srácnak, akkor senki nem fért az adatokhoz."

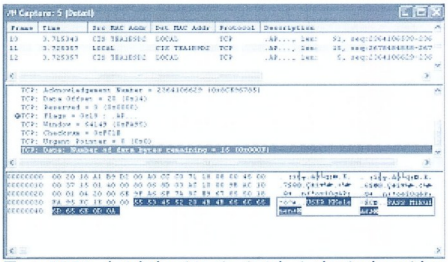
A laptopon külsérelmi nyom nincs, arra pedig senkinek nem lehetett ideje, hogy ki - és visszaszereljen egy alkatrészt. Mire használta a helyszínen a laptopot?

> „Dokumentumokat és táblázatokat nyitottam meg. És elolvastam a leveleimet. Mivel sürgős levelet vártam, kértem egy üres Ethernet csatlakozót, majd – Kelemen múltkori esetén okulva – nem OWA-val, de nem is Outlook Expressel, hanem a NAAAGY Outlookkal olvastam el a leveleimet."

Talán itt a hiba? Talán itt. Az Interneten keresztül általában nem MAPI, hanem POP3 protokollal olvasunk levelet, egyszerűen azért, mert ezt átengedik a tűzfalak. A NAAAGY Outlookot is egy POP3 csatolóval felvértezve tudjuk távolról használni. *(De már nem sokáig van ez így. Jön az ISA Server Feature Pack, benne RPC-szűrővel, és mehet a MAPI! - a szerk.)* Közismert, hogy a POP3 legalacsonyabb szintű „jelszótitkosítási" algoritmusa a Clear Text, vagyis a semmi. Ennél magasabbra menni kompatibilitási okokból nem szokás. (Van ugyan AUTH parancs a POP3-ban, de azt nem használjuk.) Mi történt tehát?



Ez alkalommal készült a partner, és rendszergazdáját ráállította a hálózatra, próbálja begyűjteni Gyula azonosító adatait, ha egy mód van rá! Nosza, feldobunk egy Network Monitort a tűzfalra (azért oda, mert ott minden csomag áthalad), és belehallgatunk a forgalomba. Mit látunk?



A POP3 bejeletkezési név és jelszó alapértelmezésben olvasható formátumban utazik a hálózaton

Gyula bejelentkezési nevét és jelszavát sodorja az ár. Ha kihallásszuk, a mienk lehet. Ha pedig egyszer a birtokunkban van, nyílik vele a laptop is. Be tudunk jelentkezni a laptopra, kinyílnak az EFS-sel titkosított fájlok is.

Van megoldás?

Természetesen az eddigiekkel nem azt akarom mondani, hogy a korábbi, felhasználónkénti hatszáz jelszavas káosz üdvözítőbb, csak fel szerettem volna hívni a figyelmet olyan biztonsági összefüggésekre, melyekre nem feltétlenül gondolunk, s így akaratlanul kiadjuk értékes tartományi azonosítóinkat másoknak. SSO használatakor tehát figyelembe kell venni, hogy az egyetlen jelszó nagy kincs! Talán nem is szabadna emberekre bízni a használatát, mert csak elkótyavetyélik. Tulajdonképpen régesrégen eljött az ideje a jelszóalapú bejelentkezés kiküszöbölésének, mert napjaink számítógépeivel bizony fél nap alatt visszafejthető egy-egy átlagos hosszúságú jelszó.

A jelszavaknál lényegesen biztonságosabb módszer például piktogram-logon. Képzeljük el, hogy bejelentkezéskor a név megadása után a képernyőn mintegy hatszáz különböző képecske jelenik meg (20 sorban és 30 oszlopban), és ezeken a megfelelő sorrendben kell kattintani. Olyasmi, mint egy jelszó (vagy inkább "jelkép"?), de ABC-je nem pár tíz, hanem hatszáz jelből áll, hossza pedig lényegesen meghaladhatja a szokásos 6-8 karaktert. Feltöréséhez, visszafejtéséhez nagyságrendekkel több erőforrás szükséges.

A csúcs természetesen a SmartCard Logon. Hány "karakter" hosszú a jelszó 1024 bites RSA-kulcs esetén? 1024 / 8 = 128 bájt. (Itt némi egyszerűsítéssel éltem, azt feltételezvén, hogy a publikus kulcs maga a jelszó. Ez nem így van, de ennyi csúsztatás még "belefér".) Mennyire bonyolult a jelsor? Ennél bonyolultabb nemigen lehet, minthogy egy önálló értelemmel nem bíró bitsorozat. (Szemben a jelszóként megadott családtagnévvel.)

Már csak egy feladatunk van: ne hagyjuk el a kártyát!

Fóti Marcell
MZ/X

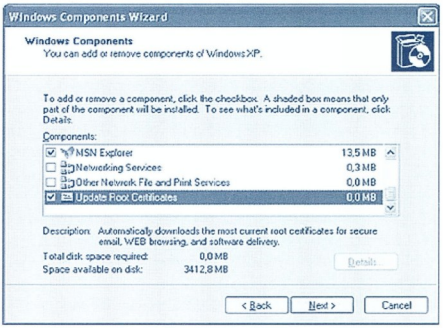
Tanúsítványok és magánkulcsok a Windowsban

Saját magánkulcsaink, valamint az általunk megbízhatónak minősített tanúsítványok biztonságos tárolása kritikus fontosságú. Amennyiben más is hozzáfér magánkulcsunkhoz, visszafejtheti a számunkra titkosított üzeneteket, vagy aláírhat a nevünkben. Ha rajtunk kívül bárki képes tanúsítvány elhelyezésére a megbízható tanúsítványtárunkban, azzal olyan aláírások elfogadására vehet rá minket, amit egyébként visszautasítanánk. Méghozzá úgy, hogy ez számunkra észrevétlen marad...

...mindaddig, míg nem nézzük át tüzetesen a tár tartalmát vagy addig, amíg nem kerülünk bajba e visszaélés miatt. A Windows esetében az ilyen típusú visszaélések mindazok számára lehetségesek, akik nevünkben vagy adminisztrátorként képesek bejelentkezni gépünkre. Éppen ezért nem árt, ha felhasználóként, rendszeradminisztrátorként vagy fejlesztőként megismerjük a Windows tanúsítvány- és magánkulcs-tárolási szokásait.

Windows tanúsítványtár

A Windows saját tanúsítvány-nyilvántartással *(Certificate Store)* rendelkezik, melyben saját tanúsítványaink, partnereink tanúsítványai, a hitelesítésszolgáltatók *(gyökér és kibocsátó)* tanúsítványai, a megbízhatatlannak minősített tanúsítványok, a szoftverhitelesítők tanúsítványai, visszavonási listák és tanúsítványbizalmi listák kapnak helyet. A tanúsítványok mellett itt tárolódnak a tanúsítványhoz felvehető tulajdonságértékek is. Ez a nyilvántartás frissen telepített operációs rendszer esetében is tartalmaz tanúsítványokat, méghozzá a Microsoft által megbízhatónak tartott szolgáltatók tanúsítványát. Némi óvatosság elkél ennek kezelésekor, mert a Microsoft korábban nem volt véresen szigorú az ide történő bekerüléssel kapcsolatban *(az idei évtől a bekerüléshez viszont már független auditori vizsgálati jelentést követel meg)*. Nagy gyengesége a listának, hogy vegyesen, minden megkülönböztetés nélkül tartalmaz teljesen eltérő erősségű szolgáltatásokhoz kapcsolódó szolgáltatói tanúsítványokat. Mindenkinek azt javaslom, hogy nézze át a listát, és nyirbálja meg, csak azokat a szolgáltatókat meghagyva, akiket valóban ismer, nekik is csak azokat a szolgáltatásukat *(tanúsítványosztályukat)*, melyek garantálják az általa megkövetelt biztonságot. A nyilvántartásban eleve szerepel két megbízhatatlan tanúsítvány is, amely a Verisign által tévedésből kiadott, a Microsoft nevére szóló tanúsítványokat takarja. A lista automatikus frissítése XP-n a Windows Component Wizards-ban *(Control Panel – Add or Remove Programs – Add/Remove Windows Components)* az Update Root Certificates opció segítségével állítható be. A Windows 2000 és a régebbi operációs rendszerek esetében a felhasználók a Windows Update website segítségével tehetik meg ezt manuálisan. A lista az Active Directory használata esetén csoportházirenddel is bővíthető. Fejlesztők figyelmébe ajánlom további lehetőségként az Internet Explorer Administration Kitet, a CAPICOM scriptelési lehetőséget és a CertEnroll vezérlőelem weblapról történő meghívását.



Gyökérszolgáltatók listájának frissítése

A nyilvántartás később saját, partnereink és az általunk megbízhatónak minősített hitelesítés-szolgáltatók tanúsítványaival szaporodik, amiket vagy Interneten keresztül töltünk le, vagy explicit beimportáljuk őket.

A tanúsítványtár a következő logikai egységekre tagozódik *(Windows 2000 esetében kevesebb logikai egység létezik)*:

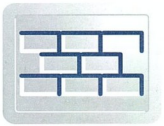
- Personal: Ez a saját tanúsítványok könyvtára, ahol olyan tanúsítványok tárolódnak, melyek magánkulcsával is rendelkezünk. Tipikusan saját részre vagy a felügyeletünk alatt álló számítógépek és alkalmazások részére kibocsátott tanúsítványok. Az itt tárolt tanúsítványok mellett a hozzájuk tartozó magánkulcsra és kriptográfiai szolgáltatóra *(CSP)* is van utalás.
- Third-Party Root Certification Authorities: Megbízhatónak minősített külső hitelesítés-szolgáltatók gyökértanúsítványai.
- Trusted Root Certification Authorities: Az előbbi tár kiegészítve a Microsoft valamint saját szervezetünk hitelesítés-szolgáltatásának gyökértanúsítványaival.
- Enterprise Trust: A szervezet tanúsítvány bizalmi listáinak *(CTL-ek)* könyvtára.
- Intermediate Certification Authorities: Kibocsátó hitelesítőegységek tanúsítványainak könyvtára.

- Trusted People: Olyan nem szolgáltatói tanúsítványok listája, melyek tipikusan önaláírtak, s ezért explicit bizalmat kellett szavaznunk számukra *(például ide kerülnek az EFS tanúsítványok).*
- Other People: Olyan nem szolgáltatói tanúsítványok listája, melyek hitelesítés-szolgáltató által aláírtak, s ezért implicit megbízhatóak *(például ide kerülhetnek levelezőpartnereink tanúsítványai).*
- Trusted Publishers: Az általunk megbízhatónak minősített szoftverhitelesítők tanúsítványai.
- Untrusted Certificates: Megbízhatatlannak minősített tanúsítványok *(általunk vagy a Microsoft által).*
- Certificate Enrollment Requests: Még el nem bírált vagy visszautasított tanúsítványkérések.
- Active Directory User Object: A felhasználó Active Directoryban publikált tanúsítványainak logikai képe.

A tanúsítványtár különböző szegmensekre is tagozódik. A gépen felhasználói azonosítóval rendelkező valamennyi felhasználónak, szerviznek és magának a gépnek is van egy szegmense, amin belül kis eltérésekkel az előbb felsorolt logikai egységekkel találkozhatunk. Egyszerűbben megérthető, ha kétdimenziós táblázatként képzeljük el a tárat: az egyik dimenzió a szegmenseké, a másik a logikai egységeké.

A logikai egységek jelentése és tartalma alapesetben megegyezik minden szegmens esetében. A Personal könyvtár természetesen mindenhol az adott felhasználóra, gépre, szervizre vonatkozik, és tartalma csak addig ugyanolyan mindenhol, ameddig üres. Amennyiben felveszünk vagy kitörlünk egy tanúsítványt az egyik szegmens egyik logikai egységéből, az általában nincs hatással egy másik szegmens *(pl. egy másik felhasználó vagy egy másik szerviz)* ugyanazon logikai egységére. Azért csak általában, mert van egy kivétel, mégpedig a gép szegmense, azon belül is a szolgáltatói tanúsítványok logikai egységei *(különböző Certification Authorities és Enterprise Trust egységek).* E logikai egységek beállításait a felhasználói szegmens ugyanezen logikai egységei is átveszik. Ez azt jelenti némi példával szemléltetve, hogy: ha a személyes szegmensünk gyökértanúsítványai közül töröljük az egyik hitelesítés-szolgáltató tanúsítványát, attól még egy másik felhasználó vagy egy szerviz, valamint maga a gép is, még meg fog bízni ebben a szolgáltatóban; ha viszont a gép gyökértanúsítványai közül töröljük az egyik hitelesítés-szolgáltató tanúsítványát, a többi felhasználó sem fog megbízni ebben a szolgáltatóban. Az MMC segítségével ezt magunk is kipróbálhatjuk. Csak arra kell vigyázni, hogy az MMC ezt az öröklést nem mutatja. Ennek hatása csak a Certificates ablakban fedezhető fel.

A belső tanúsítványtár fizikailag három különböző helyen tárol tanúsítványokat: a Registryben, állományban a lemezen és csoportháziendben *(ez utóbbi esetben a hálózati tartomány felhasználóira együttesen vonatkoztatva).* A fizikai tárolás megjelenítésével láthatóvá válik, hogy melyik tanúsítvány hol tárolódik e fizikai tároló helyek közül *(nem árt azonban fenntartásokkal kezelni az itt megjelenítetteket, mert saját tapasztalataim ellentmondanak az MMC által mutatott adatoknak).* A Registryn belül leginkább a HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\SystemCertificates alatt találhatunk tanúsítványokat *(alapos böngésző még további helyeken is),* a fájlrendszeren belül, pedig system fájl formátumban a \Documents and Settings\felhasználó\Application Data\Microsoft\SystemCertificates könyvtárban.



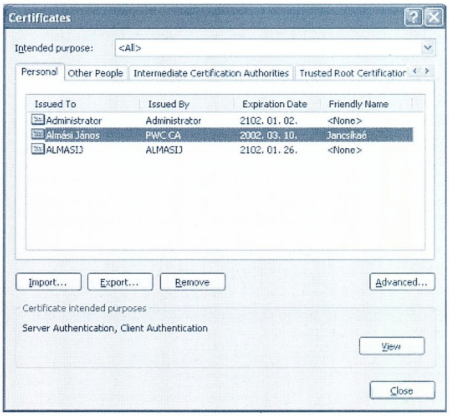
Magánkulcsok tárolása

A magánkulcsokat a Windows egy úgynevezett védett tárban (protected store) tárolja. A védelem itt egy rejtjeles tárolási formát jelent, amiről a CryptoAPI gondoskodik. A kulcsok ténylegesen a \Documents and Settings\ felhasználó\Application Data\Microsoft\Crypto\ RSA könyvtárban találhatók.

A magánkulcsok és tanúsítványok részei a felhasználói profilnak, s ha vándorló profilt használunk, ki- és bejelentkezéskor átmásolódnak a profilgazda gépre. Ugyanígy a menet közben szerzett kulcsok és tanúsítványok a profil részként kijelentkezéskor a profilgazda gépre másolódnak. Ez a kényelem a másik oldalon gyengeség, hiszen izgőmozgó felhasználóként magánkulcsunkat számos gépen otthagyjuk, ami jelentősen megkönnyíti annak rosszhiszemű megszerzését.

Tanúsítványok menedzsmentje – Certificates ablak

A belső tanúsítványtár felhasználói szegmenséhez hozzáférést biztosító Certificates ablak a Control Panel – Internet Options – Content – Certificates útvonalon keresztül érhető el (vagy Internet Explorer-ből, a Tools – Internet Options… menüpontokon keresztül). Ebben az ablakban mindig a bejelentkezett felhasználóra vonatkozó tanúsítványokat látjuk, a gép vagy egyes szervizek tanúsítványainak megjelenítésére itt nincs mód *(más felhasználó szegmensének megtekintéséhez be kell jelentkezni az ő nevében).*



Tanúsítványok kezelése a Certificates programmal

Megfelelő jogosultság esetén lehetőségünk van tanúsítványok fájlból történő importálására vagy oda történő exportálására, valamint tanúsítványok törlésére a tárból. Exportálni drag and drop módszerrel is lehet a kívánt tanúsítványt az Intéző ablakába mozgatva. Az ekkor használatos állományformátumot az Advanced opcióknál adhatjuk meg. A tanúsítványt a View paranccsal nézhetjük meg. A

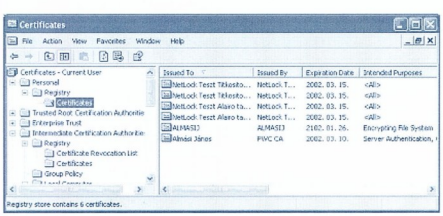
megjelenített tanúsítványokat felhasználási cél szerint szűrni tudjuk *(Intended purpose)*. Visszavonási listák importálására is módunk van itt, de az exportálás nem lehetséges, minthogy a visszavonási listák ebben az ablakban nem jelennek meg.

Tanúsítványok menedzsmentje – MMC

A tanúsítvány-nyilvántartás elérhető a Microsoft Management Console felületén keresztül is. Ehhez a Certificates modult kell a konzol felületére varázsolnunk, amit a \WINDOWS\system32\certmgr.msc sablonfájl betöltésével vagy az Add/Remove snap-in paranccsal *(Standalone fül alatt Add... gomb)* érhetünk el. Ez utóbbi esetben meg kell határoznunk, hogy a személyes tanúsítványainkhoz vagy egy szerviz-, illetve egy gép saját tanúsítványaihoz akarunk hozzáférni *(a fájl betöltésével csakúgy, mint a Certificates ablakon keresztül, csak személyes tanúsítványainkhoz férünk hozzá)*. Gépet választva megadhatjuk, melyik gép, szervizt választva pedig hogy melyik gép melyik szervizének tanúsítványai iránt érdeklődünk. Másik felhasználó tanúsítványaihoz ilyen módon nem férünk hozzá, ahhoz az ő nevében kell bejelentkeznünk.

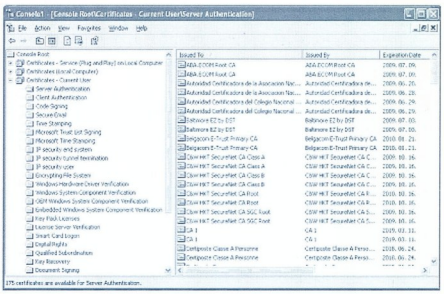
Az MMC összetettebb megtekintési és kezelési módszert biztosít a tanúsítványok számára, mint a Certificates ablak. A View menü Options ablakán *(mely az MMC bal oldali fastruktúrájának valamely felsőbb szintű elemén állva érhető csak el)* beállítható, hogy a tanúsítványok fizikai tárolóhelye is megjelenjen, továbbá mód van az archív (már lejárt érvényességű) tanúsítványok megjelenítésére is. Az MMC ettől függetlenül is képes más képet mutatni a tanúsítványtárról, mint a Certificates ablak. Ennek egyik oka, hogy az MMC az öröklési jellegű szabályokat nem mutatja *(a gép gyökértanúsítványai közül törölt elemeket megjeleníti a felhasználó tárában is, pedig azok nem élnek ott)*, de praxisomban találkoztam olyan jelenséggel, hogy Certificates ablak nem mutatott létező, a Personal könyvtárt gazdagító tanúsítványokat, azok csak az MMC-vel és más alkalmazásokkal váltak láthatóvá.

A jobb egérgomb hatására megjelenő menü Find Certificates menüpontján keresztül keresni lehet a tanúsítványok között. Ugyanezen menü All Tasks almenüjén keresztül mód van tanúsítványok exportálására és importálására, valamint a fizikai tárolóegység szintjén állva a tárolóegység adott szintjének Microsoft Serialized Certificate Store formátumban történő exportálására *(az Export List parancs csak a képernyő jobb oldali paneljának adatait írja ki szöveges formátumban, a tanúsítványok tartalmát nem)*.



Tanúsítványok menedzsmentje MMC-vel

Az Options ablakban beállítható az is, hogy a tanúsítványok ne az alapértelmezés szerinti logikai tárak, hanem a felhasználás célja szerint rendezve jelenjenek meg.



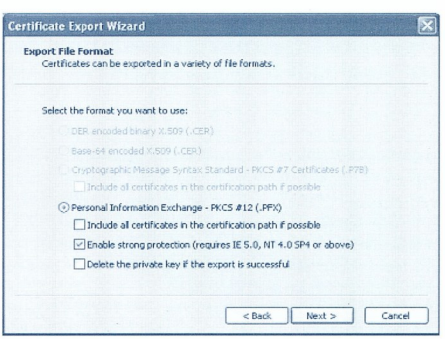
Tanúsítványok megjelenítése a felhasználás célja szerint rendezve

Az egyes tanúsítványok a tárolók között a szokásos vágólap-parancsok *(CTRL-C, CTRL-V, stb.)* és drag and drop műveletek segítségével másolhatók, mozgathatók, ennek gyakorlását azonban csak annak ajánlom, aki biztos annak következményeiben, amit tesz.

Tanúsítványok importálása és exportálása

A tanúsítványok belső tanúsítványtár kapcsán értelmezett *(onnan, illetve oda történő)* exportálása és importálása több helyről is kezdeményezhető *(MMC, Certificates és Certificate program, Intéző – Install Certificate és Install PFX parancsok stb.)*. A művelet minden esetben ugyanazon varázslón keresztül történik. A kiválasztott állományformátumtól függően egy lépésben megoldható egy vagy több tanúsítvány mozgatása. Importálásnál megadható a tanúsítvány tárolásának logikai és fizikai helye *(utóbbi a Show physical stores opció beállításával)*. Amennyiben a magánkulcs importálására is sor kerül *(PKCS #12 és SST formátumok)* meg kell adni a magánkulcs aktivizáló adatát, valamint annak belső tárolására vonatkozó szabályokat *(azt, hogy a magánkulcs később exportálható legyen-e, s használata során minden esetben figyelmeztessen-e, illetve jelszót kérjen)*.

Exportálásnál megadhatjuk, hogy a magánkulcsot is exportálni akarjuk-e a tanúsítvánnyal együtt. Erre akkor van lehetőségünk, ha a magánkulccsal rendelkezünk, a kriptográfiai szolgáltató támogatja a kulcsexportot *(hardveres CSP-k nem támogatják)*, az export nincs letiltva és van hozzáférési jogunk a kulcshoz. Magánkulcsunk exportját lehetőleg ne engedélyezzük, mert ezt kihasználva illeték-



Tanúsítványexportálási formátumok

telenek is másolatot készíthetnek kulcsunkról! Az exportáláskor a varázsló különböző tanúsítványtárolási formátumokat ajánl fel. A formátumtól függően választhatunk a teljes lánc és a tanúsítvány önmagában történő exportálása között. Amennyiben a magánkulcs is exportálásra kerül, beállíthatjuk annak erősebb védelmi szintjét és azt, hogy ezek után a magánkulcs törlődjön-e a védett tárból.

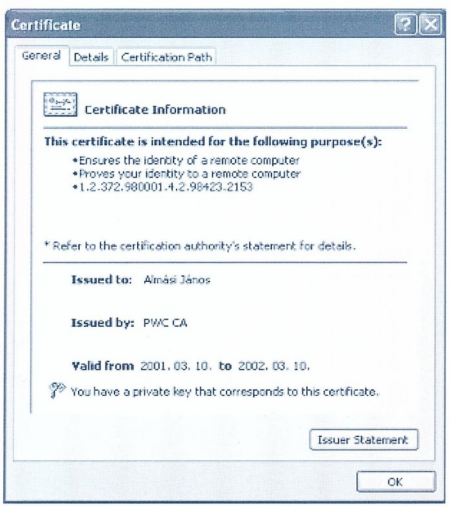
A Certificate ablak

A tanúsítványok tartalmának megjelenítését a Windows egy három–öt füllel rendelkező ablakban végzi. Az első fül alatt *(főoldalon)* a legfontosabb információk kivonata található. A tanúsítvány felhasználási célja *(a kulcshasználat, kiterjesztett kulcshasználat és Netscape tanúsítványtípus szabványos toldalékok alapján)*, a hitelesítési szabályzat OID-ja, a tanúsítvány tulajdonosának és kibocsátójának neve *(az azonosítók Common Name mezője alapján)*, a tanúsítvány érvényessége és az, hogy a rendszer ismeri a tanúsítvány magánkulcs párját.

A bal felső sarokban található tanúsítványszimbólum kinézete jelentést hordoz. Alapformában azt jelzi, hogy a tanúsítvánnyal nincs probléma. Egy piros körben megjelenő fehér kereszt feltűnése arra utal, hogy a tanúsítvány nem érvényes *(mert például lejárt)*, vagy a kibocsátó szervezet nincs benne a megbízható hitelesítőszervezetek listájában. A kis sárga háromszögben feltűnő felkiáltójel azt jelenti, hogy a Windows nem képes a tanúsítvány ellenőrzésére *(mert például nem rendelkezik a hitelesítőszervezet tanúsítványával)*. Ezen állapotokra vonatkozóan szöveges információ is megjelenik a szimbólum alatt *(a felhasználási cél helyett)*.

Az Issuer Statement gomb automatikusan megjeleníti a szolgáltató szabályzatát *(melyre a Certificate policies toldalék mutat)* egy böngészőablakban.

A második fül alatt a tanúsítvány mezőinek tételes tartalma jelenik meg. Ha a tanúsítvány tulajdonosának vagy kibocsátójának neve nem volt egyértelmű, itt a teljes azonosító megtekinthető. Ugyanígy részletesebb *(óra és perc)* adatokkal van feltüntetve az érvényességi idő is, valamint itt lehet megtudni az algoritmus típusát és a kulcs erősségét.



A tanúsítvány főbb adatinak megjelenítése

A mezőnév mellett baloldalt szereplő jel mutatja, hogy standard mezőről vagy toldalékról van-e szó, és a toldalék kritikus *(felkiáltó jel sárga háromszögben)* vagy sem *(zöld lefele mutató nyíl)*. A tulajdonságként feltüntetett mezők és értékeik olyan adatok, amelyek nem részei a tanúsítványnak, hanem a belső tanúsítványtárban mellette tárolódnak, vagy belőle következnek *(például lenyomat)*. Egy adott mezőt kiválasztva a képernyő alsó felében részletesen megjelenik a mező tartalma.

Ha a Windows nem ismeri a tanúsítványba foglalt toldalékot, akkor annak OID-ját írja ki, ellenkező esetben a mező nevét.

A harmadik oldal a tanúsítási lánc felépítését mutatja és a tanúsítvány állapotát jelzi *(a lánc azon tanúsítványának állapotát, amin állunk)*. Ha a tanúsítvány lejárt, vagy más probléma van vele, itt nem a „tanúsítvány rendben" szöveg jelenik meg.

A negyedik és ötödik fül opcionális, megjelenésük a tanúsítvány típusától függ. A negyedik, Trust fül opciói segítségével egyes tanúsítványokra lehet megadni, hogy azokban egyedileg is megbízzunk *(akkor is, ha a kibocsátójában egyébként nem bízunk meg)*, vagy pont fordítva. Ez az információ a tanúsítvány mellett tárolódik a fájlban vagy a levélben *(a tanúsítványtárban ilyen beállítási lehetőség nincs)*. Az ötödik fül a tanúsítvány Address Bookban való rögzítésére ad lehetőséget.

A második oldalon található Copy To File gomb segítségével a tanúsítványt különböző formátumú fájlokba exportálhatjuk. Az ugyanitt elérhető Edit Properties gomb hatására a tanúsítványtulajdonságok jelennek meg, ahol különböző beállítási lehetőségeink vannak a kérdéses tanúsítványra vonatkozóan. Eltárolhatunk a tanúsítvány mellett egy barátságos nevet, leírást írhatunk hozzá, és korlátozhatjuk a tanúsítvány kiterjesztett kulcs használat toldalékában feltüntetett felhasználási célokat. Akár le is tilthatjuk az összes ilyen felhasználási célt, s így az ezt ellenőrző alkalmazások számára megakadályozhatjuk a tanúsítvány elfogadását. Az Add Purpose gomb segítségével felhasználási célokat rendelhetünk azon tanúsítványokhoz, melyek nem rendelkeznek Kiterjesztett kulcshasználat toldalékkal.

Ugyanezen képernyő második fülén a kereszthitelesítéssel kapcsolatban határozhatunk meg attribútumokat. Ha a tanúsítványban nem szerepelne a kereszthitelesítésre vonatkozó cím, vagy a benne szereplő mellett újat is fel kellene venni, akkor az itt tehető meg, a kereszthitelesítő tanúsítvány ellenőrzési intervallumának megadásával együtt.

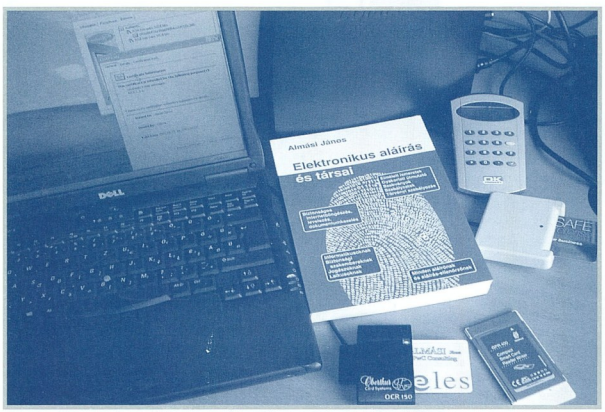
Almási János
janos.almasi@hu.ibm.com

Könyvbemutató

Almási János: Elektronikus aláírás és társai

2002. októberében egy hiánypótló szakkönyvet jelentett meg a Sans Serif kiadó. Almási János Elektronikus aláírás és társai című könyvének témája a címben is szereplő elektronikus aláírás mellett a titkosítás, az időbélyegzés, a felhasználói hitelesítés, a nyilvános kulcsú infrastruktúra és még sok más. Azok a védelmi megoldások, melyek az Internet világában a biztonságot jelentik. Az elméleti ismeretek mellett gyakorlati műszaki útmutatót is ad, s áttekinti a téma szabványait, törvényi szabályozását és a szabályzati kérdéseket.

A könyv gyakorlati példái bemutatják az elektronikus aláírás és titkosítás felhasználását az elektronikus levelezés, böngészés, fájl- és dokumentumkezelés kapcsán. Olyan általánosan használt alkalmazásokon keresztül teszi ezt, mint a Windows és az Internet Explorer, valamint a Microsoft Office termékcsalád.



Elméleti vonatkozású részei felölelik a kriptográfia, az algoritmusok, a protokollok, a tanúsítvány, a visszavonási lista, a hitelesítés-szolgáltatók és az aláíró eszközök területét. Nemcsak a hazai, de a nemzetközi könyvpiacon is olyan egyedülálló könyvről van szó, amely átfogó és mély ismereteket ad az elektronikus aláírás, tágabb értelemben, pedig az elektronikus biztonság témakörében.

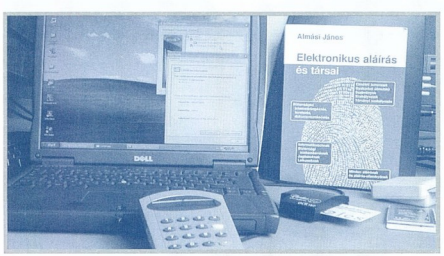
A könyv elsősorban informatikusoknak, biztonsági szakembereknek és jogászoknak szól. Könnyen érthető szövegezése, logikus felépítése és egyszerű példái miatt azonban minden olyan laikus felhasználó számára érdekes és hasznos lehet, aki biztonságos kommunikációra vágyik.

Az elektronikus aláírás és kriptográfia iránt érdeklődő szakemberek közül különösen azok a tervezők, fejlesztők, rendszergazdák és tanácsadók forgathatják nagy haszonnal, akik biztonságos alkalmazásokat kívánnak alkotni, illetve ilyen megoldásokat vezetnek be, vagy ilyen rendszereket üzemeltetnek.

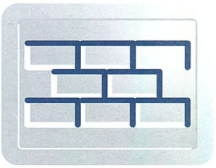
A könyv azon informatikai és biztonságtechnikai vezetők számára is hasznos lehet a döntéseik előkészítésében, akik információvédelmi szállítókat, megoldásokat választanak ki, s akik számára fontos, hogy a kiválasztott rendszer megfelelő funkciókkal rendelkezzen, korszerű, szabványos és költséghatékony legyen, s együttműködjön más rendszerekkel.

A 304 oldalon keresztül tárgyalt ismeretek megértését több mint 100 ábra segíti az igényes kivitelű kötetben.

Bővebb információ: www.sanserif.hu, info@sanserif.hu



Intelligens kulcstartók



Az elektronikus aláíráshoz szükséges kódokat tároló miniszámítógépek

Miért fontosak az elektronikus aláírás készítésénél az úgynevezett kulcsok? Mik ezek pontosan és miért fontos a magánkulcs megfelelő védelme? Hogyan lehet a magánkulcsokat megfelelő módon kezelni és tárolni ahhoz, hogy biztonsági integritásuk megmaradjon? A következőkben bemutatjuk azon végfelhasználói eszközöket, amelyek segítségével biztonságos körülmények között, megbízhatóan tárolhatóak azon ún. aláírás-létrehozó adatok, amelyeket az elektronikus aláírások készítésénél alkalmazunk.

Virtuális kommunikáció

Az elektronikus kommunikáció nagyszerű dolog. Gyors, olcsó, megszünteti a papírkötegekkel való bajlódást, a múlt gondjai közé utalja a postán vagy a hivatalokban történő végtelen sorbanállást, és az ezzel járó gondokat, kellemetlenségeket. Van azonban egy hátránya, ez pedig az, hogy nem kifejezetten biztonságos: a nyílt hálózatokon – mint például az Internet – folytatott kommunikációt számtalan veszély fenyegeti. Ezek közül talán a legnyomasztóbbak az üzenetekhez történő illetéktelen hozzáférés, módosítás mellett a személyazonosítás problémája.

A virtuális térben leselkedő veszélyek elhárításának jelenleg ismert legkorszerűbb, működő megoldása a nyílt kulcsú kódoláson alapuló elektronikus aláírás, illetve rejtjelzés alkalmazása. Anélkül, hogy itt a technológia mélyebb rétegeit kívánnánk feszegetni, tekintsük át röviden a működés főbb alapelveit, annak érdekében, hogy megérthessük az úgynevezett kulcstároló modulok, vagy intelligens eszközök szerepét és fontosságát.

A nyilvános kulcsú kódolás

A nyilvános kulcsú kódolás működése olyan lakatéhoz hasonlítható, amelynek két kulcslyuka van, egy-egy beleillő kulccsal. *(Az RSA-algoritmust lásd tavaly decemberi számunkban – a szerk.)* Ha a két kulcs közül az egyikkel zárjuk a lakatot, akkor az csak annak a párjával nyitható ki, azaz még a bezárást végző kulccsal sem. A kulcspár egyik tagját nyilvános, a másikat magánkulcsnak hívják. A nyilvános kulcsok mindenki számára hozzáférhetők, míg a privát kulcsokat csak tulajdonosaik érhetik el. E kulcsok elektronikus formában létező adatok, s könnyen előállíthatók az ismert és gyakran használt alkalmazásokkal *(böngészőkkel vagy levelezőprogramokkal)*.

Hitelesítéskor az aláíró, az üzenetnek egy speciális matematikai eljárás segítségével elkészített digitális lenyomatát kódolja saját magánkulcsával. A kódolt lenyomatot nevezzük digitális aláírásnak, amely egyértelműen jellemző az üzenetre és a kódolást végző kulcsra.

Titkosítás esetén az a kommunikáló fél, aki a másik fél számára kíván titkosított üzenetet küldeni, kódolja az üzenetét a címzett nyilvános kulcsával. Az üzenet kizárólag a címzett birtokában lévő magánkulccsal dekódolható.

A kulcspárok működéséből következik, hogy a magánkulcs egyértelműen azonosítja a publikus kulcsot, ezért már csak arról kell meggyőződnünk, hogy az adott publikus kulcs valóban a vélelmezett kommunikáló félhez tartozik-e. Ehhez szükséges a tanúsítvány, amely a hitelesítés-szolgáltatók által kibocsátott, elektronikus formában létező igazolás, amelyek az aláíró nyilvános kulcsának, illetve adatainak összetartozását igazolják. A tanúsítvány hitelességéről a kibocsátó tanúsítványon lévő digitális aláírásának ellenőrzésével lehet meggyőződni.

Kulcsok – velük vagy nélkülük?

Az előző – rendkívül leegyszerűsített – összefoglalóból látható módon, egy adott hitelesítési vagy bizalmas kommunikációt lehetővé tévő rendszer biztonságos működésének fontos kérdése az, hogy az adott kommunikáló fél miképpen tudja megvédeni privát kulcsát, amely mint láttuk nem több egy elektronikus adathalmaznál.

A magánkulcs illetéktelen kezekbe kerülése beláthatatlan következményekkel jár. Mindenek előtt a csak nekünk szóló titkosított üzenet privát kulcsunk birtokában hozzáférhetővé válik mások számára. Még rosszabb, ha aláírást végző magánkulcsunk kerül veszélybe. Ebben az esetben az a személy, aki megszerezte aláíró magánkulcsunkat, olyan elektronikus aláírásokat képes előállítani, mintha azokat mi magunk készítettük volna, ezen keresztül mi tennénk adott esetben jogi következménnyel járó nyilatkozatokat. Talán nem is szükséges mélyebb összefüggésekben bemutatni, milyen következményekkel jár például egy minősített tanú-

sítványhoz tartozó magánkulcs eltulajdonítása, illetéktelen kezekbe kerülése *(összefoglaló néven: biztonsági sérülése)*. Az elkövető ekkor pl. a kulcs jogos tulajdonosának nevében képes teljes bizonyító erejű magánokiratba, vagy ügyvéd illetőleg közjegyző által ellenjegyzett okiratba foglalt nyilatkozattal megegyező joghatású nyilatkozatot tenni.

A kulcsok védelme és tárolása miniszámítógépekben

A nyilvános kulcsú kódolási rendszerek két alapköve tehát a magánkulcs megfelelő védelme, illetőleg a magánkulcs illetve az azt használó entitás hitelt érdemlő összekapcsolása, amelyet a megbízható harmadik személyek *(hitelesítés-szolgáltató szervezetek)* által kibocsátott igazolások, úgynevezett tanúsítványok valósítanak meg.

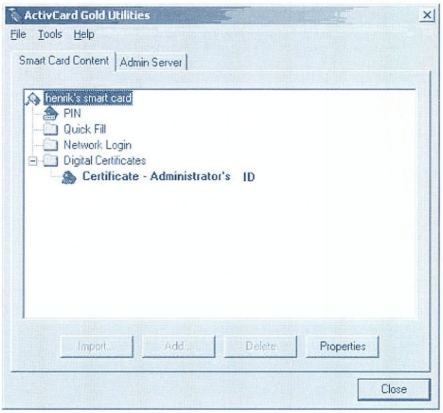
Intelligens kártya

A magánkulcsok védelmének ma ismert leghatékonyabb módja, ha a kulcsokat úgynevezett kulcstároló modulokban hozzuk létre, használjuk, illetve tároljuk. A kulcstároló modulok lényegében kis méretű számítógépek, amelyek nagyobb testvéreikhez hasonlóan rendelkeznek processzorral, memóriaterülettel, megfelelő háttértárral és a működéshez szükséges utasításkészlettel. Jelenleg két fő típusuk ismert, ezek az intelligens kártya, illetve az USB token. Ezek kizárólag külső megjelenésükben térnek el egymástól; a kártya bankkártya formátumú, az USB Token leginkább egy kulcstartóra emlékeztet.

Az intelligens kártyák és USB tokenek

Az intelligens kártyák méretükben a mindennapi használatban is előforduló mágneses sávval ellátott bankkártyák szabványait követik, felületükön a telefonkártyákon is látható érintkező látható. A kártyák soros vagy USB portra csatlakoztatható kártyaolvasóval kapcsolhatók a személyi számítógépekhez. Felületük különböző, a bankkártyákon is alkalmazott eljárások segítségével megszemélyesíthető *(a legjellemzőbb eljárások: hőtranszferes nyomtatás, dombornyomás, lézergaravírozás, vésés)*.

A kártyák felületén az adott felhasználó entitásra jellemző grafika, fénykép is megjeleníthető. Egyes kártyák alkalmasak ún. hibrid kártyaként funkcionálni, ekkor a nyilvános kulcsú kódolási műveleteket végző processzor mellett a kártyatestbe előzetesen beültetésre kerülhet például beléptető rendszerekkel együttműködni képes kontaktusnélküli chip, illetve az ehhez tartozó tekercs. Az első ilyen modellekben még nem, de az újabbakban a két chip képes lehet kommunikálni egymással, amely tulajdonság nagyban megkönnyíti a chipek kezelését, konfigurálását.



Az ActivCard egyik termékének grafikus felhasználói felülete

Természetesen a kártyák tartalmazhatnak egyéb olyan elemeket is, amelyek funkcionalitásuk számát növelik. Ilyenek lehetnek az egyéb adatok elhelyezését és leolvasását lehetővé tevő optikai eljárással írt tárterület, vagy aláírás elhelyezésére szolgáló csík. Ezek nincsenek kapcsolatban a PKI *(nyilvános kulcsú kódolási)* műveleteket végző chipmodullal.

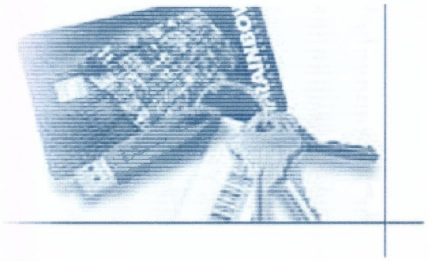
Maga a chipmodul működését tekintve lehet ún. fájl-, illetve alkalmazásalapú. A fájlalapú kártyák esetében a chipbe előre beállításra kerül az az utasítás- és objektumkészlet, amellyel a kártyának dolgoznia kell, ez később nem változtatható. Az alkalmazásalapú kártyák esetében a működtető rendszer is utólag kerül betöltésre, szabadon fejleszthető, legtöbbször Java alapú alkalmazás. Ez utóbbi esetben előny, hogy a kártyákat akár weben keresztül távolról lehet kezelni *(pl. új funkcionalitás beállítása, zárolt kártya feloldása, stb.)*, ugyanakkor ezek az alkalmazások nagyobb tárterületet és processzorteljesítményt követelnek meg.

Talán a legismertebb intelligens kártyákat a Schlumberger technológián alapuló termékeivel az ActivCard cég hozza forgalomba. Ezek közül a termékek közül mind fájl- *(Cryptoflex 16K)*, mind alkalmazásalapú *(Cyberflex Access 32K)* kártyák kikerülnek. Alkalmazási területük jellemzően a PKI, de – különösen a Java kártyák esetében – jellemzők a távoli menedzselés lehetősége, valamint különböző elektronikus funkciók – pl. elektronikus pénztárca – feltöltési lehetősége is. Az említett Java kártyát alkalmazzák az USA védelmi minisztériumában.

A tengerentúli gyártók közül érdemes megemlíteni a Rosetta termékeit. Ezeket a kártyákat használja többek között Florida Állam ügyészsége. A kártyák piacán egyre jelentősebbek az Oberthur termékek. Alkalmazásalapú kártyái *(pl.: Cosmopol IC)* szabványos Java nyelven alkalmazá-

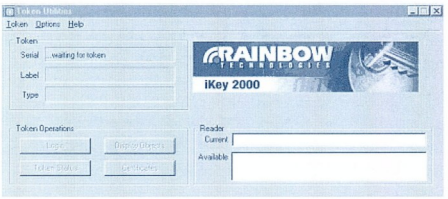
sokra, pl.: EMV kompatibilis pénztárca, biometria intelligencia, egyszeri jelszókezelés készíthetők fel.
Az USB Tokenek USB porton keresztül csatlakoztathatók a személyi számítógépekhez. Külső megszemélyesítésük, hibrid eszközként történő alkalmazásuk nem szokásos.
Az USB tokenek közül megemlítésre érdemesek elsősorban a Rainbow cég termékei. Különösen jó minőségűek és megbízhatóak a 2000 és 2032 iKey USB Tokenek. Ezek az eszközök kifejezetten PKI műveletekhez szükséges RSA kulcspárok generálására, biztonságos tárolására és védelmére alkalmasak. Egyéb funkciókat a biztonság további növelése érdekében csak egyes termékeknél találunk, ezek pl. a Windows 2000 operációs rendszerhez történő login funkció biztosítása.
Mindkét eszköztípus legfontosabb közös tulajdonságai közé tartoznak a következők:

- A magánkulcs az eszközben jön létre, biztonságos körülmények között. Az egyes eszközöket nemzetközi audit cégek vizsgálják felül biztonsági szempontból *(FIPS, CEN stb. minősítések)*. Az eszközt a magánkulcs még használat *(digitális aláírás készítése, titkosítás dekódolása)* közben sem hagyja el, illetve a magánkulcsot az eszközből kimásolni semmilyen módon nem lehet. A kulcs használata PIN számmal, jelszóval védett, melynek adott számú hibás megadása esetén a kriptográfiai eszköz blokkolja magát. Az eszközök többsége rendelkezik a blokkolást feloldó, külön *(néha többszintű)* kóddal.
- Az aláírási sebesség *(magánkulcs használat időtartama)* az eszközök többségénél néhány milliszekundum. Az eszközök a magánkulcs tárolásán túlmenően alkalmasak lehetnek egyéb adatok, így a tanúsítvány *(nyilvános kulcs)*, egyéb jelszavak, azonosítók tárolására is, jellemzően így is használják őket. A szabványos és elterjedt eszközök mindegyike támogatja a legfontosabb kriptográfiai szabványok használatát *(pl: RSA, PKCS sorozat)*.



USB Token (forrás: www.rainbow.com)

Az eszközök közül néhányat egyéb biztonsági kiegészítő funkciókkal szállítják, ezek hálózati bejelentkezés *(Windows, Novell, stb.)*, titkosított fájl állományok kezelése, vagy jelszavak rögzítése, védelme és "visszajátszása".
Az eszközöket a gyártók rendszerint csomagban értékesítik, melynek kártyák esetén része a kártyaolvasó, USB Token esetén egy USB hosszabbító kábel, illetve mindkét esetben az eszköz működtetéséhez szükséges szoftver is. Az eszközök természetesen külön is megrendelhetőek. A telepítést és használatot felhasználóbarát grafikus felületek segítik.

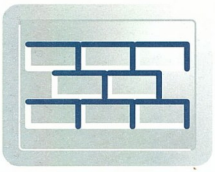


A Rainbow Inc. egyik termékének grafikus felhasználói felülete

Előnyök, hátrányok

A bemutatott kulcstároló eszközök vitathatatlan előnye, hogy hordozhatóvá, „megfoghatóvá" teszik az elektronikus aláírást létrehozó adatokat, ugyanakkor lehetővé teszik ezek nagybiztonságú kezelését. Ilyen módon kulcsaink elválaszthatók a számítógéptől, attól függetlenül biztonságba helyezhetők, elvesztésük esetén a PIN szám beállításai, illetve az adatok kódolt tárolása lehetetlenné teszik a magánkulcshoz való illetéktelen hozzáférést.
Ugyanakkor világosan látni kell azt, hogy egy adott projekt esetében, ahol ilyen eszközöket fognak alkalmazni, meglehetősen kiterjedt és részletes elemzésre van szükség ahhoz, hogy megállapíthassuk, pontosan milyen eszközre is van szükségünk.
Az alapvető szempontok természetesen a megcélzott alkalmazási feladatok, illetve az adott eszköz által kínált biztonsági funkciók elemzése. Itt figyelemmel kell lenni arra a tényre, hogy az eszközgyártók és forgalmazók által az adott eszköznek tulajdonított tulajdonságok nem mindig, minden esetben működnek pontosan a leírt módon.
Figyelemmel kell lenni az alkalmazási környezet egyes elemeire, lényeges sajátosságaira, amelyek befolyásolhatják az eszköz működését. Az eszköz biztonságos voltának megállapításához megfelelő iránymutatást adhat az a tény, hogy az adott eszköz milyen biztonsági minősítést szerzett meg *(ilyenek lehetnek pl. a FIPS vagy az Common Criteria minősítések)*.
Minden esetben a végleges döntés meghozatala előtt lehetőleg olyan rendszerkörülmények között célszerű tesztet végezni az adott termékkel, amelyek a leginkább megközelítik a célzott alkalmazási környezetet.
A fenti szempontokat is figyelembe véve érdemes az eszközöket úgy beszerezni, hogy közben a kapcsolódó iparágban – tanúsítványkibocsátás és PKI integráció – tevékenykedő szakértőtől kérünk tanácsot. Ilyen szakértelmet ma Magyarországon elsősorban a hitelesítésszolgáltató szervezeteknél találhatunk, amelyek többsége egy vagy több ilyen eszköz támogatásával is foglalkozik.
Az egyik ilyen hitelesítés-szolgáltató a NetLock Kft., Magyarország első fokozott biztonságú hitelesítés-szolgáltatója. Immár 6 éve végzi elektronikus kommunikáció hitelesítésére és titkosítására alkalmas szabványos, X.509-es tanúsítványok kibocsátását, valamint ezen tanúsítványok publikus adatbázisának kezelését és karbantartását *(hitelesítés-szolgáltatás)*.

Boromisza Zsolt
boromisza_zs@netlock.net



Hitelesítés-szolgáltatói hierarchiák

Tanúsítványláncok, tanúsítvány ellenőrzés bonyolult esetben

Az elektronikus aláíráshoz szükséges tanúsítványok kibocsátását mint a világ több más országában, hazánkban is piaci alapokon működő hitelesítés-szolgáltatók *(HSZ)* végzik. Ezen megbízható harmadik személyek rendelik össze az aláírás ellenőrző adatát jelentő nyilvános kulcsot és az entitás adatait a tanúsítványban, amit végül a hitelesség és megváltoztathatatlanság érdekében maguk is aláírnak. De vajon hogyan kapcsolódnak egymáshoz maguk a szolgáltatók és az általuk hitelesített tanúsítványok?

Hitelesítés Szolgáltatók és tanúsítványok közötti viszonyok

Még 2001. évvégén megszületett az elektronikus aláírásról szóló törvény, amely egy 1993-as uniós irányelv szellemét követve technológiasemleges szabályozási elveket vall, és próbál követni több-kevesebb sikerrel. A jogalkotó is felismerte ugyanis, hogy jelenleg a publikus kulcs infrastruktúra *(PKI)* révén valósulhat meg a gyakorlatban a hiteles; titkosító tanúsítványok esetén pedig a bizalmas elektronikus kommunikáció, adatforgalom az egyes entitások között.

Több ország gyakorlatának megfelelően hazánkban is elsősorban hitelesítés-szolgáltatásra szakosodott piaci szereplők vállalhatják fel a megbízható harmadik személy szerepét az entitás-hitelesítési folyamatban. A hazai szabályozás fokozott biztonságú, illetve minősített hitelesítés-szolgáltatókat különböztet meg annak alapján, hogy az általuk kibocsátott tanúsítványokkal aláírt elektronikus dokumentumok az írásba foglalt, vagy a teljes bizonyító erejű magánokiratok joghatásával bírnak-e.

A hitelesítés-szolgáltatók állami felügyeletét a Hírközlési Főfelügyelet látja el, amely a fokozott biztonságú HSZ-kat regisztrálja, míg a minősített HSZ státuszra pályázóknál lefolytatja a jogszabály által előírt teljes körű minősítési eljárást. Ennek keretében a hatóság áttekinti a szolgáltató működését, folyamatait, eljárásrendjeit, technikai és technológiai rendszereit, szabályzatait, szervezeti illetve személyi feltételeit.

Magyarországon eddig jelenleg a NetLock Kft. folyamodott ilyen eljárás lefolytatásáért a HÍF-hez. A társaság szolgáltatói tanúsítványai 1999-től valamennyi Microsoft-termékben megtalálhatók.

Az egyes hitelesítés-szolgáltatók a fent bemutatott szabályozás szerint tehát nem állnak egymással hierarchikus viszonyban, így az általuk kibocsátott tanúsítványok sincsenek egymásnak alá- vagy fölérendelve. Sőt, a HSZ tanúsítványaikat is ők maguk hitelesítik. Egy hierarchikusnak tűnő viszony azonban mégis csak felfedezhető, hiszen a minősített tanúsítványokhoz erősebb jogerőt fűz a jogalkotó, mint a fokozott biztonságú tanúsítványokkal aláírt dokumentumokhoz. A minősített HSZ-k természetesen mind fokozott, mind minősített tanúsítványt kibocsáthatnak ám ez utóbbiakban ezen minőségüket fel kell tüntetni.

Anomáliák

Pillanatnyilag hazánkban minősített tanúsítvány nem szerezhető be, hiszen még nem léteznek minősített HSZ-ek. Ez abból a szempontból kellemetlen, hogy az Art. *(Adózás rendjéről szóló törvény)* kötelezővé tette a kiemelt adózók számára, hogy elektronikus adóbevallásukat minősített elektronikus aláírással ellátva nyújtsák be.

Ennek áthidalására vezette be az APEH saját hitelesítő-szervezetét, amely a kiemelt adózók számára csak az APEH-el való hiteles kommunikációra használható tanúsítványokat biztosít. *(Az első minősített HSZ megjelenését követő 180. napot követő hónap utolsó napjáig. Ekkor az APEH hitelesítő szervezete jogszabály erejénél fogva megszűnik és valamennyi általa kibocsátott, korlátozott felhasználhatóságú tanúsítvány visszavonásra kerül.)*

És a külföldi HSZ-ek?

Felmerül a kérdés, hogy mi a helyzet a külföldi székhelyű, nem ritkán több országban is működő hitelesítés-szolgáltatókkal.

Természetesen az entitások szabadon választhatnak, hogy melyik szolgáltatót veszik igénybe. A hazai szabályok szerint, aki fokozott biztonságú tanúsítványt kíván kibocsátani, annak regisztráltatni, míg aki minősített tanúsítványt szeretne szolgáltatni annak minősíttetnie kell magát, ahhoz hogy az általa kibocsátott tanúsítványokkal ne csak joghatás nélküli aláírásokat lehessen létrehozni.

A jogalkotó, hogy megteremtse a hitelesítés-szolgáltatások „határtalanságát", egyrészt nemzetközi szerződés alapján elismerhet külföldi HSZ-kat illetve lehetőséget ad a kereszthitelesítési eljárásra az 2001. évi XXXV. elektronikus aláírásról szóló törvény 5. §-a alapján.

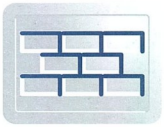
A probléma abból ered, hogy az érintett félnek *(azaz aláírás-ellenőrzőknek)* a HSZ-ek szolgáltatási szabályzata szerint egy jó hitelesítési láncot kell felépítenie. Ennek keretében meg kell

néznie, hogy az aláíró féltől kapott dokumentumon szereplő aláírást létrehozó tanúsítvány az aláírás pillanatában az azt kibocsátó HSZ hatályos szolgáltatási szabályzata szerint érvényes volt-e. Kisebb jelentőségű ügyletekben a vonatkozó szabályzatok áttekintése mellett elég visszakeresni az aláíráskor hatályban lévő CRL listát (letölthető file) és azon ellenőrizni, hogy szerepelt-e rajta a tanúsítvány. Nagy tranzakciók esetén csak akkor mondhatja el az érintett fél, hogy úgy járt el, ahogy az az adott helyzetben általában elvárható, ha emellett hasonló szempontból ellenőrizte a jobb HSZ-ek által már ma is üzemeltetett online állapottárt is.
Amennyiben a fenti ellenőrzés eredménye megfelelő, megvan a lánc első szeme. Ezután meg kell néznie, hogy az aláíró tanúsítványt kibocsátó HSZ-ban megbízik-e. Ez mindig szubjektív döntés az érintett fél részéről. Ahhoz, hogy ez a döntés megalapozott lehessen, ismerni kell azt a jogi, gazdasági környezetet, amelyben az adott HSZ működik, továbbá ismerni kell azon ellenőrzési és eljárási rendeket, amely alapján az aláírót, mint entitást a HSZ autentikálta.
Ez a gyakorlatban azért meglehetősen problémás lehet, ha például egy venezuelai HSZ által kibocsátott végfelhasználói tanúsítvánnyal aláírt dokumentum érvényességét firtatjuk.

Ekkor jön képbe a kereszthitelesítési eljárás. Itt arról van szó, hogy egy külföldi HSZ a saját aláíró tanúsítványát (amellyel ugye végfelhasználói tanúsítványokat ír alá) felülhitelesítteti egy hazai HSZ-val. Ez gyakorlatilag úgy történik, hogy számára a hazai HSZ kibocsát egy olyan láncolt hitelesítés-szolgáltatói tanúsítványt *(lásd a mellékelt ábrát)* melynek tartalma nagyrészt *(különösen a nyilvános kulcs)* megegyezik az eredeti külföldi HSZ-tanúsítvánnyal, de az issued by mezőben már a hazai HSZ szerepel.
A továbbiakban így ha a külföldi HSZ ezen tanúsítványával ír alá, a belföldi entitások olyan hitelesítési láncot tudnak felépíteni, melynek végén olyan HSZ szerepel, amely megbízhatóságáról sokkal megalapozottabb döntést tudnak hozni a fentebb leírtak miatt. Ez alapján az is elképzelhető, hogy maga az érintett fél is rendelkezik azon HSZ-nál tanúsítvánnyal, amely a hitelesítési lánc végén szerepel.

Természetesen ezen eljárás megoldást jelent akkor is, ha még az eredeti külföldi HSZ-tanúsítvánnyal aláírt végfelhasználói tanúsítványt akarunk ellenőrizni, hiszen ekkor több párhuzamos hitelesítési lánc felépítése történhet meg. Értelemszerűen elég, ha csak az egyik lánc lesz sikeres. Elképzelhető azonban olyan eset is, hogy több jó *(tehát szubjektív döntésünk alapján megbízható)* láncot tudunk felépíteni. Ekkor még biztosabbak lehetünk a hitelesítési lánc megfelelőségében.

De mi van abban az esetben, ha valami balul üt ki? Történetesen a hiba a HSZ részén történt a hitelesítési folyamatban, vagy a tanúsítvány kibocsátás kapcsán. És mi van, ha ebből a feleket kár is éri? Ez egy érdekes jogi problematikát vet fel, ami miatt feltehetően nem lesz gyakori hazánkban az olyan szervezet, amely kereszthitelesítést fog vállalni külföldi HSZ-k részére.
Ha jobban belegondolunk, a felülhitelesített külföldi HSZ-tanúsítvány révén olyan jó, bár csak utóbb megnyíló hitelesítési lánc kialakítása válik lehetővé az érintett fél számára, amely végén egy hazai *(tehát elérhető, könnyen perelhető stb.)* HSZ áll. *(az Eat. egyébként az 5. §-ban kifejezetten felelősségvállalásról szól, amelyet még a felügyeleti funkciót ellátó HÍF-nek is be kell jelenteni)* A fentebb leírtak alapján pedig, ha a belföldi entitás ezen láncot tekinti megbízhatónak, és mégis kár éri a HSZ hibájából, már feltehetően a hazai HSZ-t fogja perelni. A kereszthitelesítés által tehát a hazai HSZ maga is felelőssé válik egy olyan HSZ-hibáért, amelyet a külföldi HSZ követett el eljárásában, és ami miatt a feleket kár érte. Persze több párhuzamos hitelesítési lánc alapján a károsult választhat, hogy melyik HSZ-t perli. Választását a HSZ elérhetősége, földrajzi elhelyezkedése, mérete, pénzügyi helyzete, szabályzatainak kidolgozottsága, jogi környezete alapvetően befolyásolni fogja, hiszen ezen tényezők nagyban ki fognak hatni az esetleges per kimenetelére, folyamatára.



Láncolt hitelesítés-szolgáltatók *(LHSZ)*

A vonatkozó hatályos jogszabályok egyike sem foglalkozik még említés szintjén sem a láncolt hitelesítés-szolgáltatókkal. Ezen konstrukció lényege, hogy a HSZ valamely tanúsítványkiadója kiad egy olyan tanúsítványt, amellyel további, immár végfelhasználói tanúsítványok írhatók alá. Ezen új tanúsítványkiadóra ugyanazon szabályok érvényesek, mint az őt kibocsátó hitelesítő alegységre. Különbséget jelent azonban, hogy üzemeltetése egy másik szervezet révén történik.
Ezen másik szervezet lehet egy nagyvállalat, amely valamennyi dolgozója számára szeretne tanúsítványt biztosítani és így üzleti szempontból számára racionálisabb döntés LHSZ-t működtetni.
Az LHSZ csupán egy újabb tanúsítványkiadó alegység a HSZ-n belül, amely valamelyik tanúsítványkiadó alá tagozódik be, és amelyet történetesen egy másik szervezet üzemeltet, de amelyet továbbra is a HSZ felügyel és felel működéséért. Kis túlzással azt lehet mondani, hogy a dolog hasonlít a franchise rendszerben működő szolgáltatókhoz, üzletekhez, csak itt még szorosabb a kötődés a felek között. Az x509-es szabvány szerint ugyanis ha a HSZ úgy találja, hogy az LHSZ-t üzemeltető szervezet nem tartja be eljárásrendi előírásait, visszavonhatja az LHSZ tanúsítványát, és ezáltal mindazon tanúsítványokat érvénytelenné teszi, amelyet az LHSZ adott ki, feltéve ha azokból nem állapítható meg az adott tanúsítvány kibocsátási dátuma.
Az LHSZ által kibocsátott tanúsítványok ugyanazon joghatásokkal bírnak, mint az LHSZ-t kibocsátó tanúsítványkiadó által kiadott egyéb végfelhasználói tanúsítványok, feltéve, hogy a HSZ *(szolgáltatási utasítás)* vagy az LHSZ-t üzemeltető szervezet nem ír elő további korlátozásokat a tanúsítvány használatával kapcsolatban.

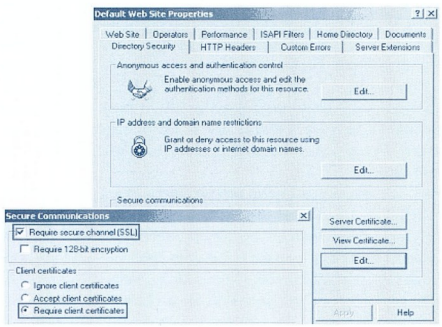
dr. Nagy Zsolt
NetLock Kft.
nagy_zs@netlock.net

Tartományi felhasználók webes azonosítása PKI segítségével

A Windows 2000 webkiszolgálója, az IIS 5.0 képes a hozzá érkező felhasználók tanúsítványalapú azonosítására is. A kiszolgáló elfogadhatja, illetve kérheti a felhasználóktól a tanúsítvány bemutatását, ami kiegészítheti, vagy kiválthatja a „hagyományos" (jelszavas) felhasználóazonosítási módokat.

HTTPS: Titkosított kommunikáció

A tanúsítványalapú ügyfélazonosítás kiszolgálóoldali feltétele az, hogy az IIS is rendelkezzen már saját tanúsítvánnyal, azaz legyen felkészítve az SSL csatorna *(HTTPS)* kezelésére. A témáról a Tech.Net magazin 2001. decemberi számában közöltünk cikket. Miután a kiszolgáló tanúsítványát telepítettük, máris engedélyezhetjük az ügyféltanúsítványok elfogadását.



A felső kijelölés az SSL csatorna, az alsó a tanúsítványkezelés bekapcsolása

Az SSL beállításokat a virtuális webkiszolgáló, bármely virtuális mappa vagy akár egy-egy fájl tulajdonságai között is megtaláljuk. A „Require secure channel *(SSL)*" beállítás hatására az érintett webkiszolgáló *(vagy mappa illetve fájl)* már csak SSL kapcsolaton, azaz https:// hivatkozással lesz elérhető. A „Client certificates" beállítás három értékének magyarázata:

- Ignore client certificates: a kiszolgáló az ügyfél által küldött tanúsítványt nem veszi figyelembe
- Accept client certificates: a kiszolgáló az ügyfél által küldött tanúsítványt elfogadja és feldolgozza, de kiszolgálja a tanúsítvány nélkül érkező kéréseket is
- Require client certificates: a kiszolgáló csak a megfelelő tanúsítvánnyal rendelkező ügyfélkéréseket szolgálja ki, minden más kérést elutasít.

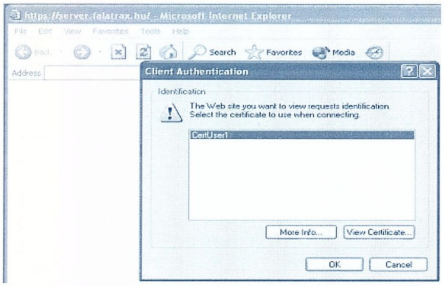
Az ügyféltanúsítványok elfogadása és feldolgozása önmagában nem jelenti azt, hogy a kiszolgáló a tanúsítvány alapján a felhasználót azonosítaná. Az IIS ilyenkor csak annyit tesz, hogy a tanúsítványt továbbadja a webalkalmazás számára, „aki" azt tesz vele a továbbiakban, amit jónak lát. ASP oldalakban például a Request.ClientCertificate kollekció segítségével férhetünk hozzá a felhasználó tanúsítványához *(a cikk során többször látható weboldal is ezt használja fel)*.

Ha egy weboldal eléréséhez szükség van a felhasználó tanúsítványára, de a böngésző azt nem küldi el *(például mert nincs a felhasználónak olyan tanúsítványa, ami erre a célra megfelelő lenne)*, a kiszolgáló elutasítja a kérést *(HTTP 403.7 – Forbidden: Client Certificate Required hibaüzenettel)*.

„Az oldal eléréséhez ügyféltanúsítvány szükséges"

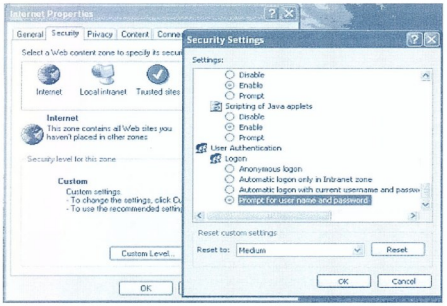
Ügyféltanúsítvány telepítése és kiválasztása

Az azonosításhoz a felhasználó nevére kiállított, legalább „Client Authentication" célra felhasználható tanúsítványra lesz szükség. Ha ilyennel nem rendelkezünk, a céges *(Windows 2000 Certificate Server)* vagy független tanúsítványkiadó szervezettől beszerezhetjük azt. Miután a tanúsítványt telepítettük, a védett oldalhoz való csatlakozáskor az Internet Explorer felajánlja, hogy válasszunk a rendelkezésünkre álló tanúsítványok közül.



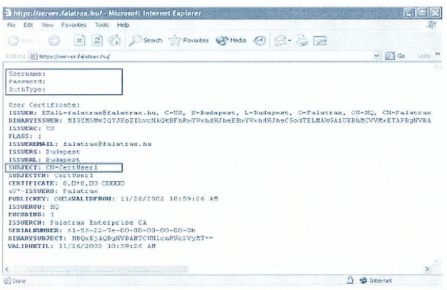
Miután telepítettük, a csatlakozáskor kiválaszthatjuk a használni kívánt tanúsítványt

Ha ez az ablak nem jelenne meg, az Internet Explorer valószínűleg automatikusan megpróbálkozik a bejelentkezéssel. Alapértelmezésben ezt akkor teszi, ha az elérni kívánt webhely a megbízott *(Trusted Sites)* vagy a helyi alhálózatba tartozó *(Local Intranet)* helyek között található *(ellenőrizzük a böngészőablak jobb alsó sarkában megjelenő feliraton)*. Az automatikus bejelentkezést az Internet Options panel Security oldalán a megfelelő zónát kiválasztva, majd a Custom Level... gombra kattintva megjelenő listában tilthatjuk le:



„Prompt for user name and password": felhasználónév és jelszó (valójában tanúsítvány IS) kérése minden esetben

Miután a böngésző elküldte a tanúsítványt a kiszolgálónak, azt a webalkalmazás feldolgozhatja. Ne feledjük el, még nincsen szó felhasználóazonosításról. Az oldalt a felhasználó – mint látjuk majd – bejelentkezés nélkül, névtelen *(anonymous)* ügyfélként éri el, bár ehhez szükség volt egy érvényes, a kiszolgáló által elfogadott tanúsítványra. Az ábrán látható oldal tetején jelölt téglalapban a bejelentkezéshez használt felhasználónév *(ha rendelkezésre áll, akkor jelszó)*, valamint a felhasználóazonosítási mód jelenik meg – láthatjuk hogy ebben az esetben ezek a mezők üresen maradtak. Ez azt jelenti, hogy a felhasználó anonymous ügyfélként csatlakozott.



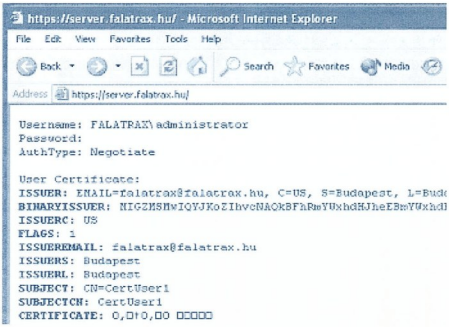
Az „első" sikeres „bejelentkezés" a felhasználó tanúsítványa segítségével

Mindezek mellett, a User Certificate: listában az elküldött tanúsítvány paramétereinek listája látható. Néhány fontosabb mező jelentése a következő:

- ISSUER: a tanúsítványt kiadó szervezet teljes neve
- SUBJECT: a tanúsítvány tulajdonosának neve
- VALIDFROM, VALIDTO: a tanúsítvány érvényességének kezdete és vége
- SERIALNUMBER: a tanúsítvány sorozatszáma

Bejelentkezés kikényszerítése

Ha a tanúsítványkezelés mellett a kiszolgálón bekapcsoljuk a felhasználói bejelentkezés kikényszerítését, a böngészőben a tanúsítvány kiválasztása előtt megjelenik a szokásos bejelentkeztető ablak IS. Nézzük csak, milyen érdekes eset állhat elő ebben az esetben:



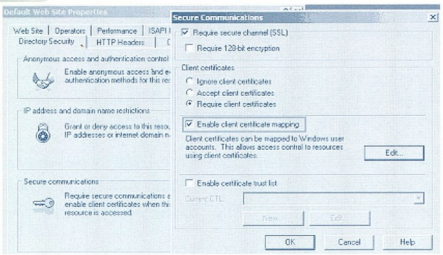
Kavarodás: a tanúsítvány a CertUser1-é, a bejelentkezés az Administratoré...

Az átadott tanúsítvány tulajdonosa CertUser1, miközben a bejelentkezéshez a tartományi rendszergazda felhasználónevét és jelszavát adtuk meg. *(A bejelentkezés típusa: Negotiate, ami az Integrated Windows felhasználóazonosítást jelzi; a jelszó ilyenkor nem látható.)* Nincs ebben valami furcsa? Tulajdonképpen nincs. Ha ismerjük az idegen felhasználó nevét és jelszavát, valamint rendelkezünk a tanúsítvány privát kulcsával, ez az eset teljesen természetes. Más kérdés, hogy ebben az esetben hogy állunk hozzá a felhasználó kilétéhez: vajon a tanúsítvány tulajdonosával vagy a jelszóval bejelentkeztetett felhasználóval állunk szemben? Az IIS mindenesetre ilyenkor NEM a tanúsítvány tulajdonosát veszi figyelembe.

Tanúsítvány-felhasználó összerendelés

Térjünk vissza az IIS tanúsítványkezelési beállításaihoz! A dialógusablakban találunk egy „Enable client certificate

mapping" opciót. Ha ezt bekapcsoljuk, az IIS a kapott tanúsítványokat az általunk meghatározott szabályok alapján hozzárendeli a létező felhasználókhoz.



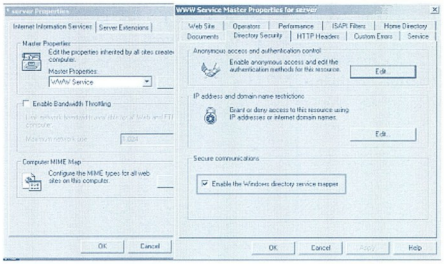
A tanúsítványok és felhasználók összerendelésének bekapcsolása

Az IIS háromféle módon képes a tanúsítványokat a felhasználókhoz rendelni, ebből kettőt az itt is látható „Edit..." gombra kattintva megjelenő Account Mappings ablakban állíthatunk be.

- 1-to-1 mapping: ebben az esetben minden tanúsítványhoz egy felhasználót rendelhetünk hozzá; a listába az Add... gomb segítségével importálhatjuk a kívánt tanúsítványt. Ilyenkor tehát szükség van az ügyfelünk tanúsítványára.
- Many-to-1 mapping: ebben az esetben viszont nem konkrét tanúsítványokra hivatkozunk, hanem szabályokat hozhatunk. A szabályok segítségével a beérkező tanúsítványokat kiadójuk *(Issuer)* vagy tulajdonosuk *(Subject)* alapján rendeljük az IIS felhasználóihoz. Mindkét mezőben használhatunk Joker karaktereket is, tehát a Many-to-1 mapping lehetővé teszi azt, hogy több tanúsítványt *(például amelyek egy adott kiadótól – mondjuk egy partnercég saját CA-jától – származnak, vagy mondjuk amelyek tulajdonosának leírásában szerepel a „komuves" szó)* ugyanahhoz az egy IIS felhasználóhoz rendeljük.

A fenti két lehetőség egymást nem zárja ki. A harmadik típus csak Windows tartományban működik, és máshol is kell bekapcsolni. Ez az Active Directory-alapú felhasználó-hozzárendelés, amikor a tanúsítványok feldolgozásához nem az IIS-en kézzel létrehozott szabályokat, hanem a tartományi adatbázist használjuk fel. A módszer neve Directory Services Mapping, és webszerver- *(figyelem! nem virtuális webhely-)* szinten kell és lehet engedélyezni, a webkiszolgáló úgynevezett Master tulajdonságai között. Ha a DS Mapping-et engedélyeztük, a 1-to-1 és Many-to-1 azonosítási szabályok az egész kiszolgálón érvénytelenek lesznek. Általában is elmondható, hogy az első kettővel szemben a címtáralapú azonosítás a Microsoft által ajánlott módszer.

A bekapcsoláshoz az IIS felügyeleti MMC modulban kattintsunk jobb gombbal a webkiszolgáló sorára, majd a tulajdonságok ablakból kattintsunk a Master Properties: WWW Service – Edit... gombra. Erre megjelenik a WWW Service Master Properties for <kiszolgálónév> tulajdonságablak, amelynek Directory Security oldalán megtaláljuk a hőn áhí-



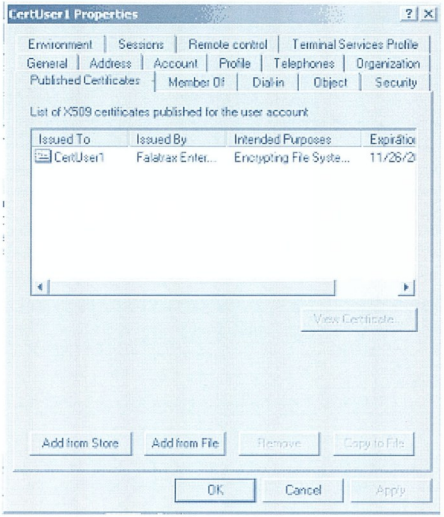
A címtáralapú tanúsítvány-felhasználó összerendelés szerverszinten kapcsolható be

tott opciót: „Enable the Windows directory service mapper". A beállítás módosítása után indítsuk újra a webszolgáltatásokat.

Fontos! A tanúsítványok kezeléséhez továbbra is szükség van az „Enable client certificate mapping" beállítására is, mindössze az összerendelés-szabályok lesznek érvénytelenek.

A címtár beállításai

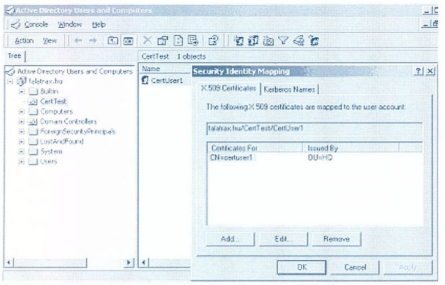
A címtáralapú tanúsítvány-hozzárendelés működésének érdekében az Active Directoryban a felhasználói objektumhoz hozzá kell rendelnünk a felhasználó tanúsítványát. Ennek két módja van, az első a felhasználó nevében közzétett tanúsítványok listája, amit a felhasználóobjektum tulajdonságai között a „Published Certificates" lista. Ide – ha a gépen telepítve van, - az „Add from Store", illetve – ha a tanúsítványt fájlba mentettük – akkor az „Add from File" gomb megnyomásával vehetünk fel újabb tanúsítványokat.



A felhasználó nevében közzétett tanúsítványok listája az Active Directoryban

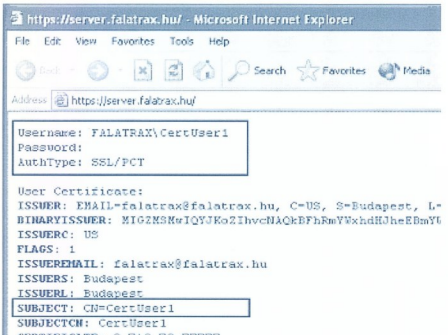
Ebbe a listába kerül bele automatikusan minden, a céges CA által a felhasználóknak kiadott tanúsítvány is. Fontos,

hogy az itt közzétett tanúsítványokat nemcsak az IIS, hanem bármely PKI-kompatíbilis alkalmazás is felhasználhatja, ha úgy gondolja, hogy éppen szüksége van az adott felhasználó nyilvános kulcsára. Ha ezt szeretnénk elkerülni, a tanúsítványt ne ide importáljuk be, hanem használjuk ki a másik lehetőséget: ez az egyszerű hozzárendelés.



A felhasználóhoz hozzárendelt tanúsítványok

Ehhez kattintsunk jobb gombbal a felhasználó nevére, majd válasszuk a Name Mappings... parancsot. A megjelenő dialógusablak X.509 Certificates oldalára beimportálhatjuk a kívánt tanúsítványt. Az IIS mindkét lista alapján képes azonosítani a felhasználót. Ellenőrizzük tehát, hogy valamelyik listában szerepel-e a felhasználó által küldött tanúsítvány, majd próbáljunk újra bejelentkezni:

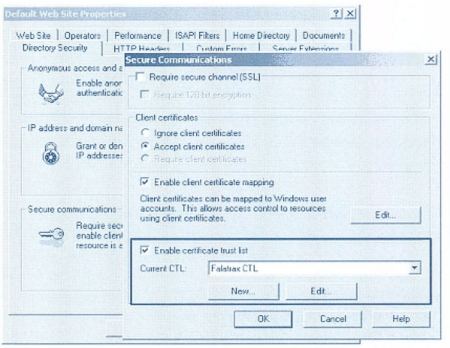


A végső bejelentkezés

Lássuk csak: az átadott tanúsítvány tulajdonosa *(Subject)*: CertUser1 – rendben. A felhasználó is CertUser 1 – ez is rendben; a bejelentkezés típusa *(SSL/PCT)* pedig jelzi, hogy tanúsítványalapú felhasználóazonosítással állunk szemben. Bingó!

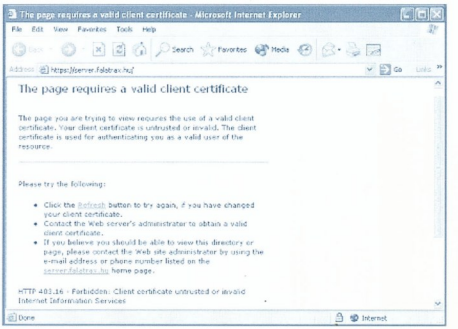
A Certificate Trust List

A webkiszolgáló alapértelmezésben minden tanúsítványt elfogad, amelyet valamely, a számítógép által megbízott tanúsítványkiszolgálója, vagy az általuk tanúsított „gyermek" tanúsítványkiszolgáló adott ki – legfeljebb nem tud hozzá felhasználót rendelni. Ha azt szeretnénk, hogy az IIS csak az általunk beállított tanúsítványkiszolgáló által kiadott tanúsítványokat kezelje, hozzuk létre és állítsuk be az általunk megbízott tanúsítványkiszolgálók listáját *(Certificate Trust List - CTL)*.

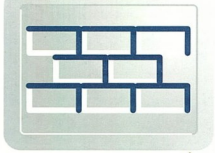


A webhelyünk által megbízott tanúsítványkiszolgálók listája

Ehhez menjünk az adott webhely tulajdonságlapjára, majd ott nyissuk meg a jól megszokott biztonsági dialógusablakunkat. Ha kijelöljük az „Enable certificate trust list" opciót, kiválaszthatunk egyet a már meglévő CTL-ek közül, vagy a New... gombra kattintva elindíthatjuk a CTL varázslót. A varázsló oldalain kiválaszthatjuk a megbízott CA-k tanúsítványait, majd elmenthetjük az újonnan létrehozott listát. Az esetünkben létrehozott, „Falatrax CTL" nevű lista például csak a saját, céges tanúsítványkiszolgálónkat tartalmazza, így az IIS a máshonnan származó tanúsítványokat nem fogadja el, hanem hibaüzenetet küld a böngészőnek *(HTTP 403.16: Client certificate untrusted or invalid)*:



Fülöp Miklós
mick@netacademia.net



PKCS, X.509, RFC, ISO

mi micsoda a PKI-szabványok világában?

Talán a nyílt kulcsú algoritmusok után a második legnagyobb logikai útvesztő a technológiával kapcsolatba hozható szabványdzsungel megértése. Borzasztó látni, és tudni, hogy egy PKCS 10-es tanúsítványkérelemre egy RCF XXXX-be csomagolt X.509 -es választ kapunk, melyben a publikus kulcs az AIN.1-nek megfelelően tárolódik.

Mi van itt? Miért nem képes egyetlenegy szabványtestület átvenni az uralmat, és rendet vágni ebben a dzsungelben? Szabványos lehet-e egyáltalán, ami felett hatan uralkodnak? Ki kivel van? És milyen formátumba érdemes exportálni a tanúsítványokat? PKCS 7? X.509 DER?

A fent említett szabványok szerencsére teljesen zökkenőmentesen együttműködnek egymással, az ember szinte nem is érti, hogyan lehetséges ez. Én megmondom. Úgy, hogy a legtöbb PKI-szabvány többféle néven szerepel, de attól még egy közös készlet része. Az egyes szabványtestületek építenek a másik által készített szabványokra, vagy egyszerűen átveszik a többiektől azokat, amelyek "megtetszettek" nekik *(ez utóbbira példa a PKCS 6, mely az ITU testületnél X.509 néven fut).*
Tovább kutatva az okok után kiderül, hogy a PKI-szabványok (szinte) mindegyike közös ősre, a PKCS-sorozatra vezethető vissza, melyet talán kényszerűségből *(nem volt más megoldás)*, talán a technológiavezető erejéből fakadóan Rivest, Shamir és Adleman vállalata, az RSA Laboratories dolgozott ki. A "fiúk", végiggondolva a nyílt kulcsú titkosítással megvalósítható funkciókat, s az ezek során előálló feladatokat, 15 javaslatot dolgoztak ki a kulcsok és dokumentumok kezelésére és tárolására. Ezeket ma összefoglaló néven PKCS-sorozatnak hívjuk *(Public Key Cryptography Standard)*, és az alábbi táblázatban olvasható, melyik milyen műveletet, algoritmust, dokumentumformátumot vagy egyéb feladatot ír le:

PKCS # 1	Az RSA titkosítás és digitális aláírás szabványa.
PKCS # 2	Már nem létezik, belekerült a PKCS # 1-be
PKCS # 3	A Diffie-Hellman kucscserélő algoritmus szabványa
PKCS # 4	Már nem létezik, belekerült a PKCS # 1-be
PKCS # 5	Jelszóalapú kulcsgenerálás (Password-base Encryption, PBE)
PKCS # 6	A tanúsítványok szabványa. Az X.509 átvette a szerepét
PKCS # 7	A titkosított üzenet szabványa
PKCS # 8	A magánkulcs tárolásának szabványa
PKCS # 9	A többi PKCS-szabvány attribútumainak gyűjteménye
PKCS # 10	A tanúsítványigénylés szabványos formátuma
PKCS # 11	Smart Card API
PKCS # 12	Bizalmas adatok, kulcsok, tanúsítványok továbbítási formátuma
PKCS # 13	Az Elliptic Curve titkosítási algoritmus szabványa
PKCS # 14	A kvázi-véletlenszámok (Pseudorandom Number Generation) előállításának szabványa
PKCS # 15	Kulcstárolás Smart Cardon

Ezek közül a hatosra, mint legfontosabbra, rávetette magát az International Telecommunications Union *(Világ Matávjai Egyesüljetek)*, és X.509-es sorszámmal saját lajstromába vette. S ha már lúd, legyen kövér: tőle átvette az Internet Engineering Task Force *(IETF)*, és *(többek között)* RFC 2510 számmal futtatja *(Internet X.509 Public Key Infrastructure Certificate Management Protocols)*. Ebből a buliból az International Standards Organization sem akart kimaradni, nála a PKCS-sorozat az alábbi sorszámot kapta: 1.2.840.113549.1
Az RSA Labs "jólelkűségére" jellemző *(biztosan kaptak néhány millió dollár fájdalomdíjat)*, hogy ezután lemondott saját korábbi változatának erőltetéséről, így a PKCS 6 befejezte evilági pályafutását. De nem így a testvérei.
Az ITU tehát szemet vetett a hatosra, ami kizárólag a tanúsítványok felépítésével foglalkozik, de nem érdekelte, hogy ezeket hogyan tároljuk, továbbítjuk stb. Az X.509 egy, a számítógép memóriájában remekül tárolható struktúra, tele bináris adatokkal *(publikus kulcs, ïngerprint, CA-aláírás stb., lásd Almási János cikkét e számunkban)*, amelyet ebben a formában tárolni sem kényelmes, nemhogy például http protokollon keresztül szállítani. Valahányszor a tanúsítvány "megmozdul", kiexportáljuk, vagy hálózaton keresztül továbbítjuk, át kell kódolnunk valamilyen hordozható formátumra.

Tanúsítványkérés

Mindaddig vágyálom csupán a PKI szolgáltatások használata, amíg nincs tanúsítványunk. Mi is a tanúsítvány? Nem más, mint RSA-kulcspárunk publikus részének egy harmadik fél által digitális aláírással ellátott példánya. A helyzet az, hogy bárki képes RSA-kulcspárokat generálni, ezzel azonban nem sokra megy, mert a fránya Windows következetesen tanúsítványokat, vagyis aláírt kulcspárokat kér.
Vajon miért nem jó neki egy sima kulcspár, amit akár számológéppel is generálhatunk? Mert az RSA-algoritmus *(csakúgy, mint a Diffie-Hellman)* érzékeny a man-in-the-middle támadásra, vagyis a publikus kulcs adatátvitel közbeni meghamisítására. Ez ellen úgy védekezhetünk, ha az átvitt kulcsot hamisíthatatlanná tesszük: ezt a funkciót (is) látja el a tanúsítványkiállító szervezet, amikor saját kucspárjával aláírja a mi publikus kulcsainkat. *(Mellesleg a tanúsítványban ezer másik adatot is kapunk tőle, de ez itt most mellékes.)*

Fontos! Akármilyen megtévesztő is egyes Certificate Authorityk felhasználói felülete, az RSA-kulcspár mindig a lokális eszközön, például a Smart Cardon, vagy a Windowsban generálódik. Ez biztosítja, hogy a privát kulcs mindenki más számára hozzáférhetetlen legyen.

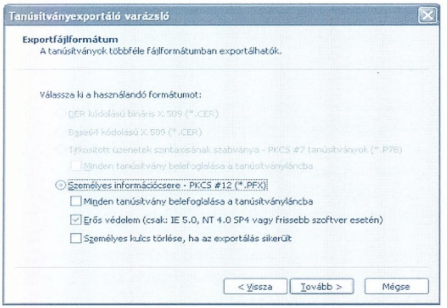
Az eszköz zokszó nélkül generál egy kulcspárt, aminek publikus tagját be kell küldenünk hitelesítésre. Ennek több módszere is ismeretes, a mi történetünk szempontjából az az eljárás érdekes, amikor a hitelesítési igény nem online módon, hanem fájlként érkezik meg a CA-hoz. Ilyenkor PKCS # 10-es formátumú adathalmazt, fájlt jutatunk el a CA-hoz.
A mai modern, Internetes világban mi szükség van a tanúsítványkérelem flopin történő benyújtására? Ennek több indoka is lehet. Egyfelől lehet, hogy nem bízunk meg a hálózati adatátvitelben, tartunk attól, hogy a publikus kulcs megváltozik, mire a céljához ér. Ez talán a gyengébb érv, tessék SSL-t használni. Másfelől vannak olyan CA-k, amelyek egyáltalán nem érhetők el hálózaton keresztül, ad abszurdum másképp sem, mert ki vannak kapcsolva és be vannak falazva (offline CA, például a Verisign-nál). Ennek igen praktikus oka van: egy kikapcsolt, betonozott szervert talán nem fog senki meghekkelni…
Az így őrzött CA-kkal „legegyszerűbben" gyalog *(vagy motoros futárral, á la Lurdi ház :-)* leszállított adathordozón keresztül kommunikálhatunk – azokban a ritka időszakokban, amikor kiveszik a szarkofágból és bekapcsolják.
Most tételezzük fel, hogy túlvagyunk a nehezén, és van X.509 tanúsítványunk, megérkezett a számítógépünkre. De a számítógép hálátlan jószág, egyik pillanatról a másikra elavul, és ki kell dobnunk. Rajta van viszont a kulcspárunk és tanúsítványunk. Hogyan mentsük meg? A helyes válasz a Smart Card lett volna, de ez a vonat már elment, a kulcsok a Windowsban csücsülnek.

Export/import

Vizsgáljuk meg, hogy a Windows milyen tanúsítvány kibeviteli módszereket ajánl fel:

A magánkulcs exportálása nem kötelező, sőt, néha lehetetlen is

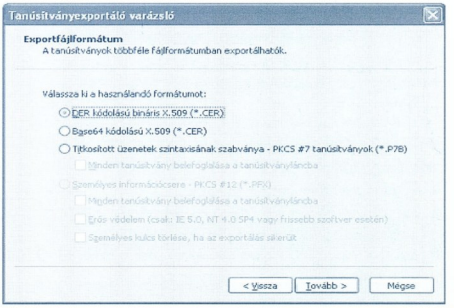
A tanúsítványok fájlba mentésének két alesete van: vagy vele együtt lementjük a magánkulcsunkat is, vagy nem. A magánkulcs akkor exportálható, ha ezt a tanúsítvány *(így kell kérni a tanúsítványt, később már ez nem módosítható!)*, illetve a hordozóeszköz lehetővé teszi. Egy okos Smart Card nem arról híres, hogy ki lehet belőle szedni a magánkulcsot! Így a most következő gondolatmenet a Windowsra bízott magánkulcsokra vonatkozik.



A magánkulcs exportálása miatt PKCS #12-t használunk

Próbáljuk kiválasztani az előző táblázatból, melyik PKCS-szabvány lenne alkalmas a privát kulcs tárolására is. A nyolcas és a 12-es jöhet szóba, ezek közül az utóbbit ajánlja fel a Windows, mivel nem egy önmagában álló kulcsot, hanem annak párját és tanúsítványát is exportáljuk:
Ha már PKCS #12, amely egy összetett formátum, megfigyelhető, hogy a fájlba belegyömöszölhetjük a teljes tanúsítványláncot egészen a gyökérig. Roppant hasznos szolgáltatás, mert így minden tanúsítvány benne lesz a fájlban a hitelesség ellenőrzéséhez.
Az erős védelem jelentése: a fájl jelszóval védett lesz. Az exportálás után dönthetünk úgy, hogy a magánkulcsot kivesszük a Windowsból *(személyes kulcs törlése)*.

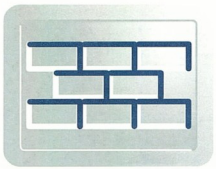
Ha a privát kulcsra nincs szükségünk, a helyzet egyszerűsödik, mert egy nyilvános X.509 tanúsítványhoz kapcsolódóan nincs titkosítandó adat. Van azonban más gond, mégpedig a tanúsítvány ellenőrzésének megoldhatósága. Hosszú tanúsítványláncok esetén, amikor nem közvetlenül egy megbízható gyökértől *(Trusted Root)* kapunk tanúsítványt, a hitelesség ellenőrzéséhez a lánc összes elemének publikus kulcsára szükségünk van. Egyetlen tanúsítvány exportálásával és továbbításával tehát nem biztos, hogy célhoz érünk. Szerencsére a PKCS #7 formátum némi kreatív felhasználásával *(titkosított üzenetek)* teljes tanúsítványláncok közös fájlba írása is megoldható.
Sima, privát kulcs nélküli exportáláskor a következő fájlformátumok között választhatunk:



Tanúsítványexportálási lehetőségek

A DER kódolás egy bináris tanúsítványforma. A Base64 kódolás azt a problémát hivatott áthidalni, hogy a memóriában masszívan bináris tanúsítványból néha pusztán ASCII-karaktereket tartalmazó fájlt kell írnunk. Ilyen eset például, ha a tanúsítványt http protokollon kell átvinni.
E rövid áttekintés után az egyes szabványok részletezésére, belső felépítésére további cikkekben térünk ki.

Fóti Marcell
MZ/X



RRAS és PKI élesben

A technet magazin korábbi számaiban is több cikk jelent meg az RRAS eszközről, az IPSec, VPN és PKI technológiákról. Mindegyik cikk kiváló összefoglalója volt az adott témának. Most mindezen eszközök együttes alkalmazását szeretném megmutatni egy projekt megvalósításának folyamatán keresztül.

A projekt célja egy nagyvállalat addigi vegyes RAS megoldásának – központi Cisco RAS eszköz ISDN PRI vonallal és vidéki telephelyeken NT 4.0 Domain Controllereken található Modemek – lecserélése egy központi, biztonságosabb megoldásra.

A cikk első részében a követelményeket, a koncepcionális döntéseket, a Pilot teszt „eredményét" ismeretetem, majd belevágunk a technikai részbe, ahol jelen számban a PKI infrastruktúrát, az IP címkiosztást és a RADIUS szervert tárgyaljuk, majd a következő számban folytatjuk a VPN és RAS szerver védelmével, packet filterekkel és a CMAK konfigurálásával.

Kritériumok

A megoldásban az ügyfél elvárta többek között az alábbi kritériumok teljesítését *(ezek azok a feltételek, amelyek nagymértékben befolyásolták a megvalósítandó rendszer kialakítását):*

- Központi ISDN és analóg hívások fogadására is alkalmas behívó szerver
- Erős adattitkosítás felhasználása IPSec-kel: IPSec használata, és 3DES adattitkosítás.
- VPN használata
- Kettős authentikáció: usernév, jelszó és Token vagy PKI alapú authentikáció egyidejű használata.
- A RAS szerver tűzfal mögötti elhelyezése DMZ zónában, a felhasználók adatforgalmát a tűzfal által ellenőrizve.
- Internetfelhasználás engedélyezése/tiltása felhasználónként

A megvalósítást leginkább érintő környezeti feltételek az alábbiak voltak:

- Active Directory címtár megléte
- Klienseken Windows 2000 és Windows XP operációs rendszerek
- Adott Cisco PIX tűzfal, amelyet nem használhattunk VPN szerverként

Koncepcionális döntések

A fenti követelményeket értékelve és elsősorban Microsoft eszközökben gondolkodva az alábbi koncepcionális döntések születtek:

Smart Card techológiát használunk a másodlagos authentikációhoz, konkrétan ActiveCard Gold kártyát és kártyaolvasót. A döntés háttere: a Smart Card technológiát a Windows 2000 és Windows XP is operációs rendszer szinten támogatja, valamint az a nem elhanyagolható tény, hogy leányvállalatunk az ActivCard cég partnere, és jelentős tapasztalatokkal rendelkezünk ezzel a termékkel.

Microsoft Certificate Server felhasználása az IPSec és VPN és Smart Card felhasználáshoz szükséges PKI infrastruktúrához. Ehhez a döntéshez nem nagyon kellett szempontokat keresni: a terméket az ügyfélnek nem kell megvárásolni, hiszen az része a Windows 2000 Servernek, a terméket ismerjük, megfelelő dokumentáció, támogatás és referencia áll mögötte.

A Microsoft IAS *(másnéven RADIUS)* szerverét használjuk AAA szerverként. Ez biztosítja az Active Directory integrált bejelentkezést, függetlenül a RAS és VPN szerver tényleges megvalósításától *(akár hardver behívószerver- NAS – is felhasználható)*, és olyan rugalmas RAS policy lehetőségeket nyújt amelyeket ki is használunk, például az Internetelérés korlátozásánál.

RAS szerverként Windows 2000 standalone szervert használunk (domain tagság nélkül), a fizikai interfészt egy Eicon Diva ISDN PRI kártya biztosítja, amely a 30 ISDN kapcsolatokon kívül mind a 30 csatornán analóg kapcsolatot is támogat.

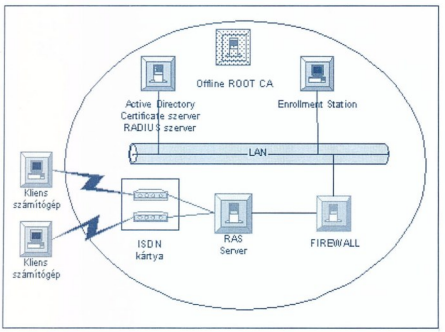
A VPN szervert a tűzfal DMZ zónájában kell elhelyeznünk: Mivel nem használhattuk fel a tűzfalat VPN szerverként, a VPN szervert is a DMZ zónában kell elhelyezni, hogy a tényleges adatforgalom a tűzfal által kontrollálható legyen – hiszen a titkosított adatforgalommal a tűzfal nemigen tud mit kezdeni. Úgy döntöttünk, hogy a tűzfal DMZ zónájában elhelyezett RAS szerver fog VPN szerverként is funkcionálni.

Az Internetfelhasználás korlátozása a RAS felhasználók részére a korábbi rendszerben IP címenként történt, az Internet elérésre jogosult RAS felhasználók más tartományból kaptak IP címet, és a proxy szerveren történt meg az IP címtartomány szerint az engedélyezés vagy tiltás.
A felhasználónkénti fix IP cím kiosztás helyett egy elegánsabb megoldást választottunk: az Internetelérésre nem jogosult kliens a RAS szervertől Windows 2000 csoporthoz tartozás alapján olyan kapcsolati beállítást kap, amelyben IP filterrel tiltjuk a proxy szerver elérését.

Az authentikációs eljárás a követelményeknek megfelelően kétszintű. Első szinten a RAS kapcsolat kiépítésénél a hagyományos usernév/jelszó páros segítségével az MS-CHAPV2 authentikációs eljárással történik, a második szinten a VPN kapcsolat kiépítésénél pedig a Smart Card segítségével EAP-Smart Card authentikációt használjuk.

A CMAK (Connection Manager Administrator Kit) segítségével készített kapcsolatkezelőt használjuk a kliens oldalon, így a felhasználónak nem kell a különálló RAS és VPN kapcsolatokat külön – külön kezdeményezni *(habár ez megold-*

ható a kliens operációs rendszerek standard kapcsolatkezelőjével: a VPN kapcsolat kiépítés képes egy másik kapcsolatot felépíteni előtte - nade a kliens melyik kapcsolatot bontsa a munkája végén?) és felügyelni.
Mindezen döntések alapján az alábbi mutatja a tervezett rendszert.



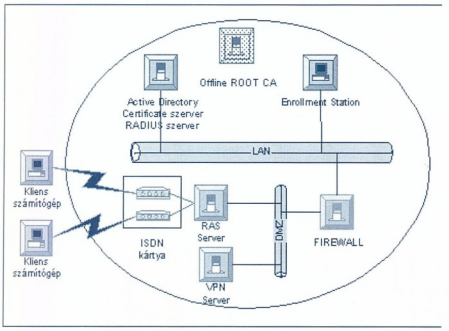
Az első terv

Pilot teszt

A rendszerterv elkészítése után természetesen egy Pilot rendszert építettünk fel, ahol teszteltük, hogy a tervezett rendszer funkciói megfelelnek-e a kívánt működésnek.
A pilot rendszer tesztelése alatt egy komoly hibát tapasztaltunk: sajnos – de mondhatjuk hogy a Pilot rendszer kialakítását standard munkafolyamatnak tekintő munkamódszerünknek hála – a RAS szerverre történő behívás után nem sikerült ugyanazon a RAS kapcsolaton L2TP alapú VPN kapcsolatot kialakítani a RAS szerver felé.

Minden más kombinációban megfelelően működött a rendszer:

- Nem RAS kapcsolatról keresztüli L2TP alapú VPN kapcsolatot is sikeresen felépítettünk a RAS szerver felé
- A RAS kapcsolaton keresztül más VPN szerver felé sikeresen felépítettünk L2TP kapcsolatot.
- RAS kapcsolaton át PPTP VPN kapcsolat kialakítása sikeres volt (ezt a tényt korábbi tapasztalataink alapján ismerve helyeztük a RAS és VPN szervert egy közös Windows 2000 szerverre).



A végleges rendszer

- IPSec nélküli L2TP kapcsolatot sem sikerült kiépíteni, a hiba tehát nem az IPSec

Mivel a rendelkezésünkre álló információforrásokban nem találtunk információt a hibáról, így a Microsoft PSS-hez fordultunk. A végeredmény: RAS kapcsolaton át ugyanarra a szerverre L2TP protokollal nem lehet VPN kapcsolatot felépíteni.

A tervezett rendszert módosítanunk kellett, a VPN szervert egy különálló Windows 2000 szerverre helyeztük. Mindazonáltal a tervezett rendszer összetettségét jellemzi, hogy további hibák merültek fel, amelyek megoldásához több hotfix-t kellett telepítenünk. Ezen hibákat az adott területek tárgyalásánál ismeretetem, mivel javításukhoz nem kellett a koncepción módosítani.
A megvalósított rendszerben vázlata ezek után a 2. ábra szerint alakult

Megvalósítás

A PKI infrastruktúra került kialakítása az első lépésben.

Helyi kétszintű PKI infrastruktúrát hoztunk létre, Offline ROOT CA szerverrel, és Enterprise *(azaz Active Directory integrált)* Subordinate CA-val

A PKI kialakításnál mind a ROOT CA, mind az Enterprise CA-nál előre specifikált AIA *(Authority Information Access: a tanúsítványt kibocsátó nyilvános kulcsa)* és CRL *(Certification Revocation List: visszavont tanúsítványok listája)* elhelyezési útvonalakat adtunk meg, a certificate server telepítése előtt egy capolicy.inf file testreszabásával, és a Systemroot *(általában a C:\WINNT)* könyvtárban történő elhelyezésével még a certificate szerver telepítése előtt.
A ROOT CA kulcs élettartamát 15 évben adjuk meg, és ezzel együtt a CRL publikáció intervallumát is 15 évre állítottuk, hogy ne járjon le a CRL érvényessége, mivel az Offline ROOT CA nem tudja rendszeresen frissíteni a CRL listát.

Az Enterprise CA telepítése előtt a címtárban publikálni kell a Root CA szerver nyilvános kulcsát, amihez a Windows 2000 Server Resource Kit dsstore.exe segédprogramját használjuk:

```
Dsstore.exe DC=intra,DC=gumikobalta,DC=hu -addroot
<root szerver nyilvános kulcs.crt> „Root CA neve"
```

```
Tipp: A DSStore.exe használatánál figyeljünk,
hogy a domainspecifikációnál a „DC"-t nagybetűvel
irjuk, egyébként a program nem találja a domain-t.
```

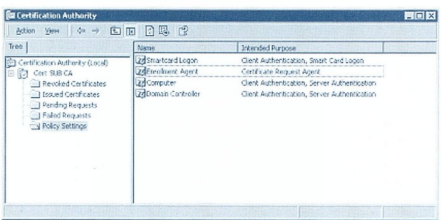
Az Active Directory címtárában a Domain NetBIOS neve és FQDN név baloldali része különböző volt *(pl. NetBIOS név: GUMIKOBALTA, FQDN név: intra.gumikobalta.hu)*, ezért a dsstore.exe nem működött megfelelően. Ennek a hibának a pilot teszt alatti detektálásához az kellett, hogy az éles címtárral teljesen megegyező konfigurációjú – beleértve ezt a domain névhasználatot is - tesztrendszert alakítsunk ki.
A hibához létezik javítás, a Q280122 Microsoft Knowledge Base cikk szerint, amely egy javított dsstore.exe-t tartalmaz.

Az Enterprise Root CA kulcsot 5 éves élettartammal szerettük volna megadni. Ehhez a ROOT CA szerveren a registryben kellett módosítani az alapértelmezett kulcs élettartamon, a

```
HKLM\SYSTEM\CurrentControlSet\Services\CertSrv\
Configuration\<Root CA neve>
```

kulcs alatt a ValidityPeriod és ValidityPeriodUnits értékek módosításával (Q254632). A registry változtatása után a CA szervízt újra kell indítani

A kulcsélettartam megadás Enterprise CA szerver esetén nem hatásos, kivéve további Subordinate szerver kulcsok generálása esetén. A kulcsélettartam a Windows 2000 Enterprise CA szerverben az egyes kulcsokfelhasználási célokhoz *(kulcstípusokhoz)* fixen rögzített. Ezen a gondon a .NET szerver CA szervere fog változtatni, ahol a különböző kulcstípusokhoz külön kulcsélettartamot adhatunk majd meg.



A használt Certificate templatek

Mivel az Enterprise Certificate szervert jelenleg csak a RAS rendszer használja fel, a kiadható kulcstípusokat korlátoztuk, mégpedig azon okból, hogy egy rendszergazda vagy egyéb feljogosított felhasználó nem tervezett módon ne igényelhessen kulcsot *(a Smart Card Enrollmentet egy erre a célra létrehozott felhasználói csoport részére engedélyeztük, az Active Directory Sites and Services MMC konzolban, a Service\Public Key Services\Certificate Templates alatt a szükséges templatek biztonsági beállításainál az „Enroll" jogosultság megadásával).*

Az engedélyezett kulcstípusokat a Certificate Server MMC konzoljában, a Policy Settings mappa alatt találhatjuk (ez csak Enterprise CA esetén van így). Lásd 3. ábra.

Az általunk használt kulcstípusok:

- Enrollment Agent: a Smart Card enrollmentet végző felhasználó és számítógép részére
- Domain Controller: a Domain Controllerek részére. A kulcskiosztás automatikus és szükséges a Smart Card authentikációhoz.
- Computer: az ügyfélszámítógépek részére az IPSec-hez használt kulcsok kiosztásához.
- Smart-Card Login: a Smart Card felhasználók részére.

A már úgyis meglévő Smart Cardokkal a felhasználók a „Smart-Card login" típusú kulccsal akár be is jelentkezhetnek a hálózatba. Ha ezt nem szeretnénk, és csak a RAS authentikációhoz akarjuk használni a Smart Cardot, a „Smart-Card User" típusú templatet is használhatjuk.

Jelenleg csak a RAS felhasználásra jogosult számítógépek részére osztunk ki Computer Certificate-t, amelyet egy adott Global Group részére korlátozott Domain szintű Group policy-val automatizálunk: A „Computer Configuration\Windows Settings\Security Settings\Public Key Policies\Automatic Certification Request" mappában felvettük a Computer certificate templatet.

IP címkiosztás

A vállalati címtartományban a tűzfal a RAS felhasználók részére meghatározott IP címtartományt engedi csak be. A DMZ zóna részére - beleértve a RAS és VPN szerver IP címét – egy külön IP címtartomány került meghatározásra, amely címtartomány felől a RADIUS kapcsolathoz szükséges portok, az időszinkronizáláshoz szükséges portok vannak engedélyezve, valamint a vállalati hálózat felől a terminál szervízhez szükséges MS RDP *(TCP 3389)* port.

A RAS kapcsolat (az analóg vagy ISDN behíváshoz és nem a VPN kapcsolathoz) részére kiosztandó IP címek egy teljesen független privát IP címtartományból kerültek kiosztásra, mivel a RAS kapcsolat sikeres felépítése után a klienseknek csak a VPN szervert szükséges és szabad elérnie. A VPN szerveren megadott statikus route biztosította, hogy a VPN szervertől a csomagok visszataláljanak a felhasználókhoz. Egy RAS kapcsolat felépítése után a felhasználók nem érték el még a tűzfalat sem, hiszen az nem ismerte azok címtartományát.

A VPN szerver az említett RAS felhasználók részére fenntartott IP címtartományból osztott IP címeket, amely IP címekről érkező megadott protokollokat, portokat a tűzfal átengedte a LAN hálózatba.

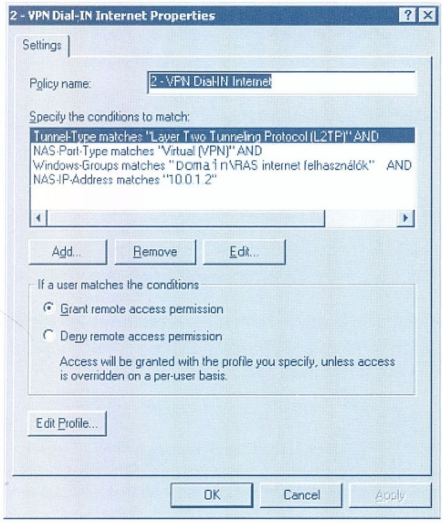
A RADIUS szerveren mindkét RRAS szervert (RAS és VPN szervert) felvettük RADIUS klienskként.
A RADIUS szerver által használt RAS policy-k két részből állnak, egy feltételrendszertől függ hogy illeszkedik-e az adott hívásra, és sikeres illeszkedés esetén egy profile specifikálja a felépítésre kerülő kapcsolat további tulajdonságait.

A RAS policy-kat a nevükben megszámoztuk, hogy egyértelmű legyen a sorrend, hiszen itt az első illeszkedő policy kerül felhasználásra.

Az első policy a RAS behíváshoz szükséges.
A feltételek a RAS felhasználók részére kialakított felhasználói csoport, ISDN porton történő behívás és a RAS szervertől érkező kérés.
A profile-ban 120 perc maximális kapcsolatot engedélyezünk. Csak MS-CHAPV2 authentikációt és Basic titkosítást állítunk be. Az IP címkiosztásnál csak a szerver által kiosztható IP címkiosztási opciót állítunk be.

A VPN kapcsolathoz két külön policy-t készítettünk, egyet az Internetelérésére jogosult felhasználóknak, és egyet az erre nem jogosult felhasználóknak. A két policy-ban külön felhasználói csoportot használunk.

A policy-k feltételei:
A RAS felhasználók részére kialakított felhasználói csoportok, VPN porton történő behívás, L2TP protokoll és a VPN szervertől érkező kérés. Lásd 4. ábra

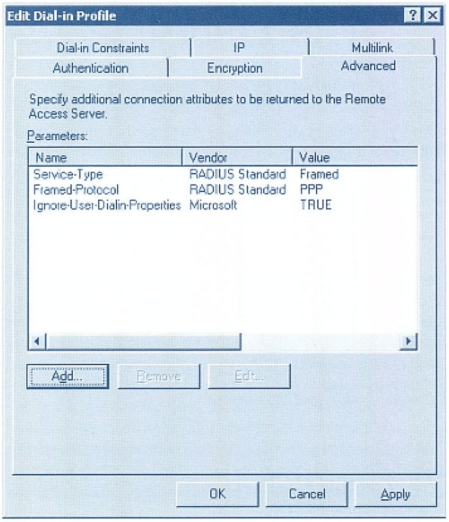


A VPN RAS policy kritériumai

A profile-ok tulajdonságai:
Titkosításnak a legerősebb titkosítást *(Strongest)* állítjuk be, ez biztosítja az IPSec 3DES algoritmus használatát.
Az authentikációnál csak az EAP authentikációt engedélyezzük, „EAP – Smart card or other certificate" kiválasztásával. Az EAP konfigurációnál kiválasztható az EAP-hoz használt számítógéptanúsítvány, ami jelen esetünkben a RADIUS szerver domain controlleren történő elhelyezéséből adódóan automatikusan kiosztásra került. Az EAP-Smart Card Authentikáció megfelelő működéséhez az szükséges, hogy a RADIUS szerver számítógéptanúsítványa és a bejelentkező felhasználók tanúsítványa *(amely a Smart Card-on tárolódik)* olyan CA szervertől származzon, amelyeknek ugyanaz a legfelső szintű tanúsítványa. Esetünkben nem csak a ROOT, de maga a tanúsítványt kiadó CA is azonos.

Mivel egy adott fix számra történő visszahívási lehetőséget is meg kellett valósítanunk, és a RADIUS szerver az Active Directory címtárból, a felhasználói account Dial-In tulajdonságainál megadott számot adta meg mind a RAS mind a VPN esetén, hagyományos PSTN telefonszámot megadva, az IP címet váró VPN hivásnál a szerver természetesen nem tudott visszahívni a hagyományos telefonszámra.

A hiba ismert és a Q289264 Microsoft Knowledge Base cikk szerint rendelkezésre áll a post SP2 hofix. A javítás telepítése után a RADIUS szerveren az adott RAS policy-nál megadhatjuk, hogy ne használja fel a címtárban az adott felhasználóhoz megadott egyedi beállításokat, köztük a visszahívási számot se. Ezt a beállítást a VPN kapcsolatot engedélyező mindkét RAS policynál természetesen beállítottuk Lásd. 5. Ábra.
Felvettük az újonnan megjelent Ignore-User-Dialin-Properties opciót, és „TRUE" értéket adtunk neki.



A VPN callback probléma megoldása

A hotfix-et telepítve azonban megállt a már működő Smart Card EAP authentikáció *(a Windows 2000 vezette be az EAP-t (Extensible Authentication Protocol), azaz bővíthető authentikációs protokoll-t, amihez mindjárt adott is egy „bővítést", a Smart Card authentikációt. Ezt követeltük meg a VPN kapcsolat kiépítése alatt a PAP, CHAP, MSCHAP és társai helyett)*

A RAS kliens az „Error 691 Access Denied because the username and/or password invalid on the domain" hibát jelezte vissza.
Úgy tűnik, belefutottunk egy hotfix által generált újabb hibába. A megoldás azonban kéznél volt. A Q289264 hotfixet már integrálták a Windows 2000 SP3-ba *(amit még nem kezdtünk el a projekt ideje alatt széles körben használni, de már rendelkezésre állt).*
Az SP3 –t kipróbálva a RADIUS szerveren *(és elegendő volt csak ott)* az EAP authentikáció és a visszahívás is megfelelően működött.

Szabó Lajos
(lalos)
MINOR Rt

Titkos kulcscsere nyílt adatokkal

A Diffie-Hellman algoritmus (PKCS #3)

Mint azt a tech.net olvasói "elviselik", időről időre megjelenik egy-egy kriptográfiai algoritmus leírása. A mostani számunk szintén ilyen. A világ legelterjedtebb kulcscserélő algoritmusát, a Diffie-Hellman eljárást ismertetem.

Nem kell megijedni, nem fogok középiskolai matematikai emlékeket meghaladó erősségű eszközökhöz folyamodni - mert nincs is rá semmi szükség. Ez az eljárás is olyan pofonegyszerűnek tűnik utólag, hogy az ember nem érti: miért nem őneki jutott először eszébe?

Titkos kulcscsere nyílt közegben?

Hol találkozunk a Windowsban a Diffie-Hellman elnevezéssel? Például a korábbi számokban többször emlegetett IPSec hálózati adatforgalom-titkosító technológia huszadik mélységben található párbeszédpaneleiben, a kulcsmenedzsment részben:

Az IPSec kulcsfrissítését Diffie-Helmannal végezzük

Mint az köztudott, mind az IPSec, mind az SSL valamilyen szimmetrikus kulcsú eljárással *(pl. 3DES)* titkosítja az adatokat. Ennek elsősorban az az oka, hogy a nyílt kulcsú algoritmusok *(pl. RSA)* sebessége meg sem közelíti az egykulcsúak *(DES, Rijndael, Bowlfish stb)* sebességét, így egyszerűen kiesnek a választékból. Ez rendben is volna, ha e kényszerű választás nem okozna egy súlyos problémát: hogyan juttassuk el a titkosítási kulcsot a kommunikációs partnerünknek? A hálózaton? Titkosítatlanul? Ennek nincs értelme. Máshogy, pl. telefonon? Ez ismerős emberek között lehetséges, de NEM ismerős NEM emberek (számítógépek, routerek stb.) között már aligha. A hálózaton, de titkosítva? Ez lenne a célszerű, de ennek akadálya, hogy nincs olyan kulcs, melyet mindkét fél ismerne. A feladat látszólag megoldhatatlan, hisz kellene egy kulcs, amivel titkosítani lehetne azt a kulcsot, amivel majd titkosítani lehetne az adatokat. Tehát kellene egy kulcs...

A kulcsleosztás problémája

Több olyan szimmetrikus kulcsú adatfolyamtitkosítást ismerünk, ahol a kezdeti kulcsproblémát sikerrel megoldották. Ilyen például a Kerberos hitelesítési protokoll, ahol a mindkét fél által előre ismert "kulcs" a felhasználó jelszava *(illetve az abból képzett hash)*. Milyen kár, hogy a leosztott kulcsokat nem használhatjuk fel adataink titkosítására! Nem adja ide a Kerberos! Vagy itt van az SSL. Ebben az esetben a kezdőkulcs leosztásához a kiszolgáló X.509 tanúsítványát, pontosabban az abban tárolt RSA publikus kulcsot használhatjuk fel, amelynek birtokában a munkaállomás az általa generált munkamenetkulcsot *(session key)* úgy tudja titkosítani, hogy azt csak és kizárólag a magánkulcs birtokosa, a megcélzott szerver tudja kibontani. *(Az X.509 tanúsítvány azért szükséges, hogy elejét vegyük a trükkös "man-in-the-middle" támadástípusnak, melynek lényege, hogy egy közbeékelődő harmadik fél az átmenő kulcsokat sajátjával helyettesíti, így az adatfolyam lelkes élvezőjévé válik.)* Az SSL-nél tehát megoldották a lehetetlent *(közös titkos adat nélkül jutnak a felek közös titkos kulcshoz)*, de az SSL hátránya, hogy – mivel az alkalmazásrétegben csücsül - csak bizonyos forgalomtípusok titkosítására alkalmas, mindenre sajnos nem képes Az IPSec éppen azért került az érdeklődés homlokterébe, mert „mindent visz", vagyis minden, az IP által szállított adatot betitkosít, legyen az akár egy egyszerű ICMP Echo. Így olyan alkalmazások is használt veszik, amelyek mit sem tudnak a titkosításról. Hogyan cseréljünk kulcsot két IPSec-gép között? Jó lenne egy, az SSL-hez hasonló nyílt kulcsú megoldás, amely nem igényel semmilyen előre leosztott közös adatot, s még csak X.509 tanúsítványt sem. Ez nem lesz más, mint a Diffie-Hellman algoritmus.

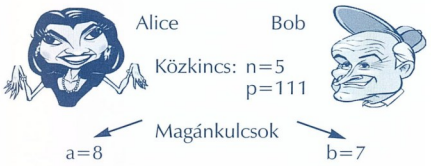
Diffie és Hellman

A két nevezett úriember 1976-ban tett javaslatot egy olyan eljárásra, mely lehetővé teszi, hogy két vadidegen fél közös publikus adatok birtokában csak kettejük által ismert szimmetrikus kulcshoz jusson. Ez elég hihetetlenül hangzik nemde? Ha minden adatot mindenki ismer, beleértve Hacker Henryt és Claudia Sniffert, a két kedvenc gonosztevőmet is, akkor hogy lehet általuk nem ismert kulcsban megegyezni? Nos, úgy, hogy nem minden adat publikus. A kulcsra vágyó felek mindegyikének van olyan saját adata, amit az algoritmusban felhasznál, de a partnerének csak közvetett módon küld el. Ha ettől sem lett tisztább a dolog, lássuk, mi lehet az a közvetett adat, ami csakis a partnernél hasznosítható! Egy hatványozás eredménye. A kulcsot generáló két fél a következő eljárást hajtja végre *(csúnya egyszerűsítéssel)*:

$$n^{a*b}.$$

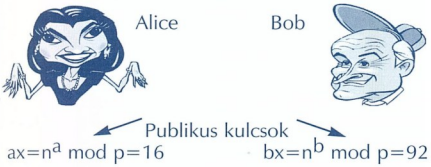
Mégpedig úgy, hogy az "n" egy kettejük által egyeztetett publikus szám, "a" az egyik fél által véletlenszerűen generált, és soha ki nem adott érték, míg "b" a másik fél "magánszáma".
Lássunk a kulcsgenerálási folyamatot, mely jól mutatja a felek által végrehajtott alaplépéseket.
Először is feltételezzük, hogy mindkét fél réges-régen megálla-

podott „n" és „p" értékében, úgyhogy ezt „közkincsnek" vesszük. Első lépésként mindkét fél, Alice és Bob is kiválaszt egy véletlen számot, ami az ő magánkulcsa lesz a folyamatban:



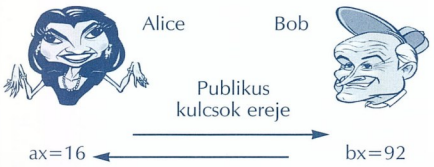
A Diffie-Hellman kulcsgenerálás első lépése

A két magánkulcs sohasem fog átmenni a hálózaton. Ehelyett a második lépésben minkét fél a saját magánkulcsával és „n", valamint „p" felhasználásával publikus kulcsot generál:



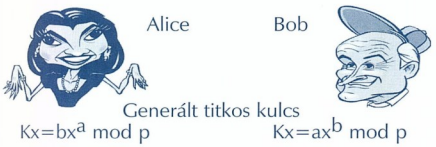
A Diffie-Hellman kulcsgenerálás második lépése

Ezután kicserélik egymás közt publikus kulcsaikat, amelyeket sajnos minden hacker elkap a hálózaton:



A Diffie-Hellman kulcsgenerálás harmadik lépése

És végül a partnertől kapott számot mindkét fél saját magánkulcsának hatványára emeli (mod p):



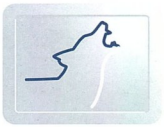
A Diffie-Hellman kulcsgenerálás negyedik lépése

Ez a mi egyszerű esetünkben mindkét félnél a 49 értéket fogja adni.

És kész. Egyszerű, nemde? Vajon miért nem én (vagy On:) találtam ki?

Lehallgatás

No és Hacker Henry? Mivel minden számítás eredményét fejbevágjuk maradékos osztással (mod p-vel, ahol p, n-hez hasonlóan szintén publikus szám), és az így kapott csonkított eredményt küldjük át a hálózaton, olyan adatokat utaztatunk, amiből nincs „visszaút"!



Vakarhatja a fejét Hacker Henry, hogy vajon mi lehetett az eredeti hatványkitevő! Hiszen a maradékos osztás eredményéből képtelenség visszaállítani a hatványozás eredeti értékét (így a kitevőt is), az ugyanis nyomtalanul eltűnt. Már csak az a kérdés, hogy ha a fenti táblázat minden számítását fejbevágjuk mod p-vel, vajon az (n ^ a mod p) ^ b mod p egyenlő lesz-e ennek párjával, a (n ^ b mod p) ^ a mod p értékkel?

Szerencsére a válasz határozott igen. A moduloval ugyanis tetszőleges helyen le lehet sújtani a számításokra: akár a legvégén, akár a részeredményekre modulózunk, a végeredmény ugyanaz lesz. *(Ez egyszerűen belátható egy számlapos falióra segítségével. Álljunk elé, és gondoljuk végig: tetszőleges időmennyiségeket adhatok össze, szorozhatok és hatványozhatok, teljesen mindegy, mikor jut eszembe mod 12-vel odacsapni, és egy részeredményt lefejezni, a legvégén mindig ugyanoda lyukadunk ki, az eredmény ugyanaz lesz.)*

Az Ember, Aki Középen Áll

Meg kell még vizsgálnunk, vajon Henry képes lenne-e az átmenő számok cseréjével becsapni a két felet? Ez azért lényeges kérdés, mert mint fent említettem, az SSL becsapható lenne. A publikus kulcs menet közbeni kicserélésével be lehet ékelődni egy SSL csatornába, ezt a tényt használja ki például az ISA Server, amikor SSL-bridge-et hozunk létre rajta keresztül. Lássuk, mit csereberélhet a Diffie-Hellman folyamat során egy köztes fél: "n" vagy "p" értékének kicserélése nem jó ötlet, hisz az egész matekozás arra alapul, hogy e két szám közös a feleknél. Ha kicseréljük valami másra, nem azonos eredményre jutnak a folyamat során, tehát nem lesz azonos kulcsuk sem. Ezzel legfeljebb a titkosított kommunikáció megakadályozását tudjuk elérni *(Denial of Service támadás)*.

A publikus kulcsok lecserélése azonban célravezető. Sajnos. Ha Alice és Bob közé beékelődik Claudia, akkor mindkettőjükkel megállapodhat egy-egy csak kettejük által ismert kulcsban. A Diffie-Hellman algoritmus nem áll ellen a "man-in-the-middle" típusú támadásoknak, ezért nem alkalmas vadidegenek közötti kulcscsere megvalósítására. S bár nyílt kulcsú, de nem titkosítási algoritmus, így semmilyen trükkel sem alkalmas adatok titkosítására.

Az IPSec korlátai

Nemrégiben megkérdeztem a Security listán, tudja-e valaki, hogy általában miért SSL-t használunk IPSec helyett publikus közegben. Sok tipp érkezett, sokan arra fogadtak, hogy azért, mert az SSL a TCP-csatornát, míg az IPSec az IP-forgalmat titkosítja. Na és? A valódi ok, hogy az RSA algoritmus, és a megbízható X.509 tanúsítvány miatt SSL-nél az ügyfél azonosítása nem szükséges, ám az IPSec alatt használt Diffie-Hellman a man-in-the-middle probléma miatt csak jól ismert felek között alkalmas kulcscserére. Ezért vagy Kerberos autentikációra van szükség, vagy mindkét félnek hiteles tanúsítvánnyal kell rendelkeznie, vagy *(nem vicc!)* előre megosztott szövegre alapuló azonosítást használhatunk.

A felek azonosítása IPSec-nél sajnos elkerülhetetlen. Most már tudjuk, miért.

Fóti Marcell
marcellf@netacademia.net
MZ/X

Távoli segítségnyújtás

Három évvel ezelőtt szorgos munka után a kollégáimmal sikerült egy teljesen automatizált Windows NT Workstation 4.0 telepítő környezetet létrehozni. Az operációs rendszert olyan kiegészítőkkel láttuk el, amelyek a termék megjelenése pillanatában még nem léteztek. Ilyen a WBEM, a WHS, a Time Service, és a VNC. Ezek közül a Windows 2000 majdnem mindent beépítve tartalmaz, A Windows XP azonban feltette a pontot az i-re, a távsegítség *(Remote Assistance)* szolgáltatás pótolta az utolsó hiányosságot is.

Mi az a távsegítség?

Mielőtt a részletekben elmerülnénk, tisztázzuk, hogy pontosan miről van szó. A szolgáltatást néha a betárcsázással *(Remote Access Service)*, néha pedig a terminal server *(TS, Remote Desktop)* nyújtotta funkcióval keverik össze.

A RAS végeredménye, hogy a telefonvonalból hálózati összeköttetést varázsolunk. A kapcsolat lassabb lesz és kevésbé megbízható, de elméletileg pontosan azokat a lehetőségeket nyújtja, mintha a vállalatunknál egy hálózatba kötött gép előtt ülnénk.

A terminal server már inkább hasonlít a távsegítséghez *(sőt mindkét szolgáltatás azonos technológiai alapokon nyugszik)*, de nem azonos a kettő. A TS egy saját munkakörnyezetet biztosít, hogy távolról is úgy dolgozhassunk, „mintha közel lennénk". A TS szolgáltatást nyújtó gép konzolja érintetlen marad, és semmi sem utal arra, hogy távolról a gépet használja valaki.

A Windows 2000 szervercsalád mellett a Windows XP is tartalmaz egy ilyen funkciót, azt távoli asztalnak *(Remote Desktop)* hívják. Mindkét szolgáltatás jellemzője, hogy „egyszereplős", azaz egyetlen felhasználó önálló munkáját segítik.

A távsegítség az a mágikus dolog, amikor egy másik felhasználó munkakörnyezetét láthatjuk, sőt, ha erre lehetőséget kapunk, akkor akár az egérkurzorát és billentyűzetét is használhatjuk. A funkciót eddig csak külső szoftverekkel lehetett megvalósítani. Ilyen volt a Symantec pcAnywhere célszoftvere vagy a VNC *(ez utóbbi ingyenes termék)*, de a szolgáltatás feltűnt a Microsoft SMS-ben, a CA RemoteIT-ben, és még a Dameware NT Utilitiesben is.

Mostantól a távvezérlés beköltözik az alapvető operációsrendszer-szolgáltatások közé, megoldva jó néhány szűkmarkúan pénzelt magyar rendszergazda problémáját.

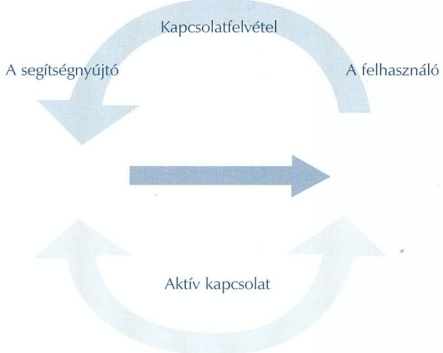
Távsegítség – ahogy a felhasználó óhajtja

A távsegítség alkalmazás nem telepíthető – a Windows XP része. Ez kényelmes a rendszergazdák és a felhasználók számára is. A funkció használatba vétele azonban egészen egyedi. Ha csoportházirend nélkül használjuk a távoli segítségnyújtást, akkor a funkciót csak a felhasználó kezdeményezése után indíthatjuk. Egy rendszergazda nem indíthatja el „csak úgy" a távoli segítségnyújtást, nem kapcsolódhat rá észrevétlenül a felhasználó képernyőjére, és nem veheti át az irányítást – nem kémprogramot kaptunk.

A Microsoft egy olyan megoldást akart, amely biztonságos, és a vállalati ügyfelektől az otthoni felhasználókig mindenkinek megfelel. Sokat is tett ezért, talán nem is mindig szerencsés megoldásokkal kísérletezve. A szoftver egyszerre szeretne a vállalati felhasználóknak és az otthoni gépbűvölőknek segíteni. A távoli segítségnyújtás három fázisban jön létre.

1. A felhasználó segítséget kér. Ez többféle módon, és szinte bármilyen környezetben megteheti, ahogy azt majd látni fogjuk.
2. A segítségnyújtó értesül a kérelemről, és válaszol rá.
3. A kapcsolat kiépül, a távoli segítség elkezdődhet.

Ha az Olvasó aggódik, mert úgy gondolja, az ő munkatársai kevésbé autonóm felhasználók, és még a segítségkérésben is segítségre szorulnak, akkor kérem megnyugtatni: ha bevezették már az Active Directoryt, akkor van mód a rendszergazda kezdeményezésére is. Nézzük először mégis azt a szituációt, amikor a felhasználó kezdeményez.

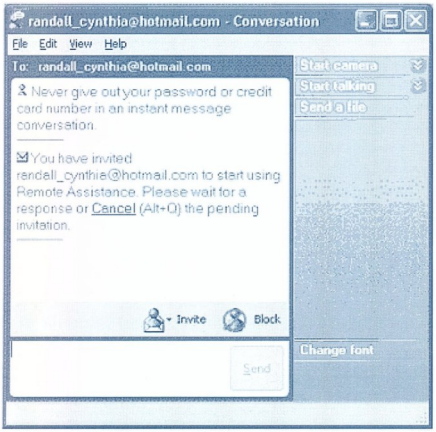


Így jön létre a távoli segítségnyújtás kapcsolat

A segítségkérésnek legalább három módja van. Először is kérhetjük Windows Messengeren keresztül *(2 ábra)*. Tools Ask for Remote Assistence

A megjelenő ablak tudatja velünk, hogy kit invitálunk a segítségnyújtásra. Ha ő is épp ugyanezt a csevegő-programot futtatja, máris megjelenik a képernyőjén a meghívás. Fontos kritérium, *(és a kapcsolat-felépítéstől független)* hogy a távoli segítség csak akkor működik, ha mindkét gép Windows XP operációs rendszert futtat – igaz, az lehet Professional és Home Edition is.

Ha a meghívott elfogadja az invitálást, elindul a távsegítség

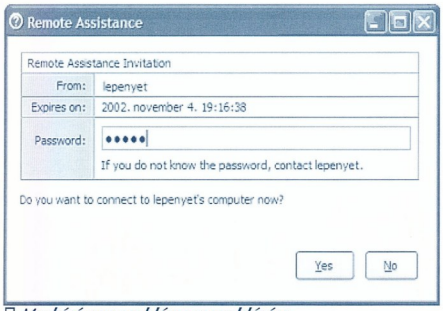


Van, akinél még nem vállalati szabvány a Windows Messenger?

program, a segítségkérő oldalon pedig egy párbeszédpanel figyelmeztet, hogy valaki a fenti módon a munkamenetünkhöz csatlakozik. Ezt egy határozott és tudatos egérkattintással engedélyezni kell, különben a képernyő nem jelenik meg a túloldalon. A szolgáltatás egészen biztosan az utolsó pillanatban készülhetett el, mert az alkalmazás menüje, feliratai és súgója minkét oldalon angol nyelvűek – hiába a feltelepített nyelvi csomag.

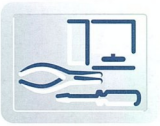
A segítségkérés másik módja egy speciális fájl elküldése levélben. El kell indítani a „Súgó és támogatás" programot, majd az Eszközök menüben a Távsegítség pontot kell választani. A jobb oldalon láthatjuk a meghívók állapotát, illetve új meghívást kezdeményezhetünk. Az új ablakban a „Meghívó mentése fájlként" lehetőséget kell választani. Egy jelszót megadva elmenthető a fájl, majd e-mail segítségével továbbítható. A fogadó oldalon a mellékletet lementve, majd megnyitva a következőt láthatjuk:

A küldő által megadott jelszó begépelése után kiépíthetjük a kapcsolatot. Gondot jelent az előzetes jelszócsere, ezért a biztonságos együttműködéshez igénybe kell venni egy másik kommunikációs csatornát is, például a telefont.



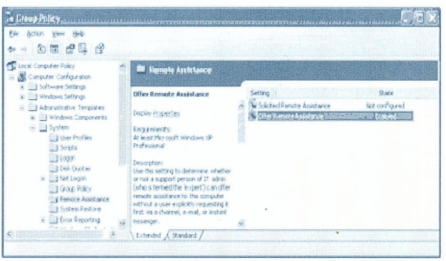
Meghívó egy probléma megoldására

A harmadik módja a kapcsolatfelvételnek egy egyszerű levélküldés – ezt ajánlja a gyártó vállalati környezetben. Ugyanazokat a műveleteket kell végrehajtanunk, mint a fájl mentésekor, csak a levélküldés pontot kell választani. Ekkor a MAPI profilunk felhasználásával automatikusan készül egy levél egy ugyanolyan melléklettel, mint amilyennel az előző módszernél találkoztunk. A probléma ugyanaz, a jelszóban valahogy meg kell állapodni előzetesen. Sőt, ha olyan vállalati levelezőprogramunk van, amely nem MAPI felületet használ, a fenti módszerrel egyáltalán nem élhetünk.



A felhasználókat kímélendő...

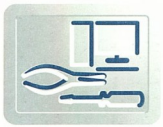
Szép, szép, hogy ennyi lehetőség közül választhat a felhasználó, de vállalati környezetben egyik sem az igazi. *(Hacsak a vállalati küldetés nem tartalmazza a felhasználók személyiségi jogainak, függetlenségének és teljes önállóságának deklarálását.)* A Windows Messengeres módszer, ha egy külső szerverhez csatlakozunk, nem biztonságos. Szükség lenne tehát egy Microsoft Exchange Serverre, ezt azonban nem mindenki engedheti meg magának. A levélküldés és a telefonos jelszócsere bonyolult és kényelmetlen, de az is lehet, hogy nem kivitelezhető. Legyünk jóindulatúak, és gondoljuk azt, hogy ezeket a lehetőségeket jobbára az otthoni felhasználóknak szánták. Ismerkedjünk meg azzal a módszerrel, ami valószínűleg a legelterjedtebb lesz a magyar vállalatoknál. Ez a megoldás sem mentes az ürömtől – csak Active Directory megléte és kizárólag a Professional változat esetén működik. A programkészítők ezúttal abból indultak ki, hogy a rendszergazda *(vagy a helpdesk)* fogja kezdeményezni a kapcsolatot. Ahhoz azonban, hogy ezt megtehesse, előbb érvényre kell juttatni egy beállítást a házirenden keresztül. Javaslom, hogy a Q307900 cikk alapján frissítsük a tartományi házirendsablonokat a Windows XP sablonjaival, majd a Computer configuration Administrative Templates System Remote Assistance útvonalon keresztül keressük meg az „Offer Remote Assistance" szabályt. Annyi a teendőnk, hogy bekapcsoljuk és megadjuk azoknak a felhasználóknak a listáját, akik jogosultak lesznek a távvezérlés felajánlására. Miután a beállítások érvényre jutottak az ügyfélgépeken, elhagyhatjuk a bizonytalan eredménnyel járó betanítási programot – a távsegítség kezdeményezhető a segítő oldaláról is. A „Súgó és támogatás"-t használhatjuk erre, ahogy arról már egy korábbi Tech.Net cikkben szóltunk.



A házirend most is segít a rendszergazdáknak

A program használata

Bárhogy kapcsolódtunk is össze a felhasználó gépével, a végeredmény ugyanaz lesz. A „Remote Assistance" keretben megjelenik egy ikonsor, egy kommunikációs panel, valamint a távoli felhasználó képernyője. A segítséget kérő

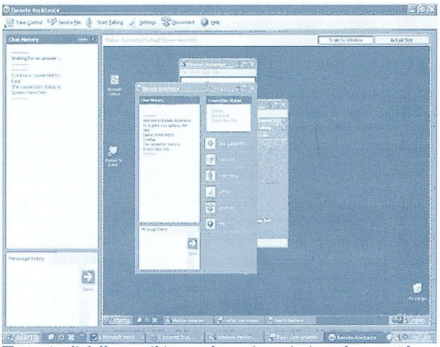


kolléga asztalán szintén megjelenik egy csevegésre felkészített pult, illetve a kapcsolattal összefüggő parancsok ikonsora. A felbontást változtathatjuk úgy, hogy vagy a felhasználó aktuális képernyőméreteit használjuk, vagy illesztjük azt a mi saját keretméretünkhöz. Ez a kicsinyítő technológia sokkal jobb képet ad, mint a VNC hasonló funkciója. A betűk továbbra is olvashatók, az ikonok felismerhetők maradnak. Ha nincs extrém különbség a két gép képernyőfelbontása között a felhasználó javára, elfogadható ablakot kapunk, és emellett használhatjuk a saját alkalmazásainkat is.

A szoftvert úgy konfigurálták, hogy a kezdeti képernyőt csak láthatjuk, de nem vezérelhetjük. Ha szeretnénk beavatkozni, akkor azt a „Take Control" gombbal kérhetjük.

Amennyiben ezt a felhasználó ismételten engedélyezte, már mindent szabad, amíg egy ESC gombot nem nyomunk.

A távolsági telefonbeszélgetések költségeit csökkentheti a



A távoli felhasználó asztala a távsegítségen keresztül

csevegőpanel. Használata nem szorul magyarázatra, dicséretül annyi elmondható, hogy jól elkülöníthető, ki mit mondott, továbbá a leírtak lementhetők a későbbi félreértések elkerülése érdekében. Ha a rendszereink és a hálózatunk jól felkészített, még hangcsatornát is felépíthetünk a két gép között. VIP felhasználóknál esetleg érdemes élni a lehetőséggel. Amennyiben szükségünk van rá, állományokat küldhetünk a két gép között. A „Súgó és támogatás" által készített diagnosztikai napló esélyes leginkább, hogy a műveletet elvégezzük rajta. Mivel az egyéb funkciók használata triviális, nem ecsetelem tovább a lehetőségeket, inkább a technikai részletekbe avatom be az Olvasót.

A technológiai háttér

A részletek taglalása előtt el kell mondanom, hogy korántsem bizonyos, hogy a kapcsolatfelvétel legelső három módja teljesen kihasználatlan marad. Előfordulhat, hogy valaki egy eladott számítógéphez távoli támogatást szeretne biztosítani például ezzel a módszerrel, de az is megeshet, hogy egy VIP kolléga otthoni gépét kell konfigurálni úgy, hogy szükség esetén távolról segíthessünk neki. Bármelyik esetben fontos lehet megérteni a szolgáltatás mögött rejlő komponensek működési elvét.

A távsegítség a Terminal Services és a Help and support szolgáltatás együttműködése révén épül fel. A kapcsolatfelépítést persze bonyolítani lehet a levelezés, az állománykiszolgálás, az AD vagy a Windows Messenger használatával, de ezektől most tekintsünk el. Technikai szempontból a segítségkérő egy XML állományt hoz létre, amelyet azután a fenti lehetséges csatornák egyikén eljuttat a rendszergazdának. Az állomány kiterjesztése .MsRcIncident. Ha kettőt kattintunk egy ilyen fájlon, a „Súgó és támogatás" program indul el, amely azután meghívja a távsegítséget.

Egy ilyen állomány létrehozásakor két esemény történik a felhasználó operációs rendszerében:

1. Az addig kikapcsolt és erős jelszóval ellátott „HelpAssistant" fiókot az operációs rendszer visszakapcsolja.
2. Egy bejegyzés készül a XML állományról. (Ezt lehet megtekinteni a „Meghívás állapotának megtekintése" menüpontnál.)

Az elkészült állomány információkat tartalmaz a felhasználó számítógépéről. Nézzünk meg közelebbről egy ilyen meghívót:

```
<?xml version="1.0" encoding="Unicode" ?>
 <UPLOADINFO TYPE="Escalated">
  <UPLOADDATA USERNAME="lepenyet"
PROBLEMDESCRIPTION="Gond van"
RCTICKET="65538,1,192.168.81.1:3389;192.168.1.1:3
89;192.168.40.1:3389;mal044.mal.priv:3389,83mY6/db
G0tDlj/EwXw=,kZM3ESypkjq0dhcxAlwJheQbkpi9jP7tw8eL/
dcmRIw=,SolicitedHelp,3lHb4vBPM1xQ6BiX2H+jDsbB+Che
ulQBV4OUfe32c3bhok0Pl6HSQ==,9hv0rmeu5d3Ex/bU2xL5g
q+gqsQ=" RCTICKETENCRYPTED="1"
DtStart="1036430198" DtLength="60"
PassStub="KJ#9Qjxlo6wWYV" L="0" />
  </UPLOADINFO>
```

A username és a problemdescription mező egyértelmű. Az rticketencrypted azt jelzi, hogy a felhasználó a jegy elkészítésekor megadott egy jelszót. A dtlength percben adja meg a jegy lejáratának idejét, az állomány keletkezésének időpontját pedig a DtStart tárolja.

A jegy központi része az alábbi információkat tartalmazza:

65538,1 – verzióinformáció.

192.168.81.1:3389;192.168.1.1:3389;192.168.40.1:3389;mal044.mal.priv:3389 – IP címek, FQDN nevek és portok, amelyek az állomány készítésekor léteztek a felhasználó gépén. A 3389-es kapuszám árulkodik arról, hogy a háttérben a terminal service működik.

83m...sQ= – a hitelesítéshez szükséges információk. A kapcsolódáskor a rendszergazda gépén a terminal szolgáltatás ezt az információt a felhasználó gépének HelpAssistant fiókjához rendeli, majd a bejelentkezési kérelmet a távoli gép GINA rendszeréhez továbbítja. Egy pillanatra fel is tűnik a képernyőn a Ctrl-Alt-Del képernyő *(a GINA látható része)*, de rögtön el is tűnik. Most már tudjuk, hogy milyen fiókkal zajlik le egy egyébként teljesen szabályos bejelentkezés.

A csatlakozás nem jöhet létre, ha a távoli segítség a felhasználó gépén nem engedélyezett. *(A kapcsoló eredeti értéke engedi a kapcsolat létrejöttét.)* Szintén nem jöhet létre a kapcsolat, ha a jegy lejárt. Ezt nem a rendszergazda, hanem a felhasználó gépén ellenőrzi a „távsegítség". A felhasználó bármikor lejárttá tehet egy invitálást, az XML állományban ezért leginkább tájékoztató jellegű a mező.

No és mi történik, ha egyáltalán nincs invitálás? Ekkor nincs XML állomány sem, hogyan lehet mégis bejelentkezni? Sajnos pontos információm egyelőre nincs, a kapcsolatfelépítés formája még nem dokumentált. Olyan nagy fantázia azért nem kell. Vélhetően a távoli *(rendszergazda)* gép aktiválja és megváltoztatja a HelpAssistant fiók jelszavát, majd belép a felhasználó gépére a fent leírt módon. A TS a „Súgó és támoga-

tás" programmal együttműködve elindítja a távsegítség alkalmazást.
Megismerve a részleteket láthatjuk, hogy sok-sok komponensnek kell együttműködnie ahhoz, hogy a kívánt végeredményt kapjuk. Ez egyben sok hibalehetőséget is jelent. A Windows XP bővelkedik a hibaelhárítás eszközeiben. Kérhetünk a helpdesktől távol segítséget, közben pedig megpróbálhatjuk a rendszer előző állapotát visszaállítani, hátha az segít. Ne tegyük! A System Restore ugyanis újra kikapcsolja a Helpassistant fiókot, és a távoli szakértő nem tud majd csatlakozni a gépünkhöz. *(Bővebben a Q304689 cikk szól a hibáról).*
Érdemes fejben tartani, hogy a szolgáltatás a terminal szolgáltatás fölé épül, ezért vannak olyan GPO beállítások, amelyek befolyásolhatják a működését. Elsősorban a kapcsolat idejére vonatkozó beállítások jutnak érvényre, nincs hatásuk viszont a színmélységet valamint a felhasználói nevet és jelszót megadó házirend beállításoknak. *(Q305898).* Akkor sem működik a szoftver, ha házirend-beállításokkal letiltjuk a terminal szolgáltatást.
Az XML kód – bármennyire is a jövőt jelenti – nem mindig üdvözítő. Tegyük fel, hogy egy lelkes felhasználó küld egy levelet, benne egy XML meghívót a rendszergazdának. Feltételezzük továbbá, hogy a mi rendszergazdánk Exchange 2000-et üzemeltet, és épp kószál egy munkatársnál. Kézenfekvő, hogy a levelet OWA-n keresztül nyitja meg. Csakhogy biztonsági okokból az OWA nem engedi az ilyen jellegű állományok megtekintését – így a meghívó használhatatlan *(Q322182).* Még eggyel több ok az AD üzembeállítására, és a meghívó elhagyására.

Különleges környezetben

Ma még talán nem általános gyakorlat, de a jövőben gyakrabban előfordulhat, hogy a Windows XP beépített tűzfalszolgáltatását bekapcsolják a rendszergazdák. (Egy MS konferencián egy előadó ezt kifejezetten ajánlotta.) Ez esetben gondoskodni kell arról, hogy a kérés kijusson a személyi tűzfalon, továbbá a kapcsolat-felépítés létrejöhessen, vagyis a megfelelő TCP kapuk legalább a meghívás idejére nyitva legyenek.
Szerencsére az XP-t felkészítették a tűzfalas környezetben va-

A tűzfal helye	Windows XP (ICS)	Windows Millennium ICS	Nem UPnP NAT eszköz	UPnP NAT eszköz
Felhaszn.	Igen	Igen	Igen	Igen
Rendszerg.	Igen	Igen	Igen	Igen
Felh. és rendszerg.	Igen	Igen	Nem	Igen

ló működésre. Az alábbi két táblázat segítségével megállapítható, hogy a kívánt kapcsolat létrejöhet-e a tűzfal megléte és a kapcsolatfelépítés módjának függvényében.
Amennyiben a Windows Messenger a kapcsolatfelépítés segédeszköze, a következő variációk fordulhatnak elő:
A táblázat azt mutatja, hogy UPnP NAT környezetben (ilyen rendszer a Windows XP és a ME ICS szolgáltatása) működik a

A tűzfal helye	Windows XP ICS	Windows Me ICS	Nem UPnP NAT eszköz	UPnP NAT eszköz
Felhaszn.	Igen	Igen	Nem	Igen
Rendszerg.	Igen	Igen	Igen	Igen
Felh. és rendszerg.	Igen	Igen	Nem	Igen

kapcsolatfelvétel, csupán két nem UPnP NAT eszköz „áttörése" lehetetlen.
Ha levél küldésével veszi fel a kapcsolatot a felhasználó, a következő lehetőségek adódnak:
A kapcsolat-felépítéshez szükséges, hogy a 3389-es kapu mind kifelé, mind befelé nyitva álljon. Mivel ez utóbbi általában tiltott, a távsegítség sem épül ki.
Még egy apróság: előfordulhat, hogy az invitálás után jut eszébe a felhasználónak, hogy az Internet veszélyes közeg. Ha ekkor kapcsolja be az ICS szolgáltatást, az szintén sikertelen kapcsolatot fog eredményezni. Ennek az a magyarázata, hogy megváltozik a portkiosztás valamint az IP cím, a változást pedig a korábban elküldött jegy nem tartalmazza. *(Q310608)*

Értékelés

Az alkalmazás előnyei egyértelműek: ingyenes, beépített, szoftver. Ha az AD integrált megoldást tekintjük, akkor biztonságos *(címtárral integrált).* Használat közben a képernyőfrissítés gyors, a nyújtott kép elfogadható, kevés konfigurációt igényel. Végezetül, *(és remélem ennek mindenki örül)* a felhasználó jogai is védettek - illetéktelenül senki nem háborgathatja, láthatatlanul nem kémlelheti.
A számos előny mellett azért bőven akadnak hátrányok is. Mindenekelőtt csak bejelentkezett felhasználó esetén működik, vagyis a bejelentkezéskor fellépő hibák elhárítására nem alkalmas. Kijelentkezéskor pedig nem látható, amikor a mappa-szinkronizáció miatt esetleg „fennakad" a PC, és nem indul újra. A szoftver csak angol nyelvű, ez épp az elesett felhasználókat zavarhatja.
Kizárólag XP-XP esetén működik, ami azt jelenti, hogy mostanában nem minden felhasználónkat fogjuk elérni.
Ha nem rendelkezünk a Messenger, a MAPI vagy az AD közül legalább az egyikkel, akkor komplikált a kapcsolat-felépítés. Márpedig létezik ilyen rendszer is. Egy AOL Instant Messengert használó, NDS címtárral működő Domino szervert birtokló cég gondban lehet.
Technikai problémák is akadnak. Több hálózati kártyás vagy hálózati csatolóval és modemmel rendelkező gépen gondok jelentkeznek, ahogy ezt a Q308210 cikk leírja (hotfixszel javítható).
Végezetül a rendszer valóban segítségnyújtásra való. A DVD lejátszás távolról nem megy. Aki azt gondolja, hogy ilyet még senki sem próbált, az téved. Erről tanúskodik a Q302899 cikk is.
Ezzel együtt azt gondolom, hasznos eszközzel gyarapodott a Windows család legújabb tagja.

Lepenye Tamás, MCSE 2000
lepenyet@mal.hu

Farkasokkal táncoló

(XII.rész) Cluster a gyakorlatban

A fürtnapló elemzését talán nem mindenki találja izgalmasnak. Ez érthető, ellenben az olvasóközönség ezen rétegének mára sem tudok semmi jót ígérni. Akik viszont felvették Sherlock „rendszergazda" Holmes szerepét, dörzsölhetik a markukat (vagy elégedetten pipázhatnak), mert mindjárt egy rejtély megfejtésével kezdjük. Pontosabban folytatjuk, hiszen ott tartottunk, hogy a nehezen összeszedett quorum.log-ot éppen töröltük.

```
[DM] DmpApplyChanges: Exit, returning 0x00000000
[FM] FmFormNewClusterPhase1, Entry.  Quorum
quorum will be deleted
```

Nos, gondoljuk csak végig, mi történt eddig. A fürtszolgáltatás elindult, konstatálta, hogy egyedül van, és magamagának kell minden erőforrását felélesztenie. Az „őskáosz és a semmi" állapotát egy ügyes trükkel, a fürtadatbázis helyi példányának beolvasásával ugrotta át. Az adatbázis segítségével a memóriában különböző objekutmokat, hálózati csatolókat, erőforrásokat stb. hozott létre. A szolgáltatás végül megtalálta a quorum erőforrást és annak helyét, a megfelelő lemezt birtokba vette, meggyőződött róla, hogy a helyes quorumra talált rá, végül az adatbázist beolvasta, pontosabban bemásolta a memóriába az esetleges chekpoint állományok alkalmazásával együtt.

Van tehát egy valódi, teljesen friss quorum adatbázisunk és egy olyan objektumhalmaz, amely ennek az adatbázisnak egy korábbi változata alapján jött lére. Előfordulhat, hogy a tényleges adatbázisban néhány objektum már nem szerepel, vagy épp ellenkezőleg, újakat tartalmaz. Célszerű tehát a korábbi objektumokat megsemmisíteni, és a memóriában tárolt érvényes adatbázis segítségével egy új, bizonyosan teljes objektumcsoportot létrehozni. Ez történik a következő fázis elején.

```
[FM] FmFormNewClusterPhase1, Entry.  Quorum
quorum will be deleted
[FM] DestroyGroup: destroying 9b26a6cb-a791-4807
9c17-bfc83050cd13
[FM] DestroyResource: destroying 54235f9f-3789
442a-98f1-e22bd2f73b66
[OM] Deleting object Cluster IP Address
(54235f9f-3789-442a-98f1-e22bd2f73b66)
[FM] FmpDestroyResource Exit.
[FM] DestroyResource: destroying 46fed62b-0817
441e-9a61-d6df53b58b7d
[OM] Deleting object Cluster Name (46fed62b-0817
441e-9a61-d6df53b58b7d)
[FM] FmpDestroyResource Exit.
[FM] DestroyResource: destroying f21ba2a4-62dd
4883-bf69-aa6c69f96320
[FM] FmpDestroyResource Exit.
[FM] FmpDestroyGroup: Group 9b26a6cb-a791-4807
9c17-bfc83050cd13 destroyed.
[OM] Deleting object MAL-CORPSERV3 (9b26a6cb-a791
4807-9c17-bfc83050cd13)
```

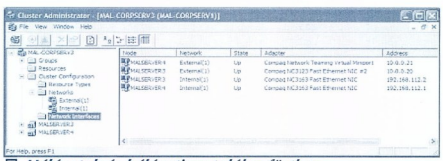
A metódust érdemes megfigyelni: az első törlendő a csoport maga, csakhogy az még további objektumokat tartalmaz. Ezért előbb az egyes erőforrásokat kell törölni, amelyet az FM végez el, majd leradírozható az objektum is, ez az OM dolga. Ha eddig nem volt teljesen világos az egyes fürt-komponensek feladata, most láthatjuk őket munka közben. A folyamat vége az erőforráscsoport objektumának megsemmisítése. Kezdődhet az új világ. *(Csak zárójelben jegyzem meg, hogy a művelet elvégzéséhez még egy ezredmásodpercre sem volt szükség, ha tehát valaki bonyodalmasnak és hosszadalmasnak érezte a fürt indulásának a módját, most megnyugodhat, az időveszteség elenyésző.)*

A következő műveletsor a hálózat beállítása, amely az alábbi lépésekből áll:

1. Állomás, hálózati csatolók és IP címek egyeztetése. (Az IP címek a hangzatos interfész nevet kapták.)
2. A rendszer jelenlegi hálózati beállításainak össze-gyűjtése
3. A hálózat és a csatoló objektumok létrehozása
4. Az előbb létrehozott objektumok regisztrálása a cluster transport segítségével
5. A hálózat és a csatolók „életre keltése"
6. Csatlakozási vizsgálat elvégzése, hálózati hibák felderítése
7. NodeHighestVersion és NodeLowestVersion értékek számítása *(lásd később)*
8. Javítások elvégzése *(vegyes környezet esetén)*
9. Az állapotváltozások rögzítése

A folyamatba ékelődik az erőforrásellenőr néhány bejegyzése, amelyek valószínűleg az FM által kiadott „cluster form process" utasítás hatására elvégzett műveleteket jelzik, de ezekkel most nem foglalkozunk. Látható még DM bejegyzés is, ám ennek pontos jelentése nem dokumentált.

Nehéz elképzelni a hálózatokat és a hálózati interfészeket mint objektumokat, pedig még a fürtadminisztrátor is bemutatja őket. Az elkövetkezőkben tehát a képen látható alkotóelemeket indítjuk el.



Hálózatok és hálózati csatolók a fürtben

```
[NM] Beginning cluster form process.
[NM] Synchronizing node information.
[NM] Creating node objects.
[NM] Creating object for node 2 (MALSERVER4)
[NM] Synchronizing network information.
[NM] Synchronizing interface information.
[NM] Running network configuration engine.
```

A szinkronizálás során a Node Manager *(NM)* összegyűjti az operációs rendszer aktuális információit az állomásokról, a hálózati kártyákról és az IP címekről, majd összehasonlítja az adatokat a fürt adatbázisában tároltakkal, hogy megállapítsa, történt-e bármilyen változás a rendszer szintjén a fürt legutóbbi leállítása óta. Az összehasonlítás műveletét az utolsó bejegyzés szimbolizálja. Ha bármilyen változás állt be, akkor azt a napló jelzi.

```
[ClNet] Tcpip is not bound to adapter D5F21544
E350-4B0A-9E07-7F5F1B01C154.
```

A fenti sor azt állítja, hogy az egyik adapterhez nem kapcsolódik a TCP/IP protokoll. A tény csak látszólag furcsa. A Windows 2000 tartalmaz egy MediaSense *(közegellenőrző)* mechanizmust, amely képes megállapítani, hogy egy adott hálózati kártya csatolójának túloldala „él-e". Ha nem, akkor a kártya azonnal „kapcsolat nélkül" állapotba vált, a protokoll pedig „lekapcsolódik" a kártyáról. A napló egy olyan fürtről származik, amelynek a magánhálózata csupán egy fordított UTP kábel. A másik állomást kikapcsoltam, a kártya túloldala nem működik, és lám, itt a bejegyzés a protokoll lekapcsolódásáról.

```
[NM] Processing network configuration changes.
[NM] Matched 2 networks, created 0 new networks.
[NM] Resynchronizing network information.
[NM] Resynchronizing interface information.
[NM] Creating network objects.
[NM] Creating object for network 50406024-5953
40da-8e15-ffad24274f4f (External(1)).
[NM] Creating object for network 52af5a7f-d330
4d34-8d5a-e28ca0b27018 (Internal(1)).
```

A változások átvezetésre kerülnek az adatbázisba, egy újabb szinkronizáció után pedig elkezdődik a hálózat-objektumok létrehozása *(meglepő módon ezt nem az OM, hanem a NM végzi)*. Egy adatbázisfrissítés után az interfész-objektumok (IP cím) létrehozása és bejegyzése következik.

```
Creating interface objects.
[NM] Creating object for interface 5964ef38-189f
4120-86e5-6ef100795c17 (Internal(1) - MALSERVER4).
[NM] Assigned index 0 to interface 5964ef38-189f
4120-86e5-6ef100795c17.
[NM] Creating object for interface 652483c9-1f77
451b-b1f3-463cc1e1cbc6 (Internal(1) - MALSERVER3).
[NM] Assigned index 1 to interface 652483c9-1f77
451b-b1f3-463cc1e1cbc6.
[NM] Registering network 52af5a7f-d330-4d34-8d5a
e28ca0b27018 (Internal(1)) with cluster transport.
[NM] Bringing network 52af5a7f-d330-4d34-8d5a
e28ca0b27018 online.
[NM] Registering interface 5964ef38-189f-4120
86e5-6ef100795c17 (Internal(1) - MALSERVER4) with
cluster transport, addr 192.168.112.1, endpoint
3343.
[NM] Registering interface 652483c9-1f77-451b
b1f3-463cc1e1cbc6 (Internal(1) - MALSERVER3) with
cluster transport, addr 192.168.112.2, endpoint
3343.
```

Érdemes megfigyelni, hogy amíg a hálózati csatolók esetén csak az adott állomás csatolóihoz keletkezett objektum, addig az interfészek esetén mindkét állomás adatait felhasználja a fürt. Az interfészeket azonos módon kell elnevezni a két állomáson. Olyannyira, hogy ha az egyik node-on átnevezzük a hálózati kapcsolatunkat, akkor a változás „replikálódik" a másik kiszolgálóra is. Aki nem hiszi, kipróbálhatja.

A cluster transport egy clusnet.sys állomány, amely a fürt állomásai közti kommunikációt bonyolítja a 3343-as UDP socketen keresztül. A regisztráció tudatja a komponenssel, hogy az adott hálózati interfész a kommunikációra felhasználható.

Az objektumok létrehozása és regisztrálása után meg kell győződni arról, hogy a hálózat az adott kártyákon valóban működik, elindul tehát egy kapcsolatellenőrző teszt.

```
Connectivity report worker thread running.
[NM] Processing local interface up event for
network 50406024-5953-40da-8e15-ffad24274f4f.
[NM] Updating local connectivity info for network
50406024-5953-40da-8e15-ffad24274f4f.
[NM] Started state recalculation timer (2400ms)
for network 50406024-5953-40da-8e15-ffad24274f4f
(External(1))
[NM] Updating local connectivity info for network
50406024-5953-40da-8e15-ffad24274f4f.
```

Nocsak! Az előbb említettük a MediaSense funkciót, itt mégis egy saját ellenőrzés zajlik. Igen. A fürtszolgáltatás kódját csupán kismértékben változtatták meg az eredeti NT4-es megjelenése óta, ott viszont még szükség volt egy saját mechanizmusra a hálózat ellenőrzéséhez. Olyan ez, mint a CPROXY: még itt van, még használjuk, de már nem lenne rá szükség.

A napló a következő sorokban kissé kaotikussá válik. A megkezdett tesztek között újabb műveletek indulnak és fejeződnek be, de minden műveletkezdést jelző sornak megtalálhatjuk a befejezést jelző párját is. Számunkra most a következő pár sor a fontos:

```
[NM] Beginning phase 1 of state computation for
network 50406024-5953-40da-8e15-ffad24274f4f.
[NM] Examining connectivity data for interface 0
(74e91f00-02e9-4892-bbb3-f09780b2e8dd) on network
74e91f00-02e9-4892-bbb3-f09780b2e8dd.
[NM] The report from interface 1 is not valid on
network 50406024-5953-40da-8e15-ffad24274f4f.
[NM] Interface 0 (74e91f00-02e9-4892-bbb3
f09780b2e8dd) is up on network 50406024-5953
40da-8e15-ffad24274f4f.
[NM] Node is down for interface 1 (b9f91c2f-3a56
4c52-8af1-7f52ef73cf70) on network 50406024-5953
40da-8e15-ffad24274f4f
[NM] Completed phase 1 of state computation for
network 50406024-5953-40da-8e15-ffad24274f4f.
[NM] Unavailable=1, Failed = 0, Unreachable=0,
Reachable=1, Up=1 on network 50406024-5953-40da
8e15-ffad24274f4f
```

A teszt összegzése után egy hálózat maradt, amely a későbbiek során használható; a GUID alapján láthatjuk, hogy ez az External (1). Adódik a kérdés, hogy miért kellett ezt az ellenőrzést elvégezni, amikor pár sorral feljebb már megállapítottuk, hogy az egyik adapterhez nem kapcsolódik TCP/IP protokoll. Nos, azért, mert elképzelhető lett volna az is, hogy ahhoz a kártyához szándékosan nem rendelünk IP-t *(csak pl. NWLinket)*, tehát az a kártya „nem játszik" a fürtben. A különböző GUID mutatja, hogy mást ellenőrzött az első és a második esetben a fürtszolgáltatás. Tanulságos, hogy hálózat (network), hálózati csatoló *(network interface)* és hálózati kártya *(adapter)* mind-mind mást jelent a fürt világában.

A naplóban pár sorral később láthatjuk, hogy az internal (1) hálózat is működik, ami annak tudható be, hogy közben bekapcsoltuk a második állomást is. Az persze még messze jár a fürt indításától, de a hálózati kapcsolat ellenőrzésekor már pozitív választ kapunk. Még nem fejeződött be a hálózatok és a hálózati interfészek el-

lenőrzése, de már egyéb folyamatok is elindultak, ahogy az alábbi sorokból ez kiderül.

```
[ClMsg] Initialized NTLM package.
[NM] Network 52af5a7f-d330-4d34-8d5a-
e28ca0b27018 is now in state 3
[NM] Interface 652483c9-1f77-451b-b1f3
463cc1e1cbc6 is up (node: MALSERVER3, network:
Internal(1)).
[NM] Network 52af5a7f-d330-4d34-8d5a-e28ca0b27018
(Internal(1)) is up.
[NM] Worker thread finished processing network
52af5a7f-d330-4d34-8d5a-e28ca0b27018.
[NM] Membership initialization complete.
[NM] NmpValidateNodeVersion: Node=1,
HighestVersion=0x00030893,
LowestVersion=0x000200e0
[NM] NmpCalcClusterVersion: status = 0
ClusHighestVer=0x00030893,
ClusLowestVer=0x000200e0
[NM] [NmpResetClusterVersion]
ClusterHighestVer=0x00030893
ClusterLowestVer=0x000200e0
[NM] Disabling mixed NT4/NT5 operation.
```

A hálózat alrendszere után a fürtszolgáltatás megállapítja a cluster pontos verzióját, ami az összes állomás verzióinak egyfajta eredője. Módszere a következő: az NM minden aktív állomástól lekérdez két számot, a NodeHighestVersion-t és a NodeLowestVersion-t. Az első az aktív állomás verziószáma, a második pedig az a verziószám, amellyel az állomás még tudna kommunikálni. Ezután kalkulálható a ClusHighestVer és a ClusLowestVer érték. Az előbbi a begyűjtött NodeHighestVersion értékek közül a legkisebb, az utóbbi viszont a NodeLowestVersion értékek közül a legnagyobb. A ClusHighestVer reprezentálja a teljes fürtre vonatkozó verziószámot, míg a ClusLowestVer az a szoftverkiadás, amellyel minimálisan rendelkeznie kell a fürtszolgáltatásnak ahhoz, hogy csatlakozhasson a közös rendszerhez. Minden egyes javítócsomag verzióváltást jelent, tehát a Windows NT 4.0 Enterprise Edition SP3 az egyes cluster verzió, az SP4 a kettes, az SP5 a hármas stb. Ha egy állomás csatlakozik a fürthöz, akkor a fenti értékeket újra kell számolni, és el kell juttatni minden állomásra.

A hálózat üzemkész állapota és a verziószámok megállapítása lehetővé teszi, hogy a fürttagságra vonatkozó információkat a quorum.log-ba lehessen rögzíteni. A naplóban ezt követhetjük a következő 10-11 sorban. A sorok között búját el a vegyes üzemmódot segítő fixup kód alkalmazása *([NM] NmPerformFixups Entry, dwFixup-Type=1)*, ám hogy pontosan mi történik, s hogy miért szükséges fixup tisztán W2K környezetben, azt nem tudni.

A következő fázis az erőforrások indítása. A feladatcsoport az erőforrástípusok felmérésével kezdődik. A szolgáltatás azt vizsgálja, hogy milyen erőforrástípusok találhatók a rendszerben, majd az egyes típusokhoz lehetséges állomásokat rendel. Egy erőforrástípus, a DHCP szolgáltatás esete tipikus, érdemes megvizsgálni.

```
[FM] processing resource types.
[FM] FmpQueryResTypeInfo: Calling
FmpAddPossibleNodeToList for restype DHCP Service
[FM] FmpAddPossibleNodeToList: adding node 1 to
resource type's possible node list
[FM] FmpAddPossibleNodeToList: adding node 2 to
resource type's possible node list
```

Miután minden típust megvizsgált a Failover Manager, egy JoinFixups2 műveletet hajt végre, ami aztán az adatbázisba is bekerül. A előző havi táblázatból kiolvasható, hogy a type 2 context 17 a PerformFixups2 műveletet rejti.

```
[NM] NmpUpdatePerformJoinFixups2: called postfixup
notifycb function with  status 0
```

Pontosan mit is jelent ez a PerformFixups2? Sajnos, nem tudjuk. Pedig talán magyarázatot adna arra, hogy az erőforrástípusokat miért kell másodszor is kiértékelni.

Az előkészületek után már létrehozhatók a valódi csoportok és erőforrások. A korábbi tapasztalatoknak megfelelően itt is előbb a csoportműveleteket hajtja végre a FM, s csak ha „észreveszi", hogy belső erőforrások is vannak, akkor kezd el velük foglalkozni. A módszer egyszerű: létrehozza a csoportot, inicializálja, megállapítja az előnyben részesített állomást, majd feltárja az erőforrásokat és azok függőségeit, létrehozza és inicializálja őket, és így tovább.

```
[FM] Processing groups list.
[FM] Creating group 48f4cb14-3e57-4199-bf8d
6c2a7a45ad89
[FM] Initializing group 48f4cb14-3e57-4199-bf8d
6c2a7a45ad89 from the registry.
[FM] Name for Group 48f4cb14-3e57-4199-bf8d
6c2a7a45ad89 is 'MAL-CORPSERV2'.
[FM] Group 48f4cb14-3e57-4199-bf8d-6c2a7a45ad89
preferred owner 2.
[FM] Group 48f4cb14-3e57-4199-bf8d-6c2a7a45ad89
contains Resource 1f9f8fdb-0779-44a2-8ad4
f04c6a2173c0.
[FM] Creating resource 1f9f8fdb-0779-44a2-8ad4
f04c6a2173c0
[FM] Initializing resource 1f9f8fdb-0779-44a2
8ad4-f04c6a2173c0 from the registry.
[FM] Name for Resource 1f9f8fdb-0779-44a2-8ad4
f04c6a2173c0 is 'MAL-CORPSERV2 IP'.
[FM] FmpAddPossibleEntry: adding node 1 as
possible host for resource 1f9f8fdb-0779-44a2
8ad4-f04c6a2173c0.
[FM] FmpAddPossibleEntry: adding node 2 as
possible host for resource 1f9f8fdb-0779-44a2
8ad4-f04c6a2173c0.
[FM] All dependencies for resource 1f9f8fdb-0779
44a2-8ad4-f04c6a2173c0 created.
[FM] Group 48f4cb14-3e57-4199-bf8d-6c2a7a45ad89
contains Resource a6536d7f-903f-42ef-bca5
6e48f3cafbe1.
```

A hosszadalmas műveletet a „[FM] All groups created." sor zárja. A folyamat látszólag unalmas és érdektelen. Ám ha egy erőforrással kapcsolatos hibát kell felderíteni, valószínűleg ez lesz az első naplórész, amit érdemes alaposan végigböngészni.

A végső állapot felé a következő lépés az erőforrás DLL állományokhoz kapcsolódó fixup műveletek elvégzése.

```
[FM] FmpFixupResourceTypesPhase1 Entry.
[FM] FmpFixupPossibleNodesForResType: dropping
CLUSCTL_RESOURCE_TYPE_STARTING_PHASE1 control code
to restype 'IP Address
…
[FM] FmpFixupResourceTypesPhase1 Exit
```

A „dropping" *(eldobás)* művelet arra utalhat, hogy nincs szükség a vegyes üzemmódú működést biztosító kódra.

A csoportok helyzete a következő:

```
[FMX] GetGroupListState, Group <MAL-CORPSERV3>
state = 3
[FMX] GetGroupListState, Group <MAL-CORPSERV4>
state = 1
[FMX] GetGroupListState, Group <MAL-CORPSERV2>
state = 1
[FMX] GetGroupListState, Group <MAL-CORPSERV1>
```

```
state = 1
```

A letölthető Excel táblából kiolvasható az állapotkódok jelentése. Ezek szerint az első csoport részlegesen működik *(hiszen a quorum lemezerőforrás működik)*, a többi még lekapcsolt állapotban van.
A FM és a MM üzenetváltása eldönti, hogy az erőforráscsoportok új tulajdonosa a jelenlegi állomás lesz.
A napló itt újra nehezen olvashatóvá válik a párhuzamos folyamatok egymást átfedő bejegyzései miatt. Az erőforrások inicializálásának koreográfiája azonban kihámozható:

1. A FM inicializálja az erőforrást a regisztrációs adatbázisból.
2. A művelet igénybe veszi az erőforrás ellenőrt is. Pl:

```
[FM] FmpRmCreateResource: creating resource
1f9f8fdb-0779-44a2-8ad4-f04c6a2173c0 in shared
resource monitor
```

3. Az erőforrás ellenőr elvégzi a tényleges munkát, s ha az erőforrásnak szüksége van a regisztrációs adatbázisra, akkor a GUM elvégzi a szükséges teendőket. Az Exchange Message Transfer Agent esetében például a következő zajlik:

```
[GUM] GumSendUpdate: Locker waiting type 1
context 0
[GUM] Thread 0x560 UpdateLock wait on Type 1
[GUM] DoLockingUpdate successful, lock granted to
1
[GUM] GumSendUpdate: Locker dispatching seq
35714371 type 1 context 0
[DM] DmWriteToQuorumLog Entry Seq#=35714371 Type=0
Size=148
[DM] DmpUpdateCreateKey: Creating key
<Resources\03244552-79e3-4165-a163
7c850f30b130\Parameters>...
[DM] DmWriteToQuorumLog Entry Seq#=35714371 Type=0
Size=148
[GUM] GumpDoUnlockingUpdate releasing lock
ownership
[GUM] GumSendUpdate: completed update seq
35714371 type 1 context 0
```

A folyamatot egy érdekes sor fűszerezi:

```
0000096c.000007e8::2002/07/19-19:39:10.560 [CP]
CppResourceNotify for resource Cluster IP Address
0000096c.000007e8::2002/07/19-19:39:10.560 [FM]
FmpRmOnlineResource: called InterlockedIncrement
on gdwQuoBlockingResources for resource 54235f9f
3789-442a-98f1-e22bd2f73b66
```

A sor pontos megértéséhez némi háttérinformációra van szükség. Előfordulhatnak olyan műveletek, amelyek azt igénylik, hogy a teljes és biztonságos befejezésük előtt semmiképpen se lehessen a quorum logot átmozgatni egy másik állomásra, mert az azzal jár, hogy a quorum adatbázist offline állapotba kellene állítani. A problémák elkerülésére a zárolás technikáját alkalmazták a fejlesztők. Kétféle zárolást ismer a fürt.

1. Megosztott zárolás *(shared lock)*: ilyen parancsot egy erőforrás adhat ki. Ez megakadályozza a fürtszolgáltatást, hogy leállítsa a quorumot.
2. Egyedi zárolás *(exclusive lock)*: A fürtszolgáltatás kérheti, ha a quorumot le akarja kapcsolni.

A két zárolási kérelem egyenrangú. Ez azt jelenti, hogy a megosztott zárolás lehetetlenné teszi, hogy egyedi zárolást alkalmazzon a fürt, ugyanakkor az egyedi zárolás feloldásáig nem lehet megosztottat alkalmazni.
Mivel több megosztott zárolási kérelem is érkezhet, ezért létezik egy gdwQuoBlockingResources nevű változó, amelynek minden kérelem beérkezésekor eggyel nő az értéke, a zárolás feloldásakor pedig csökken. Egyedi zárolás akkor alkalmazható, ha már minden szál feloldotta a maga zárolási kérelmét és a globális változó értéke nulla. Úgy kell elképzelni, mint a matematikai műveleteknél a zárójelet. Mindegyik nyitásnak van egy zárás párja, csak itt a jeleket az InterLockedIncrement és az InterLockedDecrement szavak jelzik.
A fenti zárolásnak csak „jóval később" jön a párja:

```
000097c::2002/07/19-19:39:19.653
[FM] FmpRmDoHandleCriticalResourceStateChange:
call InterlockedDecrement on
gdwQuoBlockingResources, Resource 1f9f8fdb-0779
44a2-8ad4-f04c6a2173c0
```

A fürt halad tovább: megjelennek a „type 0 context 8" tranzakciók, amelyek az erőforrások állapotváltozását rögzítik. A lemezerőforrások indítása ugyanolyan módon történik, mint a quorum lemezé, azzal a különbséggel, hogy a párhuzamosan futó egyéb műveletek eseményei nehezebben kihámozhatóvá teszik a naplót. A szakaszt ismét a fixup alkalmazása követi, íme az első sor.

```
[FM] FmpFixupResTypePhase2Cb: dropping
CLUSCTL_RESOURCE_TYPE_STARTING_PHASE2 control code
to restype 'IP Address', bFirst= 0
```

A teljes fázis lezárását a működő állapot rögzítése és a checkpoint állományok elkészítése jelenti.

```
[FM] FmFormNewClusterPhase2 complete.
[EVT] EvOnline
[EVT] Set propagation state to 0001
[EVT] EvOnline : calling ElfRegisterClusterSvc
[EVT] EvOnline: pPackedEventInfo->ulSize=61524
pPackedEventInfo->ulNulEventsForLogFile=6
[DM] DmUpdateFormNewCluster - taking a checkpoint
[LM] LogCheckPoint entry
```

A bejegyzések egyébként meglepőek, mert ténylegesen nincs szó arról, hogy az erőforrások valóban működnének. A tényleges indulást az jelzi, amikor a 129-es állapotból az erőforrás 2-es állapotba kerül, ahogy ez az alábbiakban látható.

```
0000096c.00000974::2002/07/19-19:39:12.544 [FM]
HandleResourceTransition: Resource Name =
2e284f11-66ae-40c3-beba-4bb2640d04a2 old state=129
new state=2
```

A checkpoint állomány kiírását követni meglehetősen nehézkes, mivel az egyes események között több száz más naplóbejegyzés is készülhet. Egy Excel tábla segítségével azonban összegyűjthető az összes LM esemény, és máris kirajzolódhat a pontos kép.

```
[LM] LogCheckPoint entry
[LM] DmpGetSnapshotCb: Checkpoint file
name=Q:\MSCS\chkF54D.tmp Seq#=35714381
[LM] LogCheckPoint: ChkPtFile=Q:\MSCS\chkF54D.tmp
Chkpt Trid=35714381 CheckSum=207707
[LM] LogpAppendPage : Writing 1024 bytes to disk
at offset 0x00001000
[LM] LogFlush : pLog=0x00098498 writing the 1024
bytes for active page at offset 0x00001400
[LM] LogCheckPoint: EndChkpt written. EndChkPtLsn
=0x00001408 ChkPt Seq=35714381 ChkPt
FileName=Q:\MSCS\chkF54D.tmp
```

```
[LM] LogpCheckpoint : Writing 1024 by
testo disk
at offset 0x00000000
[LM] LogCheckPoint Exit
```

Lassan a fürt elindulásának a végére érünk. Ugyan még sok bejegyzés készül, amelyek különböző regisztrációs adatbázisbeli módosításokat jelölnek, de a napló elejéről ismerős [INIT] bejegyzés mutatja, hogy a fürt munkára kész.

```
0000096c.000007e8::2002/07/19-19:39:12.685 [INIT]
Working Set changed to [1048576, 2097152].
0000096c.000007e8::2002/07/19-19:39:12.685 [INIT]
Cluster started.
```

A legutolsó sorok arról tanúskodnak, hogy a fürt DNS regisztrációt végez a cluster csoport hálózati név erőforrása számára.

```
[EVT] s_ApiEvPropEvents: Calling into
EvpPropPendingEvents, size=700...
[EVT] s_ApiEvPropEvents: Called
EvpPropPendingEvents...
[EVT] OUTQ: dropped 1, total dropped size 61524.
[EVT] s_ApiEvPropEvents: Calling into
EvpPropPendingEvents, size=660...
[EVT] s_ApiEvPropEvents: Called
EvpPropPendingEvents...
Network Name <Cluster Name>: Registered server
name MAL-CORPSERV3 on transport \Device\NetBt_If1.
Network Name <Cluster Name>: Registered
workstation name MAL-CORPSERV3 on transport
\Device\NetBt_If1.
Network Name <MAL-CORPSERV1  Network Name>:
Registered server name MAL-CORPSERV1 on transport
\Device\NetBt_If3.
Network Name <Cluster Name>: Registered DNS name
mal-corpserv3.mal.priv on IP Address 10.0.0.26.
Network Name <Cluster Name>: Registered DNS PTR
record 26.0.0.10.in-addr.arpa. for host mal
corpserv3.mal.priv.
[EVT] s_ApiEvPropEvents: Calling into
EvpPropPendingEvents, size=736...
[EVT] s_ApiEvPropEvents: Called
EvpPropPendingEvents...
```

A regisztrációt körülvevő sorok már a „semmittevő" fürt bejegyzései. Pontos jelentésüket ugyan nem tudjuk, de a tapasztalat azt mutatja, hogy ha ezek a bejegyzések megjelennek, akkor a fürtünknek nincs mit a naplóba írnia.

Összefoglaló

Bármennyire is hosszadalmasnak és részletesnek tűnt a fürt indulásának elemzése, még mindig csak bevezetésnek számít a napló analizálásakor. *(Megnyugtatom az Olvasót: most nem folytatjuk.)* Amit láttunk, az a fürt indulása – 10 másodperc, tulajdonképpen hiba nélkül. A legfontosabb fogalmakat elsajátítottuk és megértettük, ám ez nem biztos, hogy ez elég az üdvösséghez. A fürtnaplóra akkor van szükség, amikor baj van, valami nem indul, nem megy, nem működik. A cikkek ezekre a helyzetekre nem tértek ki. Nem is térhettek, hiszen sok száz hiba felsorolására és mintasorok közzétételére nincs mód. Adódik a kérdés: vajon elégséges a megszerzett tudás a hibák behatárolásához? A válasz: talán.

Látni kell, hogy a cluster.log mindenekelőtt a fejlesztők és nem a rendszeradminisztrátorok eszközének készült. Ez abból sejthető, hogy tele van kódokkal, GUID azonosítókkal, tranzakció-számokkal. Jelenlegi formájában túlságosan lakonikus ahhoz, hogy a hibát megtaláljuk. Még ha jelez is hibát a log, gyakran nem dokumentált a hibakód jelentése, vagy a net helpmsg parancs semmitmondó feloldást ad.

Érdemes akkor egyáltalán foglalkozni a cluster.log-al? Mindenképp. Nem csak a fürt belső szerkezetét lehet jobban megismerni, hanem a problémák gyökerének megállapítása is könnyebb – talán nem annyira gördülékeny, mint kellene, de támpontot biztosan nyújt. Ha pedig egyszer egy hibát megfejtettünk, a napló olvasásával azonosítani lehet egy másik környezetben is a problémát, mivel a log hasonló mintát fog mutatni. Azt ajánlom tehát, hogy minél többször használjuk ezt az eszközt, szerezzünk gyakorlatot, hogy amikor tényleg nagyon fontos lesz a gyors helyzetfelismerés és cselekvés, akkor a lehető legnagyobb munícióval indulhassunk. A sorozatban egy időre elkanyarodunk a cluster.log elemzésétől, de ígérem, hogy összegyűjtöm az általam vagy mások által megfejtett fürthibákat a hozzájuk tartozó naplókkal együtt, és egy más alkalommal mindenképp ismertetem a hibák adta mintákat és értelmezésüket. Kérek egyúttal mindenkit, ha van megfejtett fürtproblémája és hozzá tartozó cluster.log, akkor küldje el nekem, így hamarabb elkészülhet az ismertető.

Végezetül feltehetjük a kérdést, hogy vajon elégséges információval és eszközökkel rendelkezünk-e ahhoz, hogy hatékonyan hárítsuk el a fürt hibáit. Úgy gondolom, hogy a Microsoftnak még sok tennivalója van ezen a téren. Két lehetséges úton haladhat a cég a jobb hibafelderítés elősegítésére. Az egyik egy referenciakönyv elkészítése a lehetséges naplóbejegyzésekhez. A módszer hasonló lenne, mint amit a registry megismertetésekor alkalmazott a redmondi cég. Talán nem tökéletes módszer – de mérnöki.

A másik út az, ha szorosabb integrációt valósít meg a fürt és a Windows 2000 eseménynaplózó alrendszere között. Megjelenhet a DNS-hez és FRS-hez hasonlóan egy külön napló, vagy az alkalmazás logba kerülhetne több esemény. Akár az Exchange példáját is követhetnék a fejlesztők. A naplózási szinteket komponensenként lehetne meghatározni, és ha szükség van rá, akkor bekapcsolva egyre részletesebb eseménysort láthatnánk. Hosszú távon persze ez sem mentené fel a céget a referencia könyv *(vagy legalább referenciasúgó)* elkészítése alól. Aki kíváncsi, hogy a Microsoft melyik utat választja, annak javaslom a .Net Server tanulmányozását…akár már a következő hónapban ;-)

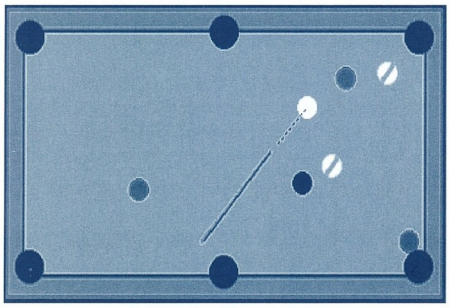
Lepenye Tamás, MCSE 2000
lepenyet@mal.hu

Az UML – 2. rész

Az előző részben igyekeztem szemléletesen bemutatni az UML felépítését, nézeteit és diagramjait. Arról azonban nem szóltam, hogy az első ötleteléstől hogyan, mi módon juthatunk el a kész termékig. Ezúttal igyekszem pótolni ezt a hiányt, és megválaszolni a felmerült kérdéseket.

Mindezt egy konkrét példán keresztül teszem meg. Tegyük fel, hogy megrendelőnk egy számítógépes biliárdjátékot szeretne készíttetni velünk. A konkrét megvalósításról nincs elképzelése *(valószínűleg nem is informatikai szakember)*, azonban a játék szabályait pontosan ismeri, ez alapján kell megvalósítanunk a rendszert.

Szabályok:
- A dákóval csak a fehér golyót lehet meglökni.
- Aki az első golyót a lyukba löki, az ettől kezdve ilyen színű golyókkal van, míg ellenfele az ellenkező színűekkel.
- Ha valaki csak saját színű golyót juttat lyukba, akkor még egyszer jöhet.
- A soron következő játékos kétszer jön, ha társa
  - a fekete golyót vagy ellenfele egy golyóját találja el először;
  - ellenfele golyóját is lyukba juttatja;
  - nem koccantja a fehér golyót más golyóhoz;
  - a fehér golyót a lyukba juttatja (ekkor a fehér golyót kezdő pozíciójának környezetébe helyezzük vissza);
  - kivéve, ha valakinek elfogyott az összes golyója az asztalról.
- Ha valamely játékos utolsó golyóját is lyukba juttatta, akkor az utolsó leeső golyó lyukával átellenes lyukba kell a fekete golyót juttatnia.
- A játékot az nyeri, akinek ez elsőként sikerül.
- A játékot elvesztette valaki, ha a fekete golyót idő előtt vagy rossz lyukba juttatta, illetve ha jó lyukba juttatta, de ezzel együtt a fehér golyó is lyukba esett.



A biliárdasztal képe a dákóval és golyóval

Első lépés: az igények felmérése

A megrendelőtől tehát csak köznapi nyelven kapjuk meg az igényt, ez alapján kell elindulnunk a megvalósítás felé. Egy szoftver fejlesztése azonban meglehetősen egzakt folyamat, pontos, szabatos fogalmak használatát igényli. Ezért szükséges, hogy a kapott definíció alapján elkészítsük a rendszer szótárát, amelyben definiáljuk a fontosabb szavak, fogalmak pontos jelentését. A fenti szabályoknak megfelelő szótár egy részlete például ez lehet:

Fal:
Az asztal széle, az asztal síkjára merőleges. A játék során a helyzete nem változik.

Lyuk:
Az asztal kör alakú része, amely elnyeli a golyókat.

Dákó:
Ezzel az eszközzel lökjük meg a golyót.

Játékosok:
A játékot játszó személyek. Két csapatra oszlanak, és a két csapat felváltva lökhet (kivéve, ha a szabályok másként nem rendelkeznek).

Golyó színe:
Egy golyónak 4 különböző színe lehet: csíkos, sima fekete és fehér.

Lökés:
A dákóval kezdősebességet adunk a fehér golyónak.

Ütközés:
A golyók egymással, vagy a fallal való érintkezése.

Felmerülhet a kérdés, hogy milyen fogalmakat érdemes belevenni a szótárba. Honnan állapíthatjuk meg, mi lényeges és mi lényegtelen a megvalósítandó rendszer szempontjából?
Néhány irányelv alapján könnyedén válaszolhatunk erre a kérdésre. Először is fel kell térképeznünk a rendszer szereplőit, akik valamilyen módon részt vesznek a folyamatokban. Esetünkben ilyen szereplő a két játékos *(akik lökéseikkel irányítják a játék menetét)*, a dákó, a golyók, a fal és a lyukak. *(Ez utóbbiak jelentősége abban rejlik, hogy létükkel és elhelyezkedésükkel befolyásolják, megváltoztatják a golyók irányát, sebességét, stb.)*. A szereplők a felhasználói specifikációban többnyire főnévként jelennek meg, és vagy ők végzik a cselekvést *(játékos)*, vagy passzív résztvevői annak *(dákó, golyó stb.)*
A szereplőkhöz különböző jellemzők is tartozhatnak, melyek közül a kiemelkedő fontosságúakat szintén szerepeltetni kell a szótárban *(pl. a golyó színe)*.
A cselekvések és események jelentik a program működésének másik kulcsát, így azokat is pontosan meg kell magyarázni a szótárban *(mit jelent például a lökés vagy az ütközés)*. Ezeket többnyire a felhasználói specifikáció igéi vagy igenevei közül szűrhetjük ki *(lökés)*, illetve az alapján adjuk hozzá a triviálisan adódókat *(ütközés)*.

Követelményspecifikáció

A felhasználó elvárásainak dokumentálása a fejlesztés első lépése. Sajnos bármilyen részletes is ez a dokumentáció, a legapróbb gondosság mellett sem tartalmazhat minden apró részletet. Nehéz megállapítani, hogy mi az a részletezettség, amely szükséges és elégséges is, hiszen ez a különböző szakterületektől, a céloktól, a környezettől és még sok egyéb tényezőtől függően nagyon változó.
A követelményelemzésnél továbbra is kívülről szemléljük a rendszert, és nem bonyolódunk bele a technikai részletekbe.

Ez a munka nagyon sok megbeszélést, egyeztetést és a felhasználóval való szoros együttműködést igényel. Üzleti folyamatelemzési technikákat és use case-eket használunk, célunk pedig (vázlatosan) definiálni a készülő rendszer funkcionalitását, az elemek viszonyát kifejező együttműködési diagramokat, vagy a folyamatok feltérképezéséhez szükséges aktivitás-diagramokat.

Már ebben a fázisban is lehet végezni különböző költség- és erőforrásbecsléseket, azonban ezek még meglehetősen durva, nagyvonalú közelítések, csak tájékoztató jellegűek lehetnek.

A biliárd esetében először tisztázzuk a fizikai részleteket. A felhasználóval egyeztetve rögzítenünk kell azokat a fizikai törvényeket, amelyek a játék során hatnak, ezeket definiálni kell, és itt szükséges annak tisztázása is, milyen pontosságot várunk el a megvalósítandó programtól.

Fizikai részletek

- ütközést detektálunk, ha két golyó középpontjának távolsága nem több a sugár kétszeresénél (megengedett maximális egymásba csúszás a sugár 1/10-e)
- két ütközés közötti lassulás megvalósítására időben lineáris függvényt alkalmazunk
- fallal és egymással való ütközés nem ideális, az ütközés utáni sebesség az ideális sebesség 95 %-a

Apróságnak tűnő, de lényeges fizikai jellemzője a játékban részt vevő tárgyaknak a mérete. Pontosan meghatározott arányban kell állnia a golyók, a dákó, az asztal méretének egymással ahhoz, hogy a számítógépes játékmodell a valós életben megszokottat élethűen tükrözhesse.

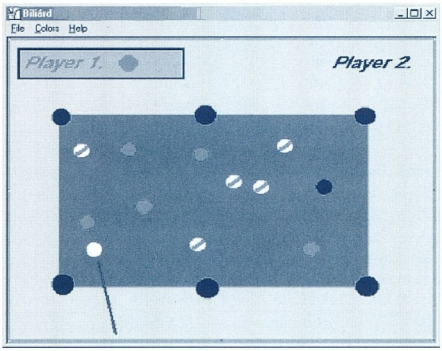
A követelmények között rögzíthetünk olyan részleteket is, amelyek nem lesznek részei ugyan az aktuális rendszernek, azonban későbbre olyan bővíthetőségi lehetőséget tartalmaznak, amely már most jól körvonalazható és leírható. Például:

A jobb játszhatóság érdekében a játékba a lökésnél (az erősség és az irány megadásánál) bevehető a véletlen. Ezt a funkciót az első verzió még nem fogja tudni, de a kódban meghagyjuk a lehetőséget a fejlesztésre.

Definiálni kell a felhasználói felület alapjait is: mit vár a megrendelő, milyen funkciók hogyan legyenek kivezetve az interfészre. A menük alapfelépítését, az esetleges ábrák elhelyezkedését, igény szerinti gyorsbillentyűket stb. határozhatunk itt meg, például az alábbi módon:

- A kezelői felületet alapvetően három részre oszthatjuk: (felülről lefelé) vezérlési rész, játéktér, állapotsor.
- A vezérlési rész tulajdonságait tekintve lehetőséget biztosít a kilépésre, az újrakezdésére (a játék végén automatikusan lehetőséget nyújt az újrakezdésre), valamint a játékosok adatainak bevitelére. A játék közben, ha az összes golyó megállt, mentési lehetőség is rendelkezésre áll.
- A játékrészen megjelenik az asztal a lyukakkal, a golyók és a dákó, természetesen méretarányosan. A lyukba esett golyók a játéktérről eltűnnek és ezek után már nem vesznek részt a játékban (kivéve, ha a fehér golyóról van szó).
- Az állapotsor kijelzi a következő lökő játékos nevét, golyóinak színét és a játék állását.

Természetesen vázlatos képernyőterv is készíthető, amely a fejlesztés további szakaszaiban még módosulhat – a megrendelővel egyeztetve. A biliárd esetében a valódi asztal képe alapján nem nehéz elképzelni, hogyan nézhet ki a játékfelület.



A biliárdjáték képernyőterve

Fontos tisztáznunk a rendszer életciklusmodelljét is, azaz azt, hogy egyedi fejlesztésről van-e szó, vagy tovább értékesíthetjük a megrendelő beleegyezésével, avagy a nélkül. Itt szükséges a fejlesztés ütemének meghatározása, a fontosabb részfeladatokra lebontva. Pontosan le kell írnunk azokat a feltételeket, amelyeket a készterméknek teljesítenie kell *(a futtatható programon kívül ide értendők a megfelelő dokumentációk is).*

A biliárdprogram esetében rendszernek magát az asztalt tekintjük, a falakkal, lyukakkal és golyókkal egy egységet alkotva. A felhasználó szemszögéből nézve ez önálló „életet él", működésébe beavatkozni csak néhány ponton lehet *(játék indítása, golyó meglökése stb.).*

Rendszerünkkel ketten állnak kapcsolatban *(ők a rendszer aktorai):*

Maga a **játékos,** a külső felhasználó, aki a golyókat meglöki, illetve egyéb utasításokat ad a rendszernek, mint például a játék újraindítása, elmentése, betöltése. A játékos a billentyűzeten és az egéren keresztül kommunikál a rendszerrel.

A **játékfelügyelő** egy programmodul, ami a szabályok betarttatásáért felelős. ő utasítja például a rendszert, hogy a fehér golyót helyezze vissza az asztalra, ha az valamelyik lyukba beleesett. Ennek a modulnak a megfelelő kialakításával elérhető, hogy a programban akár több szabály szerint is játszhassunk. A játékfelügyelő modul a programon belül, objektumok közötti üzenetek váltásával kommunikál. Működése a játékosok számára transzparens, láthatatlan.

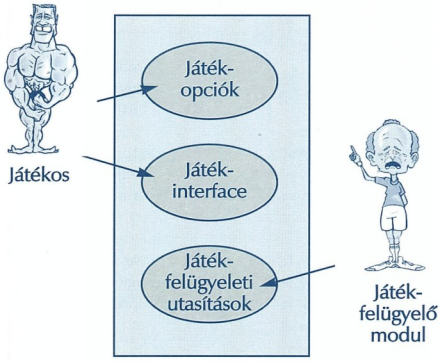
Látható, hogy a játékos kétféle, a felügyelő modul pedig egyféle utasításokat adhat. Ezek alapján három use case különböztethető meg:

- **Játék interface:** közvetlenül a játékhoz tartozó utasítássorozat. A játék elején, majd azután minden körben, miután a golyók megálltak a játékos neve megjelenik a képernyőn, majd a játékos kiválasztja a lökés irányát, ezután annak erősségét. Ezzel a kommunikáció véget ért, a játékos nem adhat újabb utasításokat a rendszernek, amíg a golyók meg nem állnak.
- **Játék opciók:** a játék interface kiterjesztése, mely során a játékos a golyók összességét befolyásoló utasításokat adhat, eldöntheti, hogy kilép a programból, újraindítja a já-

tékot, elmenti annak állását, esetleg betölt egy korábbit. Ez a kommunikációs csatorna is a golyók megállásakor aktiválódik, és véget ér, ha a játékos meglökte a golyókat, vagy kilépett a programból. Ha az állás újrakezdést/mentést/töltést választja, a use case azonnal újraindul.

- **Játékfelügyeleti modul:** a játékfelügyelő beavatkozása a rendszerbe. Működése a megfelelő események felismerésekor indul *(golyó lyukba került)*, a modul eldönti, hogy a szabályok alapján mi a teendő, majd utasítást ad a megfelelő cselekvésre *(színes golyó esetén annak levételére, fehér golyó esetén annak visszahelyezésére ad utasítást)*. Az utasítás végrehajtásával ér véget a use case.

Korábban már volt szó arról, hogy a használati eset *(use case)* a rendszer viselkedését, funkcionalitását írja le a szereplők és a feladatok megjelölésével, a felhasználó szemszögéből nézve. A fenti szereplők és szerepek alapján most tehát már felrajzolhatjuk a rendszer használati eset diagramját:



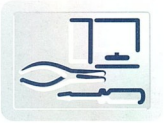
A biliárd játék use case diagramja

A követelményfelmérési fázisban tehát eljutottunk oda, hogy a felhasználói igényeket pontosan rögzítettük, definiáltuk, kitűztük a fejlesztés célját. Körvonalazódtak az elvégzendő feladatok, ennek megfelelően kialakítható az a csapat, amely ezek megoldásán dolgozik majd. Személyre szabottan kiosztathatók az egyes feladatok. Egy olyan, számunkra és a megrendelő számára ideális modellt dolgoztunk ki, amely mindenki igényeinek és korlátainak megfelel.

Osztályok leírása

Eddig csupán kívülről szemléltük rendszerünket. Meghatároztuk a megvalósítandó szoftverrel szemben a felhasználó által támasztott követelményeket, tudjuk, mit kell tennie majd a programnak, azonban még nem szóltunk a hogyan-ról. Ebbe általában a megrendelő már nem szól bele, nincs igénye *(és többnyire kellő ismeretei sem)* arra, hogy megmondja, milyen osztályokat, kapcsolatokat, együttműködéseket implementáljunk. Számára az a fontos, hogy a program működjön *(esetleg egy általa előre kikötött hardver/szoftver konfiguráción)*. Kritikus lehet még esetleg a sebesség vagy a felhasználóbarát interfész, de ezek mind olyan kritériumok, amelyek a belső megvalósítástól függnek.

Előttünk tehát a feladat, hogy mindezen követelményeknek megfeleljünk. Az igények sokoldalúságából következik, hogy a készítendő rendszert pontosan, gondosan meg kell tervezünk. A belső tervezés első lépése a különböző osztályok, objektumok meghatározása, feladatainak specifikációja, a közöttük lévő kapcsolatok feltárása. Ez alapján aztán már felrajzolható a statikus struktúradiagram.



Fontos, hogy az osztályok meghatározásánál a célszerűségre törekedjünk. Derítsük fel, milyen nagy objektumkategóriák szerepelnek a rendszerben, melyek általánosíthatók, melyeket kell ezek közül konkrétabbá tenni. Vizsgáljuk meg az egyes kategóriák közötti kapcsolatot, együttműködéseket, üzenetváltásokat. Lássuk, hogyan néz ki mindez a biliárd esetében. Gyűjtsük össze eddigi ismereteink alapján a szükséges osztályokat, és röviden írjuk le viselkedésüket:

Szabályok
Egyetlen példánya van.
Felelőssége ellenőrizni a játékszabályokat. (Továbbfejlesztés esetén ennek több példánya is lehet, lehetőséget nyújtva különböző típusú biliárdjáték játszására.)

Input interface
Egyetlen példánya van.
Figyeli a játék menetébe kívülről érkező beavatkozásokat.

Lökéskezelő
Egyetlen példánya van.
Beolvassa a játékosok lökéseit.

Menüutasítások
Egyetlen példánya van.
A felhasználói felület három egységéből (vezérlési rész, állapotkijelző, játékrész) a vezérlési rész működését koordinálja.

Asztal
Egyetlen példánya van, a biliárdasztal.
Felelősségei: Figyeli a golyók mozgását, az ütközéseket. A megfelelő golyókat egymással és a falakkal ütközteti.

Golyó
Példányai a játékgolyók.
Feladatai: tárolja a golyó helyét (asztalon, lyukban van-e, az asztalon hol van), és hogy merre gurul, milyen sebességgel.

Asztal-, golyórajzoló
Egyetlen példánya van.
A felhasználói felület három egységéből a játékrészt jeleníti meg, a megfelelő paraméterek birtokában kirajzolja a kezdőállapotot, majd a játék során a többi objektumtól kapott információk alapján frissíti a látványt.

Játékállapot-kijelző
Egyetlen példánya van.
A felhasználói felület három egységéből az állapotkijelző karbantartását végzi a kapott adatok alapján.

Lökés-/Dákórajzoló
Egyetlen példánya van.
Feladata a játék során a lökés műveletének látványtechnikai megvalósítása.

Menürajzoló
Egyetlen példánya van.
Felelőssége a felhasználói felület három egységéből a vezérlési rész megjelenítése.

Az osztálykatalógussal szinkronban az objektumkatalógus is felírható, ennek elemei a játékprogram objektumai és azok leírása:

Szabály
A main indítja el. Figyeli a játék és a pillanatnyi lökés állását. Miközben a golyók mozognak, gyűjti az adatokat. Miután értesítést kapott az összes golyó megállásáról, kiértékeli a történteket (ki következik, mely golyókat kell visszarakni, vége van-e a játéknak)
Ezután törli az események listáját, így felkészülve a következő lökésre, majd elindítja az input interface-t.

Input interface
Kezeli a billentyű, későbbiekben az egér bemenetet. Az inputot a Lökéskezelőnek és a Menüutasításoknak továbbítja

Menüutasítások
A következő menüopciók vannak:
- Játék újraindítása
Ehhez közvetlenül hozzátartozik a játékosok nevének lekérdezése, majd utasításküldés a Szabályobjektumnak, hogy helyezze el a golyókat az asztalon
- Játék mentése
- Játék visszatöltése

Lökéskezelő
A dákó mozgatását, és a lökés erősségét kezeli. Mikor a lökés megtörténik, a paramétereket áradja az asztalnak.

Asztal
Miután a lökéskezelőtől megkapta a fehér golyó kiindulási sebességét és irányát, szimulálja a golyók mozgását. Minden lényeges eseményről (első koccanás, golyó lyukba került, minden golyó megállt) üzenetet küld a Szabályobjektumnak

Golyó
Az asztal tulajdonában van, négy leszármazottja a fehér, a fekete és a hét teli és csíkos golyó

Csíkos golyó
Hét darab
Szülősztály: Golyó

Teli golyó
Hét darab
Szülősztály: Golyó

Fekete golyó
Egy darab van belőle
Szülősztály: Golyó

Fehér golyó
Egy darab van belőle. Csak ezt kehet meglökni
Szülősztály: Golyó

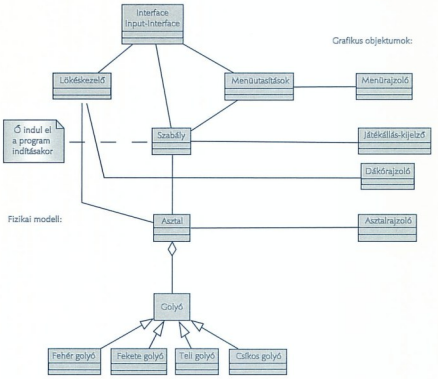
Dákórajzoló
A lökés során a dákó képét rajzolja, ill. a lökés erősségét (terv: a dákó forog a fehér golyó körül, a lökés erősségét a dákónak a golyótól való távolsága mutatja)

Menürajzoló
A menü kirajzolásáért felelős. (Első verzióban csak kiírás a gombkombinációkról, legördülő menü előreláthatólag a későbbi verziókban)

Játékállás-kijelző
A játék állásáról tudósít: ki következik, kinek hány golyója van még.

Asztalrajzoló
Kirajzolja az asztalt, ill. az azon lévő golyókat. Képes csak egyetlen golyót újrarajzolni. A fizikai modell koordinátáit átszámolja képernyőkoordinátákra

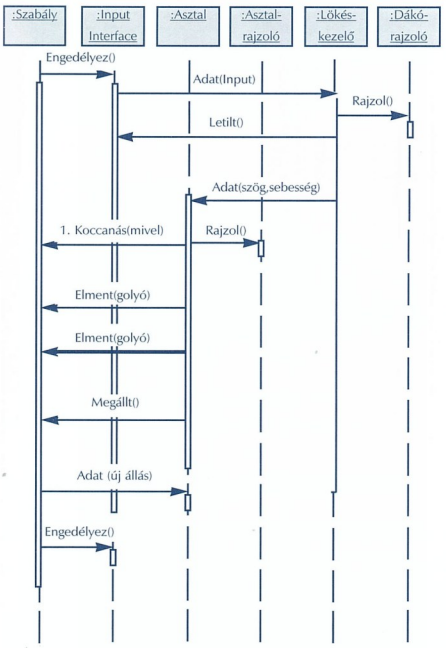
A fenti osztályok és objektumok alapján rajzoljuk fel a statikus struktúradiagramot:
Mit is láthatunk a fenti diagramon? Középen található a Szabályobjektum, amely a program indulásakor azonnal betöltődik. A Menüutasítások, az Input interfész, a Játékállás-kijelző és az Asztal objektumokkal áll kapcsolatban. Ez utóbbi kettőnek üzeneteket küld *(egy-egy esemény bekövetkezésekor hogyan módosul a játék állása, adott lökés hatására hogyan kell mozgatni a golyókat, valamelyiket el kell-e tüntetni stb.)*.
Az asztalon golyók találhatók, amelyek négyfélék lehetnek *(fekete, fehér, csíkos vagy teli)*. A bemeneti interfész kezeli a felhasználói üzeneteket, ő dolgozza fel a billentyű- és egéreseményeket, a szabályokon keresztül ő rajzolja ki a dákót illetve indítja el a golyók mozgását.
Az Input interfész szintén kapcsolatban áll a Menüutasításokkal *(a felhasználó aktiválja az egyes menüpontokat)*, illetve a Lökéskezelővel, amely a játékos utasításainak megfelelően a dákó kezelését végzi, és kirajzoltatja azt a Dákórajzolóval.



A biliárdjáték statikus struktúradiagramja

A statikus struktúra felrajzolása után rátérhetünk a dinamikus nézetekre. Ezek a program folyamatait, az objektumok egymásra hatását, kommunikációját írják le. Lássuk, milyen alapvető folyamatokból épül fel a biliárd:
Először is biztosan van egy inicializáló folyamat, amely a program betöltődésekor indul. Ezen kívül két másik folyamatot kell definiálnunk: a lökést, és a menüpontok aktiválását. Ezekkel a rendszer teljes működése leírható.
Példaként a lökés szekvenciadiagramját mutatom be, a többi ehhez hasonlóan egyszerűen megadható.

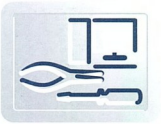
Lássuk, mi történik a fenti ábra alapján. A Szabályobjektum indulása után egy engedélyező üzenetet küld az Input Interfésznek, s az ennek következtében a bemeneti adatokat elküldi a Lökéskezelőnek. A Lökéskezelő rajzolásra utasítja a Dákórajzolót, majd letiltja a bemeneti csatornákat (amíg a lökés nem ért véget, vagyis a golyó nem állt meg, a felhasználó nem adhat újabb utasításokat a rendszernek). A golyó elmozdulásának irányát és sebességét



A lökés szekvenciadiagramja

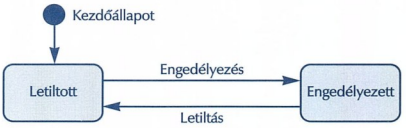
megkapja az Asztal, amely az Asztalrajzoló segítségével kirajzolja az aktuális állapotot a képernyőre. Ha a golyó egy másik golyóval koccan, a Szabályobjektum kap egy üzenetet erről *(az üzenet paramétere az a golyó, amellyel ütközött a mozgó golyó; a többszörös ütközéseket visszavezetjük több kétgolyós ütközésre)*. Hasonlóan tájékoztatjuk a Szabályt arról is, ha egy golyó elment *(beleesett egy lyukba)*, vagy megállt.
Mindeközben a Szabályobjektum folyamatosan értesíti az Asztalt az aktuális állásról, a megjelenítés céljából.
A golyómozgások leállásával a Szabály újra engedélyezi az Input interfészt, ezzel az egész fenti folyamat kezdődhet elölről.

Ezen kívül fontos lehet felírni az egyes objektumok állapotait és állapotátmeneteit is. Erre szolgál az állapotátmeneti diagram *(state-chart)*, mely rendkívül fontos a rendszer eseményorientált viselkedésének feltárásánál és leírásánál. A diagram csomópontjai az adott objektum állapotai, irányított élei az átmeneteket jelölik.
Tekintsük például az Input interfész objektumot. Milyen állapotai lehetnek? A kezdeti inicializálás után alapértelmezésben tiltva van *(nem fogadunk felhasználói utasításokat)*, majd az engedélyezés után válik engedélyezetté, és újabb tiltás után ismét letiltódik. Az, hogy kitől érkeznek ezek az üzenetek, az állapotátmenetek szempontjából nem lényeges.
Hasonlóképp rajzolhatjuk fel a többi objektum *(Asztal, Szabály stb.)* állapotátmeneteit is, ezt az Olvasóra bízom.



Összegzés

Lássuk, hova is jutottunk el idáig. A követelmények tisztázása, a megrendelő igényeinek rögzítése után szükséges a feltételek pontos, szabatos megfogalmazása, a rögzített követelmények elemzése, a rendszer analízise.



Az Input interfész állapotátmenet-diagramja

Elsőként a rendszerrel kapcsolatos szavakat, fogalmakat definiáljuk, pontosítjuk, konkretizáljuk. Ismeretes ugyanis, hogy mi, földi halandók hányféleképp tudjuk érteni ugyanazokat a dolgokat. A követelmények meghatározása hasonló célokból szükséges, muszáj egyértelművé tenni, mit is várunk *(mit vár a megrendelő)* a megvalósítandó rendszertől, mikor nevezhetjük „kész"-nek a terméket.
Ezek után kerülhet sor a használati eset diagram *(use case)* felrajzolására, amely a rendszer szereplőit és feladatait írja le. Ez már konkrét, precíz fejlesztői dokumentáció, és egész további munkánk alapját jelenti.
A statikus struktúradiagram a következő lépés, amely közelebb visz bennünket a megoldáshoz. Hogyan is fejleszthetnénk szoftvert, ha nem ismerjük annak statikus felépítését, szerkezetét? Hasonlóan fontos a különböző folyamatok részletes felírása különböző szekvencia-diagramokkal, majd pedig egy-egy objektumra koncentrálva annak állapotátmenetidiagramjával. Ez utóbbiak már a dinamikus viselkedést írják le, amely szintén elengedhetetlen feltétele a későbbi megvalósításoknak.
Jól látható, hogy mindezidáig egyetlen sor kódot sem írtunk. Tapasztalataim szerint a fejlesztők jelentős hányada sokkal jobban szereti a „vágjunk a közepébe" módszert, mint bajlódni a sok-sok dokumentációval. Pedig – mint azt példánkban is láthatjuk – a dokumentálás hosszú távon igen gyümölcsöző befektetés. Egyetlen sor kódot sem írtunk még le, sőt még arra sem utaltunk, hogy milyen technológiák segítségével, milyen nyelven végezzük majd a fejlesztési munkát, és máris mennyi mindent tudunk jövőbeli rendszerünkről!

És mennyi mindent megtudunk még, ha folytatjuk ezt a folyamatot… A következő hónap(ok)ban végigvezetem a rendszer teljes fejlesztési ciklusát, folytatva a megfelelő dokumentumok készítésével, és a programkódok, a tulajdonképpeni szoftver megírásával.
Addig is mindenkinek jó szórakozást kívánok saját rendszeréhez!

Molnár Ágnes

NetAcademia MesterQrzusok

*2003. tavaszi program **

MesterQrzus rendezvényünkön bepillantást nyerhetnek a NetAcademia oktatási tevékenységébe. Jöjjön el, legyen jelen legújabb cikkeink, workshopjaink, előadásaink születésénél!
A tavaszi előadások vendégelőadói értékes ajándékokkal is meglepik a kedves résztvevőket!

A MesterQrzus-kiskonferenciák ára: 15.000 Ft+ÁFA alkalmanként. A tech.net magazin előfizetőinek kedvezményes részvételt biztosítunk!

Keresse a tech.net magazinban a kedvezményes részvételre jogosító jegyet!

2003. január 30. 13:45 – 18:00

13:35 – 14:00 Regisztráció
14:00 – 15:00 Az Active Directory tartalmának tömeges módosítása

Néhány elterjedt szabvány ismeretének birtokában igazán varázslatos módosításokat végezhetünk az Active Directoryban. Ennyit kell tudni: LDAP, LDIF és X.500. Ezen az előadáson szóba kerül minden, ami használható export/import scriptek írásához feltétlenül szükséges.

(Fóti Marcell)

(Kapcsolódó tanfolyamaink: AD mélyvíz Workshop)

15:00 – 16:00 Adatbázisalapú webalakalmazások hackelése (SQL Injection)

Napjainkban egyre több webhely használ SQL Servert az összegyűjtött adatok tárolására, rendelések regisztrációjára. Teljesen normális, hibátlan alkalmazások is biztonsági rést jelentenek, ha a beviteli mezőket „megfelelően" töltjük ki. Ismerkedjünk meg az SQL Injection hackertechnikával, amíg nem késő!

(Fóti Marcell)

(Kapcsolódó tanfolyamaink: SQL Workshop, 2072 – SQL Server üzemeltetése, 2073 – SQL Server programozása)

16:00 – 16:20 Kávészünet
16:20 – 17:20 A csoportos házirend (Group Policy) rejtelmei

Csoportos házirendek vegyes környezetben. Miért más a csoportos házirend terminal services kiszolgálón? Hogyan lehet saját gyártású házirendet készíteni?

(Dorner Csilla)

(Kapcsolódó tanfolyamaink: A Windows .NET Server tanfolyamsorozat)

Gyakorlat: Címtármódosítás LDIF-fájllal

2003. március 27. 13:45 – 18:00

13:35 – 14:00 Regisztráció
14:00 – 15:00 Bevezetés a hálózati forgalom elemzésébe

Minél összetettebb egy vállalati hálózat, minél többféle szolgáltatást futtatunk, annál furcsább hibajelenségeknek lehetünk időnként tanúi. A hálózati forgalom elemzésével olyan hibák felderítésére is lehetőség nyílik, amelyekről semmit sem mond az Eseménynapló!

(Fóti Marcell)

(Kapcsolódó tanfolyamaink: NetMon Workshop, TCP/IP Workshop)

****A NetAcademia Kft. a programváltoztatás jogát fenntartja!***

15:00 – 16:00 A nyílt kulcsú technológia építőkockái

A digitális aláírási törvény megjelenésével egyre több vállalat fordul érdeklődéssel a nyílt kulcsú technológiák felé. Mi kell ahhoz, hogy elérhetővé váljon számunkra a PKI adta sok-sok lehetőség?

(Vendégelőadó: NetLock Kft.)

(Kapcsolódó tanfolyamaink: PKI Workshop)

16:00 – 16:20 Kávészünet
16:20 – 17:20 Nyílt kulcsú architektúra a Windowsban

A PKI felhasználási területei a titkosított hálózati kommunikációtól a digitális aláíráson keresztül a felhasználó-azonosításig terjednek. Kiszolgálóoldalon többek között a RAS, VPN, IPSec, és IIS, felhasználói oldalon a Smart Card Logon, dokumentumok aláírása, S/MIME a témaválaszték. A műsorváltoztatás jogát az újonnan megjelenő szolgáltatások miatt fenntartjuk!

(Fülöp Miklós)

(Kapcsolódó tanfolyamaink: 2150 – Windowsos hálózatok biztonsága, 2159, ISA Server)

Gyakorlat: *Felhasználói és kiszolgálóoldali PKI-gyakorlatok*

Ajándék: Hiteles NetLock tanúsítványpár elektronikus aláíráshoz és titkosításhoz. A kulcstároló eszközök a helyszínen megvásárolhatók.

2002. május 29. 13:45 – 18:00

13:35 – 14:00 Regisztráció
14:00 – 15:00 Az elektronikus levelezés védelmének lehetőségei Exchange Serveren

Az elektronikus levelezés áldás és átok egyben. A veszélyes leveleket nem szabad a felhasználókig eljuttatni. Ennek eszköze lehet az Outlook-ügyfelekhez készített központi korlátozóeszköz, vagy egy víruskereső.

(Fóti Marcell)

(Kapcsolódó tanfolyamaink: 1572 – Exchange Server üzemeltetése)

15:00 – 16:00 SOAP (Előadó: Soczó Zsolt)

A webszolgáltatások alapjait a SOAP-szabvány rakta le. Az előadásban áttekintjük a SOAP történelmi gyökerét, és az XML-technológiákon keresztül megnézzük gyakorlati felhasználását is.

(Soczó Zsolt)

(Kapcsolódó tanfolyamaink: .NET fejlesztési tanfolyamok)

16:00 – 16:20 Kávészünet
16:20 – 17:20 A Windows rendszerek támadható, és védelmi felületei

A vírusvédelem egyik faktora nyilván a jó, friss víruskereső program. A másik faktor az okos felhasználó, illetve az okos rendszergazda. Ebben az előadásban megismerkedünk a vírusok által használt támadási trükkökkel, így soha többé nem fogunk Administratorként levelezni…

(Vendégelőadó: VirusBuster Kft.)

(Kapcsolódó tanfolyamaink: 2150 – Windowsos hálózatok biztonsága, 2159, ISA Server)

Gyakorlat: *Felhasználói és kiszolgálóoldali PKI-gyakorlatok*

Ajándék: Virus Buster Personal Edition, 1 éves előfizetéssel

PKI-workshop

A NetACADEMIA és a NETLOCK Az Első Hitelesítés Szolgáltató
közös szervezésében

Szerezzen átfogó tudást az elektronikus aláírás és titkosítás vállalati szintű alkalmazásában

Technikai blokk:

- Mi az a kriptográfia?
- Hogyan tudunk nyílt hálózaton kommunikálni úgy, hogy azonosítani tudjuk kommunikációs partnerünket?
- Meg tudunk bizonyosodni üzeneteink **sértetlenségéről?**
- Hogyan működik a **titkosítás**? Mi a **kulcspár**, a **tanúsítvány**, mi a szerepe az intelligens kulcstároló **eszközöknek**?

További témakörök:

- szimmetrikus algoritmusok, nyílt kulcsú **RSA**, hash, a **X.509** tanúsítványok szerepe és **használata**,
- hitelesítésszolgáltató, a hierarchia felépítése,
- kulcstároló eszközök: **intelligens kártya** és **USB Token**,
- bejelentkezés tanúsítvánnyal (Single Sign On), SSL, S/MIME, HTTPS,
- **virtuális magánhálózatok**.

Jogi blokk:

- az elektronikus aláírási törvény célja, következményei, a végrehajtási rendeletek.
- Jogkövetkezmények, és a szükséges feltételek megléte.

Ajándék!


USB-token

vagy


Smart Card olvasó

+

és kártya

+NetLock C-osztályú tanúsítvány

Hallgatóink a tanfolyamon használt kriptográfiai eszközöket a tanfolyam után megtarthatják, és - a mellékelt NetLock tanúsítványokkal - hiteles elektronikus aláírást és nyílt kulcsú titkosítást használhatnak!

A tanfolyam időpontjai	Hossz [nap]	Bővebb információ és jelentkezés	Ár [Ft]
2003. január 14.	**2**	**http://www.netacademia.net/workshop/PKI**	**140,000**
2003. április 3.	**2**	**http://www.netacademia.net/workshop/PKI**	**140,000**

A részletekről érdeklődjön a 06/1/472-1214-es telefonszámon vagy az info@netacademia.net e-mail címen!

A tanfolyam helyszíne: Budapest 1062, Andrássy út 62.

Microsoft Project 2002 – hogy a tervekből valóság legyen!

A projektirányítás számos szervezet sikerében és bukásában játszik fontos szerepet. Egy hatékony tervezési eszköz birtokában csökkenthetők a költségek és javítható a termelékenység, ami nyereségnövekedést eredményez. A Microsoft **Project 2002 Standard** minden eddiginél egyszerűbbé teszi az ütemezések és erőforrások kezelését, a projekt állapotának közlését és a projekt adatainak kimutatását.

Cégek, nagyvállalatok részére, a csoportmunkát teljes mértékben támogató **Project 2002 Professional** kínálja a Microsoft, mely segítségével és a központi nyilvántartást és kiszolgálást biztosító **Project 2002 Server** termékkel együttműködve teljeskörű nagyvállalati, projektirányítási, tervezési és döntéshozási feladatok végezhetők el, és annak adatai bármikor, bárhonnan hozzáférhetőek, publikálhatóak.

Ismerje meg közelebbről tanfolyamainkon!

Microsoft Project 2002 felhasználói kurzusok

Az érdeklődők **kétnapos, intenzív** tanfolyamok keretében ismerkedhetnek meg a Microsoft Project 2002-es verzióinak használatával.

Rugalmas időbeosztás

Megpróbáltuk figyelembe venni, hogy a cégek dolgozóikat nem tudják nélkülözni hosszabb időre, ezért a tanfolyamokat **intenzív, egész napos formában** hirdettük meg, délelőtt 9.00 és 16.00 óra között. A tanfolyamok díja az oktatás és a tananyagok mellett az étkezést is tartalmazza a kurzus napjaira.

Project 2002 Professional és Server – testreszabott oktatások

Oktatóközpontunkon kívül, a megrendelő telephelyén, kihelyezett formában (vidéken is) is vállaljuk az adott cégprojektekhez kapcsolódva tanfolyamok vagy konzultációk megtartását és lebonyolítását, különös tekintettel az **összetett, Project 2002 Professional és Project 2002 Server használatával és bevezetésével** kapcsolatos témákban. Ezt követően igény esetén további **utólagos támogatást** és szaktanácsadást is nyújtunk.

Menjen biztosra!

Tervezze üzleti folyamatait Microsoft Project 2002-vel!

Minőség, egyéneknek kedvező konstrukciók, cégeknek partneri kedvezmények!

A képzésekkel, szakmai kérdésekkel kapcsolatosan kérjük keresse értékesítési vezetőnket, Projekt oktatónkat, Lovász Attilát a 203-0304/3040 mellékű telefonszámon.

Microsoft CERTIFIED Technical Education Center

SZÁMALK TOVÁBBKÉPZÉS

SZÁMALK

✆ 203-0304/3040 www.szamalk.hu/tisza 1115 Budapest, Etele út 68.