Biztonságos és költséghatékony kommunikáció

Dr. Keszthelyi András

Óbudai Egyetem, Keleti Károly Gazdasági Kar
Szervezési és Vezetési Intézet
keszthelyi.andras@kgk.uni-obuda.hu

1 Történeti áttekintés

Az internet történetének kezdetén – amikor még nem nevezték Internetnek a Hálózatot – a biztonság nem volt kérdés és nem volt kérdéses.

A világot mára teljesen átszövő hálózat 1969 decemberében kezdődött mindössze négy számítógép között. A robbanásszerű fejlődést – mint oly sokszor az életben – néhány kudarc váltotta ki.

Az MIT, a Bell Laboratórium és a General Electric a hatvanas évek végén belekezdett egy nagyszabású kutatási programba, amelynek során egy többfelhasználós, többfeladatos operációs rendszert (MULTICS) szerettek volna írni. Különféle nehézségek végül a program feladására kényszerítették a kutatókat.

Bár elméleti megoldásait még ma is egyetemen tanítják, a MULTICS projekt a gyakorlatban nem érte el a várt sikert, ezért a Bell Laboratórium kilépett belőle, ami egyebek mellett azzal járt, hogy a program egyik résztvevője, Ken Thompson, ekkor saját szakállára megírta a MULTICS egyfelhasználós változatát egy használaton kívül lévő PDP-7 számítógépen. Az új operációs rendszer kiherélt változata volt a MULTICSnak, erre utal szarkasztikus elnevezése is, amelyet Brian Kernighan ragasztott rá: eunuch multics, röviden UNICS (Uniplexed Information and Computing Service). Bármennyire földszintes is volt Ken Thompson operációs rendszere, a MULTICS-szal szemben volt egy behozhatatlan előnye: működött. 1969-et írtak ekkor.

Ez a nem elhanyagolható tulajdonság késztette arra Dennis M. Ritchiet, hogy egész osztályával együtt csatlakozzon a UNIX (közben megváltoztatták a helyesírást) fejlesztéséhez. Az így megerősített csapat nekilátott, hogy elkészítse az operációs rendszer PDP-11 gépen futó változatát. Az ős-unix assembly nyelven készült, így újra kellett volna írni az egészet, valahányszor egy új, más típusú gépre szerették volna adoptálni. Mivel operációs rendszert assembly nyelven írni nem gyerekjáték, ezért Dennis Ritchie megalkotta a C nyelvet, és egy fordítóprogramot is írt hozzá a PDP-11 gépre. Ken Thompsonnal közösen újraírták a Unixot ezen a tökéletesített nyelven. A C nyelv elég magas szintű

ahhoz, hogy a hardverkülönbségeket elfedje a programozó elől, ugyanakkor azonban elég alacsony szintű ahhoz, hogy roppant hatékony kódot lehessen vele fejleszteni, akár operációs rendszert is.

A másik kudarc az volt, hogy nem lehetett áruba bocsájtani az új operációs rendszert. A Bell Laboratórium anyavállalata, az AT&T akkoriban ki volt tiltva a számítógépek piacáról, így a Unixot nem árulhatta pénzért. Semmi sem indokolta, hogy az operációs rendszert – forráskódjával együtt – ne adják oda bármelyik egyetemnek. Így is tettek. A másik körülmény, hogy a Unix PDP-11-en futott. Az egyetemeken akkoriban szinte nem is volt más gép, mint PDP-11. És ezen a ponton a DEC cég is alaposan besegített a Unix világsikerébe: a PDP-11 operációs rendszere borzalmas volt. Ezek után könnyű kitalálni, hogy mi történt: egyetemek százain kezdték használni a Unixot, a forráskód birtokában pedig megindult a véget nem érő buherálás.

A számtalan Unix-buheráló egyetem közül kiemelkedik a Berkeley Egyetem. ők számos ponton javítottak az AT&T-től valaha ingyen kapott Unixon, gyorsabb fájlrendszert írtak, beépítették a hálózatkezelést (ami TCP/IP néven legalább olyan elterjedt a hálózati protokollok között mindmáig, mint a C a programozási nyelvek terén), valamit számos segédprogramot (csh, vi, fordítóprogramok stb.).

Már az 1970-es években működött az email, az FTP, a távoli gépre történő bejelentkezés. Mivel ekkoriban a hálózat, egyáltalán: a számítógép kevés szakember tudományos kutatásának tárgya volt. Ők „igazi programozók”[1] voltak, rosszindulatú kíváncsiskodók, digitális betörők és jámbor felhasználók még a képzelet síkján sem léteztek. Ennélfogva az összes korai kommunikációs protokoll nyílt szöveg alapú: az összes adat, ideértve a bejelentkezési neveket és a jelszavakat is, nyílt szöveg formájában továbbítódnak a hálózaton, így igen könnyű ezeket lehallgatni (SMTP, POP3, IMAP, telnet, rlogin, ftp), l. 1. ábra.

```
$ cat tcp.dat
81.183.16.75.33413 > 193.225.224.240.21:
0x0000   4510 003e 5972 4000 4006 dc63 51b7 104b        E..>Yr@.@..cQ..K
0x0010   c1e1 e0f0 8285 0015 521a ceeb 5dc8 c973        ........R...]..s
0x0020   8018 16b0 6dab 0000 0101 080a 002a 97ef        ....m........*..
0x0030   162c 46dd 5553 4552 206b 6561 0d0a             ..F.USER.kea..
81.183.16.75.33413 > 193.225.224.240.21:
0x0000   4510 0045 5974 4000 4006 dc5a 51b7 104b        E..EYt@.@..ZQ..K
0x0010   c1e1 e0f0 8285 0015 521a cef5 5dc8 c993        ........R...]...
0x0020   8018 16b0 c9f5 0000 0101 080a 002a 996e        .............*.n
0x0030   162c 47de 5041 5353 2057 6172 646c 6539        ..G.PASS.A_passw
0x0040   726f 630d                                      ord.
```

1. ábra
FTP bejelentkezés adatai

[1] L. pl. http://www.caesar.elte.hu/progmat/kultura/humor/igazi.html

Aztán a helyzet megváltozott. Az 1990-es évek elejére a személyi számítógép és a hálózati hozzáférés általánossá vált a világ szerencsésebb részén. 1990-ben indult a svájci CERN-ben, Tim Berners-Lee vezetésével a web-projekt, és lassanként az üzleti élet is fölfedezte az új eszközt. Ahogy az internet általános adatátviteli eszköz és kommunikációs lehetőséggé vált, úgy erősödött az igény a biztonságos kommunikációra.

1991-ben Phil Zimmermann elkészítette a PGP titkosító programot, amely nem más, mint az 1976-ban publikált RSA-algoritmus[2] gyakorlati megvalósítása. Az évtized közepére megjelentek a legfontosabb eszközök, így az SSH (1995) és az SSL (1996).

2 Kommunikációs helyzetek

A kommunikáció biztonsága napjaink egyik legfontosabb kérdése. Számos típusos kommunikációs helyzet és probléma van, ennélfogva számos különféle megoldás is létezik. Ezek közül veszünk szemügyre néhányat illusztrációként, messze nem a teljesség igényével.

Általánosan megfogalmazva az üzleti életben a leggyakoribb kommunikációs probléma számítógépek és hálózati szolgáltatások távoli elérése. Hogyan képes egy munkatárs

- emilt küldeni saját vállalatának hálózatán keresztül,
- letölteni ugyanonnan saját emiljeit,
- elérni a vállalati intranetet,
- különféle vállalati alkalmazásokat futtatni, akár a vállalati kiszolgálókon
- stb.

biztonságosan távolról, akár az ellenérdekű fél helyi hálózatából indulva? Nemcsak a problémás helyzetek lehetnek sokfélék, hanem az ezek kezelésére alkalmas eszközök is különfélék, mind árban, mind hatékonyságban, mind járulékos szolgáltatásokban és biztonsági szintben.

---

[2] L. pl. Ködmön József: Kriptográfia. Computerbooks Könyvkiadó, Budapest, 2000. Elektronikus változat pl. http://mail.de-efk.hu/~csajzo/PGP/Kodmon%20-%20KRIPTOGRAFIA_KONYV_V1_0.pdf

3 Szabad szoftverek

A Szabad Szoftver Alapítvány megfogalmazása szerint

„Szabad szoftver alatt értünk minden számítógépes programot és dokumentációt, amely kielégíti az alábbi feltételeket:

- A szoftver bármilyen célra felhasználható.
- Lehetőség van a szoftver működésének szabad tanulmányozására és módosítására.
- Szabadon terjeszthető, továbbadható.
- Lehetőség van a szoftver továbbfejlesztésére és a fejlesztés közreadására.
- A szoftver tanulmányozásának, módosításának, illetve továbbfejlesztésének előfeltétele a forráskód elérhetősége."[3]

A forráskód birtoklása a legbiztosabb eszköze és bizonysága a biztonságnak. Ennek vizsgálatával lehet meggyőződni arról, hogy a használt program nem tartalmaz hátsó kapukat és egyéb biztonsági kockázatokat, sem szándékosan beépítetteket, sem pedig véletlen programozói hiba következményeképpen.

Mivel a szabad szoftver többnyire ingyenes is, ezért a szabad szoftverre alapozott megoldások igen költséghatékonyak lehetnek.

4 SSH és a számítógép kapui (portjai)

Az SSH (Secure Shell, biztonságos bejelentkezés) a számítógépközi adatátvitel biztonságát lényegesen növelő zseniális eszköz, amely három fontos biztonsági szolgáltatást biztosít számunkra:

- **Hitelesítés.** Két különböző módszer, a nyilvános kulcs és a jelszó alapú hitelesítés kombinálható. A nyilvános kulcs (és annak titkos párja) olyan dolog, amelyet a felhasználó birtokol, míg a jelszó olyasmi, amit a felhasználó tud, hogy digitálisan bizonyítsa (önmagával való) azonosságát.
- **Titkosítás.** Az SSH a teljes adatforgalmat titkosítja ipariági szabványnak számító eljárásokkal (mint pl. a Blowfish vagy az AES).

---

3 L. pl. http://fsf.hu/about/mivel-foglalkozunk/mi-a-szabad-szoftver/

- **Adatépség.** Az SSH digitális aláírása garantálja, hogy a nem biztonságos hálózaton keresztül történő adatátvitel során az adatok nem változtak meg.

Az SSH lehetővé teszi a felhasználók számára, hogy

- távoli számítógépre bejelentkezhessenek és ott programokat futtassanak;
- fájlokat másolhassanak biztonságos módon távoli számítógépek között (scp);
- távoli hálózati szolgáltatásokat érhessenek el biztonságos módon, mintha VPN-szolgáltatást vennének igénybe (kaputovábbítás, port forwarding).

Mi is az a számítógépes kapu, vagy port? Nem más, mint egy virtuális kapu, amelyet programok ideiglenes fájlok használata nélküli, közvetlen adatcserére használhatnak. Ezeken keresztül kapcsolódik össze a bejövő adatáram és az azt feldolgozó program, pl. hagyományosan a 25-ös kapu használatos az elektronikus levelek továbbításra (SMPT), vagy a 110-es a bejövő levelek letöltésére (POP3), vagy a 3389-es a Windows távoli asztal elérésre.

A kapuk tehát leginkább egy hivatalhoz hasonlíthatók, aholis az egyes hivatali ügyeket a megfelelő ablakoknál lehet, illetve kell intézni.

5 Tipikus problémák ingyenes megoldása

Tegyük fel – kiindulásként, matematikus módra –, hogy a munkatársak laptopjain és/vagy otthoni gépein Linux operációs rendszer van, amely ugyancsak szabad szoftver. Természetesen arra az esetre is van megoldás, ha a szóban forgó gépeken Windows fut, feltéve, hogy ez legalább XP.

Az általános technikai követelmények és körülmények a következők: a munkatársak tehát Linuxot futtatnak, X-Window grafikus rendszerrel, a vállalati hálózatot tűzfal védi, amely a külvilág felől csak az SSH kapcsolatokat engedi be egy bizonyos gépre, az SSH-kiszolgálóra. Ez utóbbinak nem feltétlenül muszáj különálló gépnek lennie, futtathat egyéb szolgáltatásokat is – kérdés persze, hogy biztonsági megfontolások alapján és a teljesítménykorlátokat is figyelembe véve ez mennyire szerencsés.

5.1 Levélküldés a vállalati hálózaton keresztül

Mivel a levélszemét évek óta komoly probléma – az összes elektronikus levél többsége spam –, az elektronikus levelek küldését végző kiszolgáló gépeket (SMTP server) elég szigorú beállításokkal üzemeltetik. Kimenő leveleket normálisan csak a saját belső hálózatba tartozó gépektől fogadnak el, azok IP-címe

alapján (vannak persze más lehetőségek is). Az otthoni számítógépeknek többnyire, a vándorló laptopoknak szinte mindig dinamikus IP-címe van, így az SMTP-kiszolgáló az IP-cím alapján nem tudja, nem tudhatja, hogy „saját" gépről lenne szó, így levélküldést sem fogad el ezektől. Ráadásul, ha egy otthoni gépnek statikus IP-címe volna is, azt egy spammer hamisíthatná, tehát a legjobb megoldás általában véve az, ha a levélküldő kiszolgáló csak és kizárólag a vállalati belső hálózatból fogad el továbbításra leveleket.

A másik probléma az, hogy az SMTP protokoll is nyílt szöveg alapú. Tegyük fel – ahogy a matematikusok mondják –, hogy a vállalat egy munkatársa bizalmas levelet szeretne küldeni a főnökének – egy másik vállalat hálózatából (mondjuk pl. ahol tárgyal). Ez esetben a levél tartalmát is védeni kellene, hiszen ezt a helyi hálózaton, amelyhez a laptopja csatlakozik, a legkönnyebb lehallgatni, ráadásul itt van a legerősebb indíték is minderre.

Mindkét problémára a kaputovábbítás (port forwading) a megoldás.

A dolgozó laptopján a levelezőprogramot (pl. Mennydörgő Madár – Thunderbird) úgy kell beállítani, hogy az a saját gépet (localhost, 127.0.0.1) használja kimenő levelező kiszolgálóként (SMTP server), mondjuk a 2525-ös kapun (feltéve, hogy azt egyéb program nem használja azt). Mivel a dolgozó laptopján nyilván nem üzemel levelezőkiszolgáló, ezért a 2525-ös helyi kaput valahogy össze kellene kötni a vállalati levelezőkiszolgáló SMTP kapujával (alapértelmezett esetben: 25). Az SSH ezt megteszi nekünk alkalmas paraméterezéssel:

SSH -f -N -L 2525:smtp.vallalat.hu:25 smtp.vallalat.hu

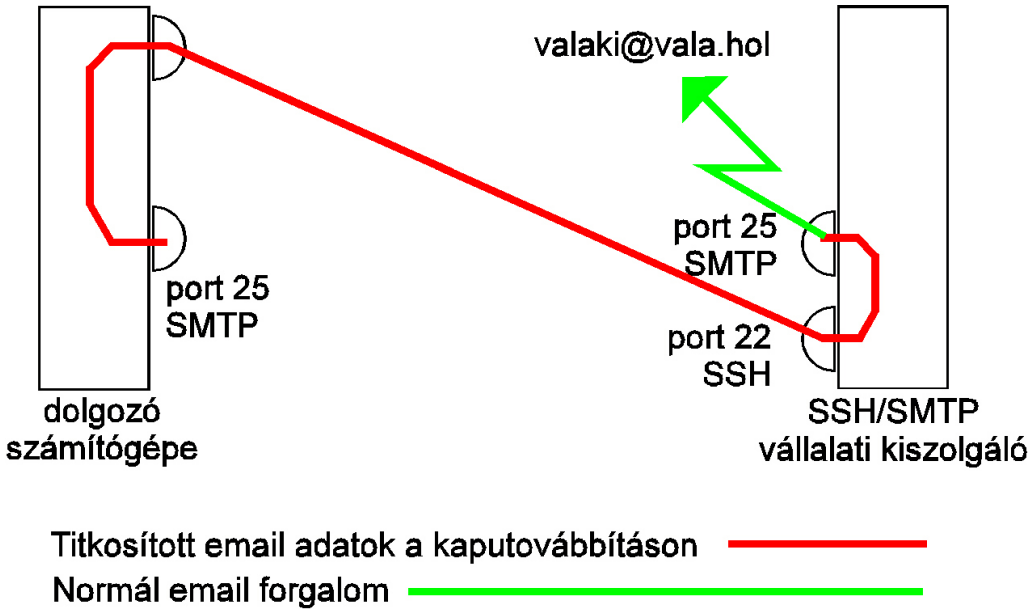
ahol az smtp.vallalat.hu gépnév nyilván a szóban forgó vállalat SMTP kiszolgálójának a neve. Esetünkben további feltétel, hogy a levelezés kiszolgálása mellett SSH-kiszolgálóként is működjön, azaz fusson rajta az SSH démon is. A parancs egyes kapcsolói:

-f: az SSH a háttérben fusson;

-N: ne hajtson végre a távoli gépen semmilyen programot (felhasználói profil, pl. ~/.bash_profile);

-L 2525:smtp.vallalat.hu:25: a helyi 2525-ös számú kapu adatforgalmát az smtp.vallalat.hu nevű gép 25-ös kapujára kell továbbítani (ahol a tényleges SMTP kiszolgáló várja a kimenő elektronikus leveleket);

smtp.vallalat.hu (végén): a kapuátirányítást végző SSH-kiszolgáló esetünkben ugyanaz a gép, mint az SMTP kiszolgáló (2. ábra).



2. ábra
Elektronikus levél küldése SSH kaputovábbításon keresztül

5.2 Elektronikus levelek fogadása POP3 protokollon

A POP3 protokoll használatával könnyen letölthetők a felhasználó beérkezett levelei a kiszolgálóról, bárhol legyen is a felhasználó. Ehhez a kiszolgáló nevét, a felhasználó nevét és jelszavát kell ismerni. A POP3 protokoll használatakor nem merülnek föl a vállalati erőforrások jogosulatlan használatának olyasféle lehetőségei, mint az SMTP protokoll kapcsán (spam), mégsem javasolt a használata megbízhatatlan hálózat esetén.

Nemcsak a letöltött levelek tartalmát lehet lehallgatni, de a felhasználói nevet és a hozzá tartozó jelszót is. Mivel a legtöbb ember nem szeret számos különböző jelszót memorizálni, a lehallgatónak jó esélye van arra, hogy a felhasználó a most megszerzett jelszót használja egyéb, esetleg minden más bejelentkezésénél is, legyen szó akár más vállalati erőforrásokhoz való hozzáférésről, akár a privát szféráról. Ennek veszélyeit nyilván nem lehet túlbecsülni.

A levelek tartalma is nyílt szövegként halad a hálózaton, számos olyan helyzet van, amelyben ez nem javasolt, vagy egyenesen tiltott.

A helyzet a levélküldéssel teljesen azonos módon kezelhető. A dolgozói gépen a levelezőprogramot úgy kell beállítani, mintha a POP3 kiszolgáló a saját gép (localhost, 127.0.0.1) 1110-es portján lenne. Természetesen valójában a helyi gépen semmilyen POP3 kiszolgálás nincs, tehát az SSH-ra vár az a feladat, hogy a helyi 1110-es kaput összekösse a vállalati kiszolgáló POP3 kapujával (alapértelmezett esetben a 110-es):

SSH -f -N -L 1110:pop3.vallalat.hu:110 pop3.vallalat.hu

Itt a pop3.vallalat.hu a cég igazi POP3-as kiszolgálójának a neve, ugyancsak feltételezve, hogy egyben SSH kiszolgálóként is működik. Tekintettel a levelezés sajátosságaira, feltehetően azonos lesz az SMTP kiszolgálóval.

5.3 A vállalati intranet távoli elérése

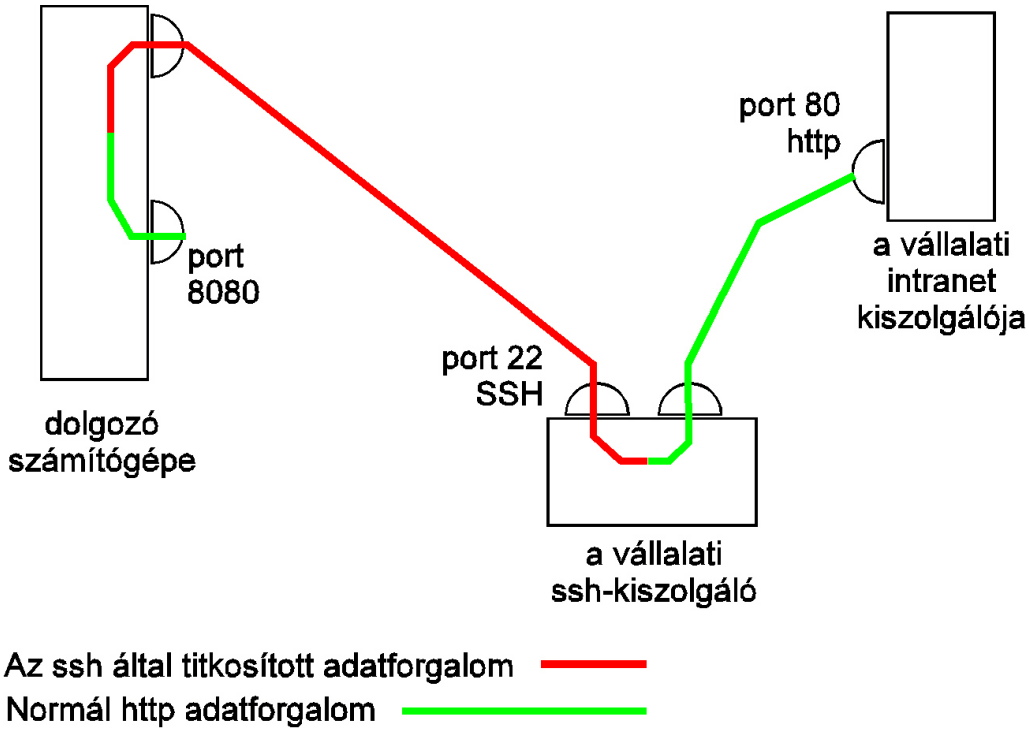
Tegyük fel, hogy munkatársunk üzleti tárgyaláson van valahol a nagyvilágban, és közben szüksége lenne arra, hogy a vállalati intraneten keresgéljen. Mindezt úgy, hogy a helyi hálózat, amelyen keresztül az internetre csatlakozik, nemcsak megbízhatatlan, de annak üzemeltetője kimondottan érdekelt lehet abban, hogy dolgozónk adatforgalmát lehallgassa.

Az intranet nevéből kis következik, az internettel ellentétben, hogy ez nem világháló, hanem „cégháló”, azaz csak a cég megbízhatónak tekintett belső hálózatából lehet elérni, tekintettel arra, hogy a http protokoll ugyancsak nyílt szöveg alapú, és nyilván egy cég sem szeretné a saját belső dolgait az egész nagyvilág orrára kötni.

De mit lehet tenni, ha a dolgozónak az internet valamely távoli pontjáról is el kell érnie a vállalati intranetet? Ismét az SSH kaputovábbítási szolgáltatását hívjuk segítségül, de most még azt is feltesszük, hogy az intranetet kiszolgáló web-szerver nem futtat SSH-kiszolgálót is egyben.

SSH -f -N -L 8080:intranet.vallalat.hu:80 ssh.vallalat.hu

Ahhoz, hogy a dolgozó a vállalati intranetet elérje, a böngészőbe a http://localhost:8080 url-t kell beírja. Ez esetben azonban a két gépnév különbözik, az ssh-kapcsolatot az ssh.vallalat.hu gép fogadja, és ő továbbítja az adatforgalmat az intranetet kiszolgáló webszerver 80-as kapujára (alapértelmezett http kapu). Természetesen a vállalati tűzfalszabályoknak ezt lehetővé kell tenniük. Ez esetben nyílt szövegű adatforgalom csak a vállalati intranetet kiszolgáló webkiszolgáló és a vállalati ssh-kiszolgáló között zajlik, de ez megbízható belső hálózatnak számít. L. 3. ábra.



3. ábra
Az intranet távoli elérése

5.4 Windows-alkalmazások távoli futtatása

A fent bemutatott példákban a távolról elért vállalati kiszolgálók Linux operációs rendszert futtatnak, már csak azért is, mert a tipikus hálózati szolgáltatásokat többnyire linuxos gépek nyújtják. Mit tehetünk azonban akkor, ha a dolgozónak egy Windows operációs rendszerű gépet kellene elérnie távolról, és azon programot futtatnia? A kaputovábbítás ez esetben is használható, egy apró segédprogrammal (rdesktop) kiegészítve, amely távoli Windows gépek elérését teszi lehetővé.

Esetünkben nemcsak az adatforgalom titkosítása fontos, hanem a a helyi Linux-gépről történő programfuttatás a távoli Windows-gépen is érdekes. Ismét feltételezzük, hogy a vállalatnak külön ssh-kiszolgálója van, a kaputovábbítás pedig a megszokott módon történik:

SSH -f -N -L 3389:windowsgep.vallalat.hu:3389 ssh.vallalat.hu

ahol a 3389-es kapu a Windows-asztal távoli elérésének alapértelmezett kapuja. A helyi 3389-es kapu átirányítása után a helyi gépen a dolgozó a rdesktop nevű programot indítja:

rdesktop -f -a 24 -k hu -u dolgozó localhost:3389

Az egyes paraméterek jelentése a következő:

- -a 24: 24 bites színmélység;
- -k hu: magyar billentyűzet-kiszotás;
- -u dolgozó: a bejelentkezéshez szükséges felhasználónév dolgozó;
- -f: teljes képernyős mód.

Ha minden rendben van, akkor a dolgozó helyi linuxos gépének grafikus felületén megjelenik a távoli Windows bejelentkezési képernyője.

5.5 Ha a dolgozó laptopján Windows fut

Az összes fenti példában azt feltételeztük, hogy a dolgozó laptopján vagy otthoni gépén Linux operációs rendszer van. Ha evvel ellentétben valamely Windows változat fut a dolgozói laptopon, a kapuátirányítás ugyanúgy elvégezhető, mindössze egy apró segédprogramra van szükség hozzá. A PuTTY.exe nevű freeware programra lesz szükség, amelynek grafikus felhasználói felületén a szükséges, illetve kívánt kaputovábbításokat egérkattogtatással lehet beállítani. A PuTTY a telnet ill. az ssh implementációja 32 bites Windows operációs rendszerekre. Kifejlesztője és elsődleges karbantartója Simon Tatham. A legutóbbi változat a 0.60 béta verzió. Hivatalos honlapja:

http://www.chiark.greenend.org.uk/~sgtatham/putty/

6 Szoftver és hardver feltételek

Különféle VPN-megoldásokkal ellentétben semmilyen különleges hardver- vagy szoftverelemre nincs szükség. Természetesen az SSH-ra szükség van mind a ügyfél, mind pedig a kiszolgálói oldalon. Több megoldás is létezik mind Linux/Unix, mind pedig Windows esetére.

Az OpenSSH szabad szoftver, erős titkosítással (3DES, Blowfish, AES stb.), kaputovábbítással (titkosított csatorna a hagymányos protokollok számára), erős felhasználói hitelesítéssel. Gyakorlatilag bármely Linux/Unix-szerű operációs rendszeren és Windowson használható. Bővebben l. http://www.opoenssh.com.

Az SSH Tectia az SSH egy kereskedelmi változata mind Linux, mind Windows számára, üzleti célokra. A nem üzleti célú felhasználása szabad. Bővebben l. http://www.ssh.com.

A VanDyke Software csak Windows platformra biztosít kereskedelmi megoldást. A kiszolgáló oldal számára a VShell, az ügyfél oldalán a Secure CRT a program neve. Teljes funkcionalitás mellett harminc napos próbaidő lehetséges. Bővebben l. http:/T/www.vandyke.com.

Irodalom

[1] The Free Software Definition. http://www.fsf.org/licensing/essays/free-sw.html

[2] Dwivedi, Hirmanshu: SSH a gyakorlatban. Módszerek biztonságos hálózati kapcsolatok kialakítására. Kiskapu, 2004.

[3] Gagné, Marcel: Linux-rendszerfelügyelet. Kiskapu, 2002.

[4] Flickenger, Rob: Linux Server Hacks. O'Reilly & Associates, Inc., 2003.

[5] Hagen, Bill von – Jones, Brian K.: Linux Server Hacks, Volume Two. O'Reilly Media, Inc., 2006.

[6] www.openSSH.com

[7] www.SSH.com

[8] www.vandyke.com