

FLOSSzine

Free/Libre/Open Source Software fanzine

2008. július

I. évfolyam 1. szám



FRISS!!!

ÚJDONSÁG!!!

**SZABADNA,
KÉREM?**

SZENZÁCIÓ!!!

LITE:

Vezércikk - 3. old.

Szabadna, kérem? - 4. old.

A nyílt forrású és szabad szoftverek helyzete az Európai Unióban - 8. old.

Csoportmunka Open Source környezetben (eGroupWare) - 14. old.

A Frugalware Linux múltja, jelene és jövője - 19. old.

Jake2 - a függetlenség szobra - 22. old.

Interjú Nagy Róberttel (OpenBSD) - 25. old.

Interjú Micskó Gáborral (Hungarian Unix Portal) - 27. old.

PRO:

Integrált Fejlesztői Környezetek Linux alatt - 29. old.

Hello World! (Programozás Linux környezetben) - 31. old.

Linksys NSLU2, az energiatakarékos mindenek - 35. old.

A GNU Screen terminálkezelő program - 37. old.

FreeDOS - 41. old.

ModSecurity - 44. old.

Védtelen véletlenek (Debian/OpenSSL bug) - 48. old.

Főszerkesztő, alapító:

Horváth Örs Apor - rendszermérnök - Budapest

Szerzők:

Jankovich Oszkár - újságíró - Budapest

Kovács Zsolt - informatikus - Debrecen

Lőcsei Gábor - unix rendszergazda - Budapest

Medve Zoltán - linux rendszergazda - Szeged

Miklós Csaba - szoftverfejlesztő - Pécs

Pfeiffer Szilárd - szoftvermérnök - Mosonmagyaróvár

Sütő János - rendszermérnök - Budapest

Szőke József - informatikus - Mikepércs

Vajna Miklós - egyetemista, rendszerfejlesztő - Budapest

Vomberg István - szoftvermérnök - Budapest

A FLOSSzine elérhetőségei:

E-mail: info@flosszine.org

Web: www.FLOSSzine.org / www.FLOSSzine.hu

IRC: #FLOSSzine ; #FLOSSzine.hu ; #FLOSSzine.org (irc.freenode.net)

Köszönet az *FSF.hu* Alapítványnak a tárhelyért!

Az e-fanzine elkészítéséhez kizárólag nyílt forráskódú, szabad és ingyenes szoftvereket használunk. A lap teljes tartalma saját szerzemény, nem átvett és/vagy idegen nyelvből fordított. A cikkekért a szerzői jogdíj a szerzőket illeti, minden további jog fentartva az alapítónak.

Tisztelt Olvasó!

Elsősorban köszönet, hogy letöltötte szerény kis elektronikus magazinunkat, reményeink szerint talál benne az Ön számára érdekes olvasnivalót.

Bízunk benne, hogy a következőt is letölti hasonló céllal.

Miért?

Mert igény van rá. Szerencsére egyre nagyobb. Percről-percre több magyar szoftverfelhasználó kerül kapcsolatba nyílt forráskódú és/vagy ingyenes szoftverekkel, de Magyarországon jelen pillanatban (tudomásunk szerint) egyetlen hasonló célközönségnek, hasonló céllal író, nyomtatott vagy elektronikus úton terjesztett újság sincs (főleg ingyenesen hozzáférhetőről nem tudunk).



Ha úgy tetszik, magunkért tesszük, és csak remélni merjük, hogy másnak is lehetünk a virtuális asztalára néhány kis könnyed és (a lap életciklusa során remélhetőleg) kevésbé könnyű (de annál hasznosabb), olvasmányt. Nem szeretnénk (mert nem is tudnánk) etalon lenni, csupán fokozottabban szeretnénk a figyelmet felhívni a másik oldalra, nevezetesen, hogy van élet a kereskedelmi forgalomban lévő, gyakran drága, ugyanakkor

kifogásolható minőségű, zárt forráskódú szoftvereken túl is. Nem titkolt célunk az is, hogy a magyar FLOSS közösség jelességeit megismertessük Olvasóinkkal, magyar FLOSS projektek (operációs rendszerek, programok, honlapok, fordítócsapatok, alkotók stb.) bemutatói, bemutatkozóai által.

Mennyiért?

Ingyen. Pontosabban annyiért juthatunk hozzá, mint a szabad és nyílt forráskódú szoftverekhez. Szabadidőben, szabadon (akkor foglalkozunk vele, amikor ráérünk és kedvünk is van hozzá).

Semmi sincs ingyen?

Ez így igaz! Már eddig is rengeteg időnk, munkánk, energiánk és esetenként pénzünk van ebben a szerény kis lapban. De mi hobbiból is a kitartóbbak közül valóak vagyunk... ;-)

Hivatkozások:

<http://www.flosszine.org>

<http://www.flosszine.hu>

<http://en.wikipedia.org/wiki/FLOSS>

<http://en.wikipedia.org/wiki/Fanzine>

<http://www.linuxvilag.hu>

Miért pont most?

Mert megközelítőleg egy éve szűnt meg az egyetlen, profilját tekintve tán hasonló, nyomtatott és CD/DVD melléklet nélkül sem kevés pénzért, el-/kiadott magazin, a *Linuxvilág*, mely előtt, mindentől függetlenül, ezúttal is tisztelgünk. Egy év hosszú idő és mi -mint reményeink szerint az olvasóink többsége- olyan emberek vagyunk akik szívesen olvasunk hasonló „irodalmat”.

Kinek?

Minden olyan kedves Olvasónak, akit még nem sikerült teljesen elvakítani a kényszerszerűen (pl. lásd a legutóbbi OKM-Microsoft esetet), drágán, forráskód nélkül eladott szoftverrengeteggel.

Meddig?

Amíg erőnk, energiánk és megfelelő kapacitásunk van rá.

Itt hívnám fel a figyelmet, hogy mindenki, aki hasonló témakörben publikált volna, de például egy M\$ érdekeltségű lapban nem jelenhetett meg (az anyaga), most bátran küldje el hozzánk az írását, mert igyekszünk a magyar FLOSS közösség fanzine-jának megfelelően egy szabad (de a megszokottnál mindenképp szabadabb) „fórum” lenni, mint amire eddig lehetősége nyílt az írói, szerkesztői (FLOSS „fanatikusi”) vénával rendelkező olvasóknak.

Lehetne jobb is?

Kétségtelenül.

Tartalmasabb, szebb, jobb is lehetne, ezt mi is tudjuk és törekszünk is rá, de mivel mindannyian (elsősorban) főállásban dolgozunk (többnyire az informatika területén), csak a kevés szabadidőnkben tudunk ezzel foglalkozni, így ez időnként a minőség rovására is mehet. Azt megtudjuk fogadni, hogy igyekszünk a menet közben felgyűlt tapasztalatokat minőségibb formában eljuttatni az Olvasóinkhoz.

E-fanzine? Nyomtatott fanzine? Reklámok/kompromisszumok? Támogatók?

Rajtunk nem sok fog múlni, de érthető módon a szabadság iránti igényünk gyakran nagyobb mint a kompromisszumkészségünk ;-). Az előbbiektől függetlenül minden támogatást szívesen veszünk és előre is köszönjük!

Szerzőtársaim nevében is, tisztelettel és köszönettel,

Horváth Örs Apor
az alapító, főszerkesztő
2008 július 17, Budapest

SZABADNA, KÉREM?

Kezdetben vala a nyílt forráskód, majd lőn a szabad felhasználású operációs rendszer és szoftver, közben pedig szépen szaporodott a globális fejlesztőközösség. Sokasodtak a szabad programok felhasználói is, csekélyebb mértékben. Észak-Amerikában, Ázsiában és Nyugat-Európában gyorsabban, Kelet-Közép-Európában lassabban. Cseh-, Lengyel-, Orosz- és Horvátországban, valamint Romániában nagyobb ütemben, Magyarországon ráérősebben. Az utóbbi néhány évben megmozdultak a vállalatok, az intézmények is: Nyugat-Európából és más földrészekről egyre-másra jönnek a hírek, hogy megint áttért egy-egy szervezet valamelyik nyílt forráskódú Linux használatára, Magyarországon alig bővül az átállók köre, mi inkább olvastgatjuk a lefagyásokat és fizetünk.

A helyzet

Ä,©, valamint œ, ☺, ♥ és ■. - Amikor az átlagfelhasználó egy korábbi dokumentumában vagy egy honlapon hasonlókat lát, megrémül. Ő ugyan kipróbálta a nyílt forráskódú (*FLOSS - Free/Libre/Open Source Software: szabad/nyílt forráskódú program, nem tévesztendő össze a zárt forráskódú, ám mégis ingyenes, azonban legtöbbször reklámokkal megtűzdelt programokkal, amelyek a freeware kategóriát alkotják) szövegszerkesztők, böngészők egyikét, de lám, rögtön beigazolódott, hogy ami ingyenes, az persze vírusos is - gondolja. Nem tud a karakterkódolás rejtelméről, és fogalma sincs a saját rendszere beállításairól. Ezért aztán rögtön le is szedi a szabad szoftvert és titokban tartja az egész kis kalandot. Picit még szorong, hogy főnöke, felesége észreveszi-e a csínyét, mert akkor kikap.

A hasonló sorsú érdeklődők számosan lehetnek, ha nincs is róluk statisztikai kimutatás. Egy-



egy - inkább elavult, mint friss - lista és dokumentum azonban már fellelhető arról, hogy mely hazai közintézmények, közigazgatási szervek és oktatási intézmények vezettek be FLOSS-t. A legteljesebb, 2006-ban indított, e cikk írása előtt utoljára 2008. április 7-én módosított lista az OpenOffice.org szabad irodai programsomag intézményi felhasználóit tartalmazza, önkéntes regisztráció alapján:

<http://hu.openoffice.org/references.html>

A névsorból sajnos nem minden esetben derül ki, hogy melyik cég, illetve intézmény mikor vezette be a program használatát, így az sem világos, hogy a legutóbbiak mikor álltak át. Ha azonban tovább kutat az ember, talál még sok oktatási intézményt és céget, ahol szabad szoftverek, vagy rendszerek (is) futnak. Ilyen például még a Magyar Szabadalmi Hivatal (1), ilyen az OTKA (Országos Tudományos Kutatási Alapprogramok) rendszere, sok - főleg műszaki - egyetem és főiskola, a Vám- és Pénzügyőrség, vagy ilyen volt a megboldogult Informatikai és Hírközlési Minisztérium is.

Jóval a Teleház-projekt elindítása után, ma már léteznek kezdeményezések a szociálisan hátrányos helyzetű települések lakosságának számítógéppel, internettel és ezekkel együtt szabad operációs rendszerrel való ellátására, mint az Index-alapító Nyíró András és a Netbox "Login Initiative" nevű, tavaly indult kezdeményezése (2), vagy a szegedi Biogreen Alapítvány használgépadományozó akciója - amelynél azonban nem világos, mikor mi fut a gépeken (3).

De egy 2005-ös dokumentum szerint kacérkodott már a magyar kormány is a nyílt forráskódú megoldások támogatásának a gondolatával (4), talán az EU-s eMayor projekt (5) hatására, amelyből Magyarország rövid úton kimaradt. Végül az Oktatási Minisztérium 2005 után 2008 elején ismét a piacvezető zárt

Hivatkozások:

1. <http://www.kronika.matav.hu/2007szept/Novell.htm>
2. <http://index.hu/tech/net/login2298/>
3. http://www.pafi.hu/_pafi/palyazat.nsf/f4193bd634e17813c1256fd5002c9e30/17129e78a9fd58a3c1257313002f039a?OpenDocument
4. http://misc.meh.hu/binary/7182_kietb_ossii.pdf
5. http://www.szsi.hu/kozlemenyek/sajto/2007/01/17/eMayor_project/
6. <https://kelt.mokk.bme.hu/lists/pipermail/cc-hungary/2007-December/002116.html> ; <http://www.manco.hu/index.php?gcPage=/public/hirek/hir.php&id=15803> ; <http://hup.hu/node/49313>

forrású rendszer mellett kötelezte el a jelenlegi és a leendő adófizetőket (6). Ami a nyílt szoftverek használatának hazai összeírását illeti, a Szabad Szoftver Intézet honlapján elérhető egy cégfelmérés kérdőíve (7). Az eddig - kb. 3 év alatt - befutott adatok sajnos egyelőre nincsenek összesítve, vagy legalább is nem publikusak. A letölthető kérdőíven azonban ismét csak nem szerepel kérdés arra vonatkozólag, hogy a cég MIKOR választotta egy-egy FLOSS-program alkalmazását, így az adatbázis, ha létezik, még korlátozottan sem alkalmas a magyarországi tendenciák megfigyelésére.

A FLOSS globális terjedéséről mindenki képből lehet, aki a hup.hu és más IT-portálok híreit figyeli. A wiki.hup.hu portálon további információk lelhetők fel a szabad szoftverek térnyeréséről (8). A nyílt forráskódú, Unix-alapú operációs rendszerek használatáról pedig a distrowatch.com oldal nyújt átfogó és naprakész tájékoztatást, a felhasználók bejelentései alapján. A szabad szoftverekkel távolabbi viszonyt ápoló olvasók kedvéért rögtön ide kívánczok a megjegyzés, hogy ha pl. csak az otthoni felhasználóknak szánt operációs rendszereket vesszük, akkor Unix/Linux-alapúból egyidejűleg több mint 400 féle változat - disztribúció - létezik, amelyeket az önkéntesek párhuzamosan fejlesztenek. Bár ezek között egymáshoz igen hasonlóak is szép számmal vannak, jelentős különbségek is mutatkoznak közöttük. Ezzel szemben a zárt forráskódú, pénzért vásárolható operációs rendszerek verziói időbeli egymásutániségben léteznek, tehát mindig egyetlen rendszer újabb és újabb - esetleg szűkebb vagy tágabb képességű - változatairól van szó, amelyet a kibocsátó cég továbbfejleszt. A Distrowatch statisztikáiból kiderül, hogy a szabad Linux-disztribúciók közül a világon az utóbbi három hónapban az Ubuntu volt a legnépszerűbb operációs rendszer, amelyet stabilan az openSUSE követett, a bronzérmes pedig a Fedora volt. A Linuxok legmegbízhatóbbikának tartott, egyik legkorábban indított disztribúció, a Debian az elmúlt évben pedig a 7., 8. helyet tartja.

A nem Linux típusú, de mégis Unix-alapú operációs rendszerek közé tartoznak a Sun, illetve az Apple cég termékei is, amelyek közül csupán néhány Sun-rendszer szabad szoftver (pl. a Nexenta, vagy az OpenSolaris - (9), a többi nem sorolható a FLOSS-kategóriába. Azonban a nemzetközi fejlesztő-közösség annál több szabad programot tett és tesz folyamatosan elérhetővé a Windows-

felhasználók számára is (10).

Milyen operációs rendszer, illetve program mellett döntsek? - A kérdést minden felhasználó és rendszergazda felteszi magának, aki éppen felújít, vagy újat szerez be. Itt csupán a szoftverválasztás legfőbb szempontjait vesszük sorra, az egyes hardver-elemekről szóló megfontolásokat nem. Ideális esetben egyébként az előbbi megelőzi az utóbbit.

Hogy miért érdemes FLOSS-t választani, arra - tetszik, nem tetszik - itt és ma a legelső válasz az, hogy: mert ingyenes. Persze, a cégek jó, ha költségelemzést folytatnak egy új operációs rendszer beszerzése előtt és az ingyenes szoftver mellé az alkalmas rendszergazda munkabérét, a munkatársak esetleges betanítási díját, vagy a fizető ügyfelek igényeit is figyelembe veszik. Otthoni felhasználók esetében az ingyenes beszerzés mellé néhány órányi tanulás is járul - több, ha önálló telepítésre vállalkozunk, kevesebb, ha másvalaki telepít, és mi csak használjuk a programokat. A telepítés a kiforrott disztribúcióknál egyszerű, grafikus telepítővel megoldható, csupán a particionálásról és a fájlrendszerekről szükséges valamilyen előzetes ismereteket szerezni, valamint egy idegen nyelv (angol vagy más világnyelv) legalább alapszintű ismerete is kell hozzá.

A miéltre a második válasz az, hogy mert a nyílt kódú (nagy, kiforrott) rendszerek stabilabbak, mint a piacon kapható operációs rendszerek. Ennek oka, hogy nem egy zárt, főként fizetéssel motivált csapat fejleszti őket, hanem egy általában sokkal nagyobb létszámú, önkéntes, nemzetközi közösség: több szem, több szempontból többet lát (*ld. erről Eric S. Raymond "A katedrális és a bazár c. klasszikus esszéjét - (11)). A stabilitás tapasztalat, csupán statisztikailag lenne bizonyítható: aki nem hiszi, próbálja ki, itt csak annyit, hogy soha sem veszi át hosszú percekre a háttérben egy alkalmazás a rendszer irányítását - hacsak azt mi magunk be nem állítottuk előzőleg.

Harmadszor: vírustámadás alig irányul linuxos gépek ellen. Bár léteznek linuxos vírusvédő programok is, nem kell újra és újra a beszerzésükkel, a licenszük meghosszabbításával, a frissítésükkel foglalkoznunk ahelyett, hogy a munkánkat végeznénk.

Hivatkozások:

7. <http://www.szsz.hu/projektek/2005/06/17/cegtar/>

8. http://wiki.hup.hu/index.php/Szabad_szoftverek_térnyerése

9. <http://www.nexenta.org/os> ; <http://opensolaris.org>

10. http://wiki.hup.hu/index.php/Desktop_alkalmazások#Windows ; <http://download.winboard.org>

11. <http://magyar-irodalom.elte.hu/robert/szovegek/bazar/> ; <http://www.catb.org/~esr>

Negyedszer: a Unix-alapú rendszerek mindegyike modulárisan épül fel, ezért gazdaságosabb. Sokkal jobban feladatokra, személyekre szabhatóak már a telepítéskor, mint a piacon kapható rendszerek, amelyeknél a gépeket a fölöslegesen futó alkalmazások is terhelik.

Még egy fontos érv, ami talán egy kis országban nem mindegy: a párhuzamosan több nyelven történő felhasználást a Linux-disztribúciók jobban támogatják. Ugyanakkor szögezzük le rögtön: az alapvető (szövegszerkesztő, képfeldolgozó, böngésző-, levelező- stb.) programok mindegyike teljes egészében magyar nyelven, magyar súgóval is működik - hála a szoftverhonosítók folyamatos munkájáért.

A szabad operációs rendszerekkel szembeni, első felvetés, hogy "vajon meddig tart a disztribúció támogatása?". A legelterjedtebbek esetén a fejlesztői munka évekre előre tervezett és folyamatos, ugyanúgy, mint a zárt forráskódú operációs rendszereknél, illetve a támogatás is legalább ugyanolyan sokáig tart, ha hivatalosan nem is garantált. Viszont a szabad rendszereknek a támogatása is nyílt: a segítő cikkek a laikus számára is logikusak, érthetőek, nem hallgatnak el odavágó információkat, a megadott linkek pedig valóban tovább vezetnek a probléma megoldásában. A frissítések szintén átláthatóak, és sokkal jobban az adott gépre/feladatokra szabhatóak, mint a fizetős rendszerek, illetve szoftverek esetén. Az ugyan igaz, hogy sok disztribúció dokumentáltsága kívánnivalókat hagy maga után, de a legnagyobbaké (Debian, Ubuntu, openSUSE) legalább olyan teljes és folyamatosan frissített, mint a vásárolható rendszereké, miközben sokkal praktikusabban használható.

A Linuxokkal szembeni, egyik legfőbb, közhelyes érv, ami sokszor hozzá nem értésből elhangzik, hogy "nem elég felhasználóbarátak". Erre a válasz, hogy másféle felhasználói beidegződések UTÁN egy kis odafigyelést igényel az új rendszer kezelése (tenisz után a pingpong-tanulás), de korábbi beidegződések, megszokások NÉLKÜL könnyebben vagy ugyanúgy elsajátítható, mint a nem szabad rendszerek használata (tenisz- vagy pingpongtanulás). Ide kapcsolódó dilemma a szabad vagy nem-szabad szoftver közötti választáskor, hogy érdemes-e a majdani felhasználó(k)ra ráerőltetni egy neki(k) szokatlan grafikus felhasználói felületet - ugyanis az "áttanulás" leggyakrabban ezt jelentheti vállalati szinten. A Linuxok két legelterjedtebb grafikus felülete két különböző filozófián alapul: a Gnome arra törekszik, hogy mindig csak azt az - ablakkezelési, menü-beállítási - lehetőséget

mutassa meg, amit a felhasználó éppen keres. A KDE ezzel szemben lehetőleg mindenről egyszerre áttekintést kíván nyújtani. Mindkét megjelenítőre igaz azonban, hogy a rendszertelepítés után azonnal számtalan egyéni beállítási lehetőségre ad lehetőséget a rendszertálca és a több asztal viselkedésétől kezdve a szín- és más grafikus beállításokon keresztül a menüjellemzők beállításának a módozataiig - vagyis: ez is személyre/feladatokra szabható, sokkal nagyobb lehetőségekkel, mint az árusított vetélytársak legtöbbjénél.

Utoljára maradt a "rendszerváltás" előtt talán döntő fontosságú szempont, az egyes programformátumok kompatibilitásának a kérdése. Itt is sorjáznak az ismerethiányból fakadó közhelyek, mint pl.: "a Word nem kezeli a nyílt dokumentumformátumba, ODF-be mentett dokumentumokat". Erre az első válasz, hogy viszont a szabad szövegszerkesztők (OpenOffice.org, AbiWord, Kword) mind ismerik a Word formátumát, a .doc-ot és az is beállítható bennük, hogy alaphoz abban mentsenek mindent. A Word2007 .docx formátumát pedig hamarosan szintén felismerik majd, azt azonban a korábbi Word-verziók maguk sem ismerik fel egyből, hanem csak egy kiegészítő frissítés telepítése után - okoz is ez elég bajt a cégeknek mostanában. A második válasz, hogy egyébként a Word2000, a WordXP és a Word2003 is kiválóan felismeri a nyílt .odt formátumot és ment is abba, miután telepítettük a Sun következő pluginjét: http://www.sun.com/software/star/odf_plugin/specs.jsp. A kompatibilitás a szövegszerkesztésnél tehát teljes, sőt a táblázatformátumoknál is csaknem az.

A kép-formátumoknál tökéletes az átjárás, azzal, hogy a FLOSS programok (pl. Gimp, akár Windows alatt is) telepítés után máris nagyon sok formátumot ismernek. A média-formátumok esetében a linuxos rendszerek egyetlen hátránya (bizonyos szempontból) a vezető operációs rendszerben található médialejátszóhoz képest, hogy a MIDI-hangfájlok kezelését nem integrálták a CD-DVD-lejátszóba, hanem ahhoz a külön alkalmazások (zeneszerző-, kotta-tanító-, hangszervezélő programok) egyike szükséges. A linuxos lejátszók sokasága kezelni tudja viszont egyből a legtöbb ismert formátumot, beleértve Quick Time-féle .mov-ot is, amellyel sokan először a fényképezőgéppel készített videófelevételek lejátszása HELYETT találkoznak. (Erre is egyszerű megoldást ad - Windows alatt is - a magyar fejlesztésű és világszerte méltán népszerű mplayer, vagy a VLC nevű FLOSS-program.)

A FLOSS-böngészők gyorsabbak a kereskedelmi szoftverekkel szállított

böngészőknél és néhány tekintetben biztonságosabbak is. Elég itt csak a Mozilla Firefox utóbbi években tapasztalt térnyerésére hivatkozni az Internet Explorerrel szemben. Ami a levelező programokat illeti, ez az egyik olyan alkalmazásterület, ahol igazán a megszokás az úr. Míg az Outlookok a többi Windows-alkalmazással kiválóan együttműködhetnek, addig a legnépszerűbb FLOSS-levelezőprogramok (Evolution, Mozilla Thunderbird) linuxos és windowsos környezetben egyaránt megállják a helyüket, nulla forintért. Az azonnali üzenetküldő alkalmazásoknál hasonló a helyzet, míg az RSS-olvasóknál minden platform esetében az ingyenesek hódítanak, akár FLOSS, akár freeware-változatban.

Piac és valóság

Nos, ha mindezeket megfontolta a felhasználónk, esetleg utána is nézett ennek-annak pontosabban, hogy dönt? Valószínűleg másképp, mint amikor a többség-elv szerint választ: "azt használom, amit a többiek is használnak". Másképp, mint amikor naivan hisz a marketingnek: "csak az a minőség, amiért fizetsz", lehetőleg sokat és sokszor. Másképp, mint amikor úgymond idő- és pénzhiányban szenved: "nincs időm, nincs pénzem kísérletezni, tervezni", és aztán megfizeti majd a spórolást időben is, pénzben is. Ami pedig a rendszergazdákat és a cégeket illeti, köztudott, hogy sajnos Magyarországon mostanában nem a tervezés és az előrelátás az erősségük. A gazdasági és kulturális környezet tükröződik nagy valószínűséggel a számítógépes rendszerek és alkalmazások megválasztásában és kezelésében is: nem a tervezés és a lépések kidolgozása a módszer, hanem a spontán rögtönzés, majd - baj vagy új igény esetén - a tűzoltás (tisztelet a kivételeknek!).

Az, hogy mégis nő a FLOSS-felhasználók aránya, annak is tulajdonítható, hogy a pénzhiány gondolkodásra késztet. Közben a professzionális termék-marketing pedig egyre nagyobb ráfordítással képes csak viszonylag egyre kisebb sikereket elérni. Lehet, hogy merész, de ebből az a következtetés, hogy a szabad szoftverhasználatot támogató közösségek számára most van itt a pillanat, hogy szervezettebb és kiszámíthatóbb marketinget folytassanak a felhasználók megnyeréséért. Mondjuk, a csakis szakmai, programozói kérdésekre összpontosító, (bocsánat) sokszor inkább belterjes, mint nyitott tevékenységek mellett. Hogy félreértés ne essék: szükség van, nagyon, arra is, hogy legyenek magyar fejlesztések, legyenek magyar fejlesztők. De tetszhalott közösségekre, kezdeményezésekre nincs.

A boltokban meg-megjelenő FLOSS-rendszerekről csakis a hardvergyártók döntenek. Egy Linpus Linux egy laptopra előtelepítve (Acer, Asus) grafikus

felület nélkül többet árt a nyílt forráskódú operációs rendszerek elterjedésének, mint amennyit használ: visszariasztja a hozzá nem értőket. A géphez mellékelt Knoppix-live-CD, amiről az eladó melleleg azt hiszi, hogy az a gépre van telepítve, ugyancsak félrevezető. Ez ellen-marketing. Közben pedig Magyarországon alig akad egy-egy hely, könyvtár, művelődési ház, vagy bolt, ahova bárki bemehet és kipróbálhatja a

Hivatkozások:

12. <http://www.pecsa.hu/programok/cyber.htm>

szabad operációs rendszerek és programok működését. Jó ellenpélda lehet azonban a néhány internet-kávézó, ahol Linux (is) fut, a legutóbbi áprilisban nyílt meg a budapesti Petőfi Csarnokban (12). Szóljon, aki tud még ilyet! Mert önmagában a szöveg soha sem elég meggyőző.

Jankovich Oszkár

2008.05.10.

A cikkhez kapcsolódó fórum címe:

http://www.flosszine.org/szabadna_kerem



A NYÍLT FORRÁSÚ ÉS SZABAD SZOFTVEREK

HELYZETE AZ EURÓPAI UNIÓBAN

1 Háttér

1.1 Licenckek

Miért is szükségesek licenckek egy nyílt forrású, vagy szabad szoftver számára?

A kérdésre adott válasz a fennálló társadalmi rendszerben, valamint az azok által meghatározott jogi viszonyokban található meg.

A szerzői jog (2.1), valamint a szoftverszabadalmak (2.2) keretei között érvényesülni igyekvő fejlesztési modellek, ha nem is kénytelenek, de szükségképpen ráutaltak ezen eljárás módokra. A különbözőség a hozzáállásban keresendő. Míg a zárt forrású rendszereket fejlesztő csoportok védeni kívánják saját szellemi tulajdonukat azáltal, hogy másokat kizárnak azok hasznosításából, addig itt ezzel éppen ellentétes gondolatmenet fogalmazódik meg. Ebben az esetben mások jogainak kiterjesztése a cél, a közösség érdekének figyelembe vételével. Egyetlen megszorítás van, mellyel számolnunk kell; nem élhetünk vissza a nyílt forrás adta lehetőségekkel, vagyis az információ áramlását nem törhetjük meg oly módon, hogy valamely nyílt forrású szoftverkomponenst úgy használunk, vagy fejlesztünk tovább, hogy az ebben elért eredményeinket nem tesszük közzé. Az ilyen típusú licenclés betartatása kényes feladat [9], hiszen bármi nemű erőszakos fellépés ellentétes lenne magával a filozófiával.

Maga a nyílt forrás csupán azt jelenti, hogy hozzáférhetünk egy adott szoftver kódjához, ám nem jelenti azt, hogy azt szabadon használhatjuk fel, hogy ingyenesen érhetjük el, még ha ezen két feltétel jellemzően adott a nyílt forrású alkalmazások esetén. Röviden az alábbi pontokban foglalhatóak össze a nyílt forrású licenckek feltételei:

1. Szabad terjeszthetőség: a forráskód szabadon továbbadható vagy terjeszthető.
2. Forráskód: a szoftverrel együtt terjesztendő, vagy szabadon

elérhetővé teendő.

3. Származtatott munka: az újbóli terjesztéseknek vagy módosításoknak elérhetőeknek kell lenniük.
4. Integritás a tulajdonos kódjával: a módosítások esetenként csak mint patchek terjeszthetők.
5. Nincs diszkrimináció személyek vagy csoportok ellen: senki nem zárható ki.
6. Nincs diszkrimináció felhasználási terület ellen: a kereskedelmi célú felhasználók nem rekeszthetők ki.
7. A licenc terjesztése: a terjesztőknek el kell fogadniuk a licencet és nem támaszthatnak további licencfeltételeket.
8. A licenc nem lehet termékspecifikus: a program nem licenclhető egy nagyobb disztribúció részeként.

9. A licenc nem korlátozhat más szoftvereket: a licenc nem írhatja elő, hogy más szoftverek is nyílt forrásuként kerüljenek terjesztésre.

10. A licenc technológia-független kell legyen: nem várhatóak el médiumfüggő módjai a licenc elfogadásának.

1.1.1 Szabad szoftver licenckek

Magán a nyílt forrás közösségén (Open Source Community) belül is számos licenclési eljárás használatos és ezek egy része nem tekinthető kompatibilisnek a többivel. A legelterjedtebben használt GPL licenc az, ami de facto szabványként egyre jellemzőbben szolgál viszonyítási alappal az ilyen típusú licenckek között. Az egységesítési törekvések azonban nem sok eredményt hoznak a magyarországi használat tekintetében, hiszen a honi jogalkotás küzd némi lemaradással abban a vonatkozásban, hogy a GPL alakilag érvényes formájú szerződésként érvényesíthető legyen [3]. Ennek egyik oka, hogy "Magyarországon jelenleg a kereskedelmi forgalomban megjelenő szoftvereken túl valamennyi szoftverre felhasználási szerződést kellene kötni, méghozzá írásban", így az interneten megkötött szerződések esetében sincs mód eltekinteni ettől a procedúrától. Másik jelentős akadályként említhető meg, hogy "a magyar jogrend jelenleg nem teszi lehetővé, hogy a szerző a felhasználásra



”bárkit” jogosítson”. Fentiek azonban nem jelentik, hogy egyáltalában nincs mód a GPL-re mint szerződésre hivatkozni, ”hiszen a bíróság mérlegelési jogkörébe tartozik, hogy annak tartalmát mennyiben tekinti a felek jogviszonyában irányadónak”.

A nemzetközi tapasztalatok fényében különösen időszerűnek látszik néhány módosító javaslat indítványozása. Elegendő megemlíteni a Harald Welte által a Skype Technologies SA ellen indított GPL-sértési ügyet, mely végül az eBay kötelékébe tartozó vállalat visszavonulásával végződött. Az alperes hosszas egyeztetéseket követően sem volt hajlandó eleget tenni a GPL-ben foglalt kötelezettségeinek, így a felperes azzal a keresettel élt a német bíróság felé, hogy az az alperest a tárgyalás időtartamára a kifogásolt tevékenységtől (a GPL-t sértő szoftvert felhasználó termék forgalmazása) tiltsa el. Válaszlépésként a vállalat jogászai a GPL német jogrendben való érvényességét vonták kétségbe (melyre hasonló esetben akár Magyarországon is lehetőségük lenne), amire azonban a bíróság véleménye szerint elegendő bizonyítékot bemutatni nem tudtak, így az első fokú elmarasztaló ítélet elfogadására kényszerültek. A GPL megsértésének és a szerzői jogi nyilatkozat érvényesítésének nem egyedi esetével állunk szemben, így okkal feltételezhető, hogy a fent említett hiányosságok nem feltétlenül vezetnének valamely szabad szoftver licenc-érvényesítésének kudarcához.

2 Jogi háttér

Az Európai Unió területén a dolgozat írásának időpontjában nincs mód szabadalmi igények érvényesítésére, ám a helyzet mégsem teljesen egyértelmű.

2.1 Szerzői jog

Magyarországon az 1999. évi LXXVI. törvény szabályozza egy szellemi alkotás szerzőjének jogait, lehetőségeit, valamint a szerzői jog gyakorlásának korlátait. A törvény bevezetője szerint:

”A technikai fejlődéssel lépést tartó, korszerű szerzői jogi szabályozás meghatározó szerepet tölt be a szellemi alkotás ösztönzésében, a nemzeti és az egyetemes kultúra értékeinek megóvásában; egyensúlyt teremt és tart fenn a szerzők és más jogosultak, valamint a felhasználók és a széles közönség érdekei között, tekintettel az oktatás, a művelődés, a tudományos kutatás és a szabad információhoz jutás igényeire is; gondoskodik továbbá a szerzői jog és a vele szomszédos jogok széles körű, hatékony érvényesüléséről.”

A szerzői jog az, amely biztosítja az alkotó számára –a 31.§ alapján 70 évig– hogy szellemi termékét tulajdonként kezelhesse és afelett a tulajdonos esetében megszokott módon a kizárólagos jogokat gyakorolja. Fontos megjegyezni, hogy a 9.§ értelmében a szerzői jog a publikálás tényétől függetlenül is létezik, továbbá a személyhez kötődő jogokról lemondani és azokat átruházni nem lehet. Ezen rendelkezés kapcsán is felmerülnek aggályok a GPL alkalmazhatóságával kapcsolatban, melyről már a korábbiakban tettünk említést. A szerzői jog fontos jellemzője továbbá –ellentétben a szabadalmi joggal– ingyenessége és úgymond automatikus mivolta, ami annyit tesz, hogy alkotót a jog bármiféle jogi procedúra (igénylés, lajstromba vétel) nélkül megilleti.

A szerzői jog alkalmas ugyan arra, hogy a jogtulajdonos a szellemi alkotását –mely szoftverek esetén forráskód formájában jelenhet meg– oltalom alá helyezze, azonban nem képes a szoftver alapját képező elgondolás, ötlet védelmére. Ennek okán, és a szerzői jogot kiegészítendő hozta létre a jogalkotó a szabadalmi jogot.

2.2 Szabadalmi jog

A szabadalmi jog az iparjogvédelem egy kulcsfontosságú oltalmi formája, mely a feltalálót hasonló módon védi találmánya kapcsán, mint azt a szerzői jog teszi az alkotó viszonylatában. Konkrétabban fogalmazva a szabadalmas tulajdonosává válik saját találmányának és annak hasznosításáról a szabadalmi oltalom ideje alatt szabadon rendelkezhet. A jogalkotó alapvető célja nem más, mint a feltalálók ösztönzése, a verseny elősegítése, valamint a kutatás és fejlesztés területén felmerülő jelentős költségek megtérülésének biztosítása azáltal, hogy a szabadalmast átmeneti monopolhelyzetbe juttatja versenytársaival szemben kreativitása jutalmaként.

A szerzői joggal mutatott hasonlóságok korlátosak, mivel mind az eljárás menetében, mind annak költségeiben eltérések mutatkoznak. Ellentétben a szerzői jog kapcsán megismertekkel, a szabadalmi joghoz nem elégséges feltétel, hogy a feltaláló a szellemi alkotásnak –mely ez esetben egy találmány formájában jelenik meg– bizonyíthatóan ”szerzője” legyen. Azt szabadalmazási eljárás alá kell vesse, ahol vizsgálat tárgyává teszik a formai követelmények mellett többek között azt is, hogy a szabadalom megfelel-e a jogszabályi definícióknak, így valóban lehetséges-e rá szabadalmi oltalom kiadása. A szabadalom megszerzésének joga kizárólagosan az elsőként benyújtót illeti meg, még akkor is, ha szabadalom kihirdetése előtt más feltaláló(k) az előzőektől bizonyíthatóan függetlenül, a korábbival hasonló, vagy

akár teljesen azonos felfedezést tettek. Utóbbiak semmilyen módon nem tarthatnak igényt felfedezésük szabadalmazására.

3 Számítógéppel megvalósított találmányok szabadalmazhatósága

A kifejezés maga az Európai Bizottság, Európai Parlament elé benyújtott előterjesztésében szerepel és később, mint szoftverszabadalom vonul be a köztudatba. A megtevésztő kifejezés elterjesztésében a média nem csekély szerepet vállal, – azzal a nyilvánvaló szándékkal, hogy a hosszú és bonyolult szakkifejezést egyszerűbb formára hozza és ezzel a közérthetőséget elősegítse – ám elkövette azt a hibát, hogy az eredeti jogalkotói szándék, mely épp a szoftverek szabadalmazhatósága ellen irányul, kerül az elnevezésben negálásra.

3.1 Az Európai Unió célja

Az Európai Unió szándékait számos fórumon ismertette, azok kikristályosodottak, a közösség előnyét szolgálják és összhangban vannak a közép- és hosszútávra megfogalmazott irányelvekkel. Ezen célkitűzések az alábbiakban foglalhatóak össze:

- A tagállamok szabadalmi politikájában tapasztalható különbségek felszámolása és ezáltal a megteremtett egység és átláthatóság hozta versenyelőny minél nagyobb mértékben történő kihasználása.
- A szabadalmi törvénykezésben az eltérések – illetve a korábbi rendelkezések okán esetlegesen meglévő pontatlanságok – megszüntetése.
- Saját szabadalmi politika kidolgozása a nemzetközi tapasztalatok alapján, elkerülve az Egyesült Államokban tapasztalható visszás helyzet okozta problémákat.
- A kis- és közép vállalkozások ösztönzése a kutatás és fejlesztés területén történő erősebb mértékű szerepvállalásra a Maastrichti Szerződéssel összhangban.
- A szoftverek szabadalmazhatóságának kizárása, annak érdekében, hogy az egyes alkalmazásuk önmagukban ne képezhessék szabadalom tárgyát.
- A műszaki jelleget nélkülöző találmányok szabadalmi oltalommal történő védelmének elutasítása.

A célok által kijelölt irány természetesen helyes, ennek kritikája csupán szélsőséges esetekben fogalmazódott meg. Ugyanakkor a választott módszerek tekintetében komoly fenyegetések fedezhetők fel, melyekre leginkább a szabad szoftverek, valamint a nyílt forrású fejlesztések elkötelezett hívei figyelmeztették a közösséget. Aggodalmaiknak a legkülönbözőbb fórumokon adtak hangot.

3.2 Az Európai Unió irányelve

A komoly vihart kavart – a médiában csak szoftverszabadalmi direktíva néven elhíresült a számítógéppel megvalósított találmányok szabadalmazhatóságáról szóló irányelvjavaslat (továbbiakban: irányelvjavaslat), az érdekelt felek olykor szöges ellentétben álló nyilatkozatait követően végül elutasításra került. A kedélyek azonban nem látszanak csillapodni, mivel mindkét fél vesztesnek érzi magát a szabadalmak felett vívott csatában és egyre inkább úgy tűnik, hogy a 2005-ben hozott döntés nem egyéb, mint hosszúra nyúlt tűzszünet.

Az Európai Bizottság és az Európai Tanács akkor együttdöntési eljárás keretében próbált kompromisszumra jutni, mely folyamat az alábbiakban foglalható össze: Az Európai Bizottság előterjesztése Az Európai Bizottság 2002. február 20-án terjeszti elő az irányelvjavaslatot.

Európai Parlament módosító javaslata Az Európai Parlament 2003. szeptember 24-én a beérkezett 64 módosító javaslatot elfogadja, ezek közül az Európai Bizottság 21-et ítélt elfogadhatónak.

A Versenyképességi Tanács döntése Az Európai Unió Versenyképességi Tanácsa 2004. május 18-án elfogadja a javaslatot.

Viták az irányelvjavaslatról Ekkor kezdődnek a később szakmaiatlanná váló és túlzásoktól sem mentes viták a javaslatról.

Az Informatikai és Hírközlési Minisztérium ellenállása Az Európai Koordinációs Tárcaközi Bizottság számára készült előterjesztés, mely a javaslat támogatása mellett áll ki, a minisztérium képviselőinek tiltakozását váltja ki.

A javaslat a Kormány elé kerül A Kormány kitart korábbi álláspontja mellett a 2004. szeptember 8-i ülésen.

Az Országgyűlés egyeztetést kezdeményez A Kormány ennek folyományaként 2004. október 6-án beterjeszti álláspontjavaslatát.

A közös álláspont elfogadása Az EKTB 2004. november 9-én elfogadja a magyar állásponttól szóló nyilatkozatot.

3.3 Megfogalmazott kritikák

3.3.1 Az Európai Szabadalmi Egyezmény

Elfogadására 1973-ban kerül sor, akkor még csupán néhány állam (Belgium, Egyesült Királyság, Franciaország, Hollandia, Luxemburg, Németország és Svájc) mond le ezzel szabadalmi ügyeinek nemzeti elbírálásáról. A meghívott tagországok közé később további 24 állam kerül (köztük Magyarország 2003. január 1-én), ami egyúttal azt is jelenti, hogy az Európai Unió tagállamai –néhány országtól

eltekinve– nem különböznek az Európai Szabadalmi Egyezményt elfogadó országoktól. Az alapító államok 1977-ben létrehozzák az Európai Szabadalmi Hivatal, melyről hangsúlyozandó, hogy nem áll jogi kapcsolatban az Európai Unióval, mégis jelentős befolyással bír a szoftverszabadalmi kérdések körében. A 2006-os adatok tanúsága szerint összesen 208500 szabadalmi kérvény került iktatásra, ami összehasonlítva a 2005-ös adattal (197400) 5,6%-os növekedést jelent. A teljes létszámból ijesztően magas arány (48,5%) származik az Európai Unión kívülről, 25,7% az Egyesült államokból, 16,4% Japánból, 9,4% pedig egyéb államokból. A vezető három szabadalmas 2006-ban benyújtott szabadalmi igényeik alapján a Philips (4425), a Samsung (2355) és Siemens (2319).

Nem csupán arra szükséges felhívni a figyelmet, hogy milyen külső függőséget okozhat az Európai Unió számára egy elhibázott javaslat elfogadása, hanem arra is, hogy az unión belül működő multinacionális vállalatok ilyen mennyiségű új találmányt évente aligha "állítanak elő", így gyaníthatóan valamely joggyakorlati hiba húzódik meg ezen jelenség hátterében.

Amint az oly sokszor elhangzott, az Európai Szabadalmi egyezmény kapcsán annak 52. cikkelye (2) és (3) bekezdésében kifejezetten tiltja a számítógépes programok szabadalmazhatóságát, ha azokra a szabadalom csupán ezen minőségükben vonatkozik (az angol terminológiával élve "as such"). Itt azonban érdemes egy megjegyzést tenni. Számos alkalommal került viták keresztüztüzebe ez a pont és az angolszász jogrendben használatos kifejezés, ugyanakkor kevés szó esik arról mit jelentene, ha a szoftvereket, mint olyanokat mégis szabadalmi oltalom alá lehetne vonni. Változás tulajdonképpen nem állna be, hiszen a szoftver –mint szellemi termék– már a szabadalmak nélkül is védhető a szerzői jog (2.1) eszközeivel, mivel az vonatkozik nem csak a forráskód, hanem a tárgykód formájában létező szoftverekre is. Azon érvelés a direktíva mellett, miszerint tiltaná a szoftverek szabadalmazhatóságát, gyenge lábakon áll, a valós problémáról csak eltereli az érdeklődő figyelmét.

A szerző előtt nem ismert olyan érdekelt fél, amely ne ismerne el egy szoftvert, mint fejlesztési eredményt és ne osztaná azt a véleményt, hogy a szerző, munkájának okán díjazásban kell részesülnön, ha arra igényt tart. Az igazi problémát a fejlesztési ötletek, módszerek esetleges szabadalmaztathatósága jelenti, mely Bill Gates szerint a következőket vonná maga után:

"Ha az emberek megértették volna, hogyan adnak szabadalmakat, amikor a ma használatos ötleteket kitalálták, és szabadalmat kértek volna rájuk, akkor az ipar ma teljes mozdulatlan állapotban állna."

Az érdekelt felek többsége kénytelen belátni, hogy a Európai Szabadalmi hivatal gyakorlatában egy sajnálatos tendencia kezd egyre erősebbé válni, mely a feltalálói tevékenység kapcsán egyre alacsonyabb mértéket tart elfogadhatónak. Itt szembesülünk azzal a ponttal, ahol az Európai Unió jogalkotói szándéka élesen

válk el a joggyakorlattól. Ennek okán erősödik a szoftverszabadalmi direktíva ellenzőinek ellenállása, mely oda vezet, hogy inkább helyezkednek teljesen a szabadalmazhatóság ellen, mint hagyják magukat kiszolgáltatni egy esetleges jogi kiskapunak, amelyről némi gyakorlati tapasztalattal belátható, hogy a tőkeerős cégek –hatásosan alkalmazott formális és informális eszközeik révén– számukra könnyedén érvényesíthető jogi eszközt faraghatnak versenytársaik kiszorítására. *"Egy jövőbeni, saját szabadalmakkal nem rendelkező, induló vállalkozásnak meg kell fizetnie azt az árat, amelyet az óriáscégek majd rákényszerítenek. Ez az ár magas is lehet, ugyanis a már jól menő cégeknek érdekében áll kizárni a jövőbeni versenytársaikat."*

Bill Gates (1991)

Fentieket alátámasztandó lássunk egy egyszerű összefüggést. Az Európai Szabadalmi Hivatal jelenleg 30000 €-ban határozza meg a szabadalmi eljárásának díját. Csak a 2006-os "szabadalmi nyertes" Philips több mint 132 millió euró (több, mint 30 milliárd forint) bevételt jelent a hivatal számára, ami kellő ösztönző erővel bír a szabadalmak mind népesebb táborának további gyarapítására.

3.3.2 A direktíva nyelvi gyengeségei

Erőteljes támadás éri a direktíva szövegezését a civil szervezetek részéről (a teljesség kedvéért meg kell azonban jegyezni, hogy a szabadalmi jogban járatos szakemberek védelmükbe veszik azt), ami egyfelől bizonyítja a társadalmi vitára való bocsátás intézményének létjogosultságát, másfelől táptalajt ad többeknek amúgy is gyökeret vert gyanakvásnak az Európai Unió bürokratikus mivolta felől, hiszen a megfogalmazott bírálatok hatása a későbbi módosításokra csekély. a direktívát. Vegyük tehát szemügyre az egyik legkritikusabb definíciót:

"b) "műszaki hozzájárulás": a technika állásához való hozzájárulás a technika egyik területén, ami új és a szakember számára nem nyilvánvaló. A műszaki hozzájárulást a technika állása és a teljes egészésként tekintett szabadalmi igénypont közötti különbség vizsgálata alapján kell megítélni, amely igénypontnak műszaki jellemzőt kell tartalmaznia, függetlenül attól, hogy ahhoz társulnak-e nem műszaki jellemzők is."

20/2005/EK közös álláspon 2. cikk [2]

A fenti meghatározás az ellenzők szemszögéből túlságosan is tág teret enged az értelmezésbeli különbségeknek, ami a félelmek szerint oda vezethet, hogy a már most is megfigyelhető alacsony hozzájárulási szintet engedő szabadalom-engedélyezési gyakorlat tovább súlyosbodik. Az LME és az IVSZ felvetette a számítógép és a szoftver fogalmak definiálásának szükségességét, mely első olvasatban túlzónak tűnhet, be kell azonban látnunk, hogy a piac hemzseg az olyan

termékektől, melyek bizonyos tekintetben számítógépnek tekinthetők, hiszen valamilyen mikroprocesszor-vezérelt elemet tartalmaznak, ugyanakkor bonyolultsági szintjük meg sem közelíti a mára általánosan elterjedt személyi számítógépeket. Helyénvaló a kritika, mely figyelmeztet, hogy nehéz időtálló meghatározást adni a fenti fogalmakra a technika jelenlegi fejlődési üteme mellett. Ezen túlmenően javaslat született arra is, hogy különbséget kell tenni a számítógéppel vezérelt eszközökre és a szoftverekre vonatkozó szabadalmak között. Ennek módszere a szabadalom tárgyának vizsgálat alá vonása, annak megállapítására, hogy az hatást gyakorol-e a fizikai világra, és ezt tenni meg a szabadalmaztathatóság peremfeltételévé. Ez a megoldás talán nem a gordiuszi csomó átvágása, mégis igen egyszerű és logikailag tiszta módszer lehetett volna a fogalmi útvesztők elkerülésére. Nincs kétség ugyanis afelől, hogy a törvényi szöveg technicus terminusokkal való feltöltése nem segít a jelenlegi helyzeten, sőt mi több, elszigeteli a szoftverek szabadalmazhatóságáról szóló jogszabályi részt az egyéb passzusoktól. Talán pont ez a jel hívhatja fel a figyelmünket arra, hogy a már említett technikai fejlődési szint láttán szükséges lehet elgondolkodni azon, nem érdemes-e a gazdaság egyik húzó ágazatának alappilléreire, a szoftverre, akár a szabadalmi törvény más területeitől némiképp elkülönülő részt szentelni a törvényben.

Az általánosságokon felülemelkedendő álljon itt egy közelmúltbeli példa, melyből egyértelműen kitűnik mi az, amit korrigálandó gyakorlatnak neveznek azok, akik a szoftverszabadalmi direktíva ellen foglalnak állást. A szabadalmas ez esetben a Huawei Technologies Corporation, a szabadalom tárgya pedig naplóbejegyzések átvitelének egy adott módszere, melynek része –az üzenet küldője által kezdeményezett– titkosított csatorna kiépítése, majd azon történő adatátvitel. Ahogy az a szabadalom leírásában több helyütt szerepel is, az elv maga már ismert. Itt kell megjegyezni, hogy a számos protokoll (pop3, smtp, ldap, imap, ...) részeként már létező metodika kevéssé tekinthető újszerűnek, és jelent a ”technika állásához való hozzájárulást” csak azon oknál fogva, mert egy újabb alkalmazási terület lehetősége merült fel. Kérdéses továbbá, hogy ennek a széles körben elterjedt és sikerrel alkalmazott módszernek szabadalmaztatása nem jelent-e kifejezett kockázatot azon protokollok alkalmazhatóságának tekintetében, melyek a szabadalom részeként lefektetett eszközt már korábban is használták. Mindemellett jogosan merül fel az a gondolat is, hogy az alkalmazási terület ugyan az informatika és a megvalósító eszközök számítógépek, mégis mennyiben tekinthető számítógéppel megvalósított találmánynak a konkrét módszer, mely végül a szabadalmi leírás részeként bejegyzésre került.

3.3.3 Nemzetközi tapasztalatok

Nincs bizonyíték (kontroll csoport hiányában) arra, hogy a direktíva az Egyesült Államok szabadalmi gyakorlata felé vezető úton indította volna el az Európai Uniót, mégis ésszerű figyelembe vennünk a hozzánk képest tapasztalati előnnyel rendelkező országok helyzetét.

Nemzetközi szinten tendenciaként figyelhető meg az országhatárokon, és nagyobb volumenű szövetségeken is átnyúló nemzetközi nagyvállalatok szabadalom-hajhászása. Természetesen nincs kétség afelől, hogy ezen cégek különösen nagy hangsúlyt fektetnek a kutatás és a fejlesztés előmozdítására, ugyanakkor észlelhetők az áthallások a hidegháborús katonai versenykezés és szabadalmak végletekig történő halmozása között. Nyilván senkinek nem áll szándékában ködös orwelli világképet festeni, ugyanakkor megneemtámadási szerződésként is felfoghatóak a kölcsönös szabadalom-hozzáférési megállapodások, melyeket a legnagyobb szabadalmasok kötnek egymással. Egyfelől természetesen ünnepelhetjük mindezt az innováció érdekében tett erőfeszítésként, de a realitások talaján maradva látni érdemes, hogy ezen vállalatok a szoftverpiacot képletesen szólva csatatérre változtatták, ahol a szabadalmak jelentik az elaknásított területeket. Egyelőre nem utal jel arra, hogy ezen aknák élesítésével bárki is érdemben foglalatoskodna, ugyanakkor lehetőséget adhatnak arra, hogy a feltörekvő versenytársaknak komoly akadályt jelentsenek. Nincsenek érdemben tapasztalatok szabadalomsértési peres eljárások tekintetében, hiszen kinek-kinek a kellő számú szabadalom birtoklása elég védelmet jelenthet és irracionális lépésként jelenik meg csupán a hasonló helyzetben lévők ”megtámadása”.

Még ha a fenti borús kép részeiben meg is valósul, a jelentős szoftvercégeknek félnivalója nincs egymástól egyfelől szerződéseik okán, másfelől azon oknál fogva, hogy bevételeik forrása az előállított szoftver melyben potenciálisan előfordulhatnak szabadalomsértések megteremtve ezáltal az egyensúlyi helyzetet. Van azonban egy kivételes vállalatípus melynek módjában áll a status quo felborítása. Képzeljünk el egy olyan szervezetet, mely fejlesztői tevékenységet egyáltalán nem folytat, ugyanakkor szabadalmi arzenálját a lehető legnagyobbra duzzasztja, így a fegyverkezési verseny élére tör. Ellene nem jelent hatásos fegyvert a szabadalmi raktárkészlet bevetése, hiszen bevételeinek eredője nem a szoftver, hanem maga a szabadalom. Jövedelmei csak annyiban függenek a versenytársaktól, hogy milyen mértékben tudja érvényesíteni saját szabadalmi bejegyzéseit. Természetesen a szoftverfejlesztő cégek tönkretételében így nem, de a rajtuk való élösködés minél hatékonyabb formájának kidolgozásában annál inkább érdekelt. Mára az angolszász szakirodalomban fogalom (patent troll) is született az ilyen

tevékenységet folytató cégekre, ami különösen jól példázza a jelenség terjedését. Az ártatlanság vélelme azonban mindenkit megillet és egyáltalán nem elhanyagolható az óriásvállalatoknak a szoftveripar fejlődésében betöltött szerepe.

3.3.4 Nyílt forrás gazdasági szerepe

Gyakran kerülnek előtérbe a jelen témában alkotott elemzések kapcsán azok a statisztikai adatok, melyek az egyes felek gazdasági súlyát és véleményét állítják párhuzamba. Egyértelmű az összefüggés a nagyobb potenciált képviselő szervezetek és azok munkahelyteremtő képessége között, de nem tévesztendő szem elől az a tény sem, hogy a lobby-tevékenységre való alkalmasság aránya is ezen összefüggés szerint alakul. Hangsúlyt kapnak azok az adatok is, melyek alátámasztani látszanak a szoftverszabadalmak létjogosultságát azáltal, hogy azon vállalkozások véleményét teszik közzé, melyek a szabadalmi rendszer nélkül nem látják biztosítva kutatásra és fejlesztésre fordított költségeik megtérülését. Ezek az elvárások természetesen jogosak, bár szabad piaci körülmények között csak ezen szándék nem feltétlenül kellene, hogy monopóliumok kialakításához vezessen. Mindemellet kevés szó esik azokról a fejlesztő cégekről, melyek a nyílt forrás és a szabad szoftverek költségcsökkentő hatásait kihasználva tudnak versenyképesen jelen lenni a piacon és ott említésre méltó eredményeket elérni.

Ezekén túlmenően számos Európai Unió tagállam közigazgatásának példája adhat okot az elgondolkodásra abban a tekintetben, hogy miért is választják ezek a társadalmilag némiképp fejlettebb országok azokat a megoldásokat, melyek mérséklék egy adott piaci szereplőtől való függőségüket, valamint miért értékelik úgy, hogy mindez költségeik jelentős csökkenését eredményezi.

3.3.5 Megoldási javaslatok

Az eddig született javaslatok az alábbiak szerint foglalhatóak össze:

Szabadalmi oltalom időtartamának csökkentése
Tekintettel a szoftver, mint termék életgörbéjének egyre rövidülő mivoltára, valamint az egyre csökkenő elévülési időre, kézenfekvőnek látszik a oltalom időtartamának akár drasztikus csökkentése is.

Szabadalmi oltalom költségeinek csökkentése
Az Európai Unió elsődleges céljai között szerepel a kis- és középvállalkozások támogatása. Ezzel összhangban az oltalomhoz jutás költségének csökkentésével elősegíthetővé válna ezen piaci szereplők innovatív részének számára a szabadalom hozta előnyök kihasználása.
Szabadalmi rendszer áttekinthetőbbé tétele
Az

Európai Unió kívüli szervezetek befolyásának megszüntetése a jogalkotási folyamatra.
Szabadalmi hivatal függetlenítése
A szabadalmak számosságának növelésében fennálló érdekeltség megszüntetése.

Alternatív licenclési megoldások integrációja
Korábban már említett magyarországi és az ahhoz hasonló törvényi hiányosságok kiküszöbölése.

Garanciális elemek beépítése az újítások minősége kapcsán
A műszaki hozzájárulás minimális mértékének legalább megközelítő meghatározása.

4 Konklúzió

Két gondolat összegzésképpen:

- A gazdasági potenciál és a innovativitás között a fejekben élő egyenes összefüggés megkérdőjelezhető, lévén a kisebb vállalatok gyakorta alkalmazkodóképesebbek, kényszerülnek olyan végletesnek tűnő megoldások megfontolására, melyek valóban újító erejük lehetnek. A humán tőke tekinthető szinte kizárólag megújuló energiaforrásnak ezen a téren, így különösen elgondolkodtató, hogy az a pénzmennyiség, melyet például a Philips a 2006-os évben szabadalmi eljárások keretében elköltött, Magyarországon (2 millió Ft/fejlesztő/hó költséggel számolva) több, mint 1000 fejlesztő foglalkoztatására lenne elegendő egy éven át. Jó néhányan az innovatív tevékenység védelmére hivatkozva tartják szükségesnek a szabadalmak kiterjesztését. Ugyanakkor a fenti példa is mutatja, hogy az adminisztratív megoldások

helyett választható olyan út is mely valódi fejlesztést jelent és egyszersmind az adott cég piaci verselőnyének biztosítását is jelentheti.

- A szabadalmak árnyékában elhomályosulni látszik a gondolat, mely John Forbes Nash, közgazdasági Alfred Nobel-emlékdíjas amerikai matematikus fejében született meg, miszerint a legnagyobb összhaszon akkor könyvelhető el, ha mind a magunk, mind a csoport érdekeit figyelembe vesszük. Ennek szellemében a hangsúlyt a szabadalmakról inkább a szabványokra –különösen a nyílt változataikra– kellene helyezni, melyek a szellemi termékek hozta előnyök kisajátítása helyett a közös álláspont, a konszenzus irányába mutatnak. Maga a gondolatmenet nem is újszerű, hiszen már maga Pál apostol is ezt írja a Filippibelieknek: ”Ne keresse ki-ki a maga hasznát, hanem mindenki a másikat is.”.

Pfeiffer Szilárd
2008. május 16.

A cikkhez tartozó fórum címe:

http://www.flosszine.org/a_nyilt_forrasu_es_szabad_szoftverek_helyzete_az_EUban

Hivatkozások:

- Alapítvány a szabad szoftverek magyarországi népszerűsítéséért és honosításáért.
- Eurlex.
- Szabad szoftver intézet.
- Bednay Dezső. Gazdasági és üzleti jog. LSI Informatikai Oktatóközpont, Budapest, 2001.
- Vámosi Zoltán-Kovács Edina. Az európai integráció kérdései. INOK Kft., 2005.
- Józsa Gábor. A számítógépes programok mint szellemi termékek védelmének lehetőségei és korlátai. Iparjogvédelmi és Szerzői Jogi Szemle, 108. évfolyan 1. szám.
- Ficsor Mihály-Hajdú Tamásné-Kiss Marietta-Penyigei Krisztina. Elemzés a számítógéppel megvalósított találmányok szabadalmazhatóságáról szóló irányelvjavaslatához fűződő hatásokról. Iparjogvédelmi és Szerzői Jogi Szemle, 109. évfolyan 6. szám.
- Dr. Ficsor Mihály. Végtelen történet - folytatódó viták a számítógéppel megvalósított találmányok szabadalmazhatóságáról. Iparjogvédelmi és Szerzői Jogi Szemle, 104. évfolyan 2. szám.
- Eben Moglen. A gnu gpl betartatása, 2001.
- Erik S. Raymond. A katedrális és a bazár. Kiskapu Kft., Budapest, 2004.
- Hámoriné Gál Éva. A számítógéppel megvalósított találmányok szabadalmazhatósága. Iparjogvédelmi és Szerzői Jogi Szemle, 109. évfolyan 3. szám.

CSOPORTMUNKA OPEN SOURCE KÖRNYEZETBEN

Mi is az a csoportmunka? Például a kubikosbrigád rakétaszerű előrehaladása az árokásási folyamat során? Talán. Szerencsére a definíciókat nekünk már nem kell meghatározni. Sőt, nem fogunk foglalkozni az informatikai környezetet nélkülöző csoportmunkával sem.

Innentől egyszerű a dolog, csoportmunka számítógépes környezetben. A számítógépek elterjedésével a különböző irodai feladatok elvégzését szövegszerkesztő, táblázatkezelő, stb programok segítették. Az irodai környezetben ez előtt, és ez után is létezett csoportmunka, viszont a megvalósítása sokat változott az évek során. A hálózatok elterjedése előtt a számítógépes adatcsere nehézkessége hátráltatta az igazi csoportmunka kialakulását, bár különböző próbálkozások már ekkor is voltak.

A számítógépes hálózatok elterjedése után megjelentek az olyan szoftverek is, amelyek lehetővé tették, hogy különböző emberek, különböző fizikai helyekről, ugyanazokon az anyagokon, állományokon dolgozhassanak. Mielőtt e cikk keretein belül feleslegesen belemerülnénk a csoportmunka szoftverek taglalásába, definícióiba, használjuk ki, hogy elektronikus kiadvány vagyunk, így azok akiket a téma ez irányú kifejtése érdekel, az alábbi linken találhatnak egy rövid összefoglalót.
<http://www.niif.hu/rendezvenyek/networkshop/97/tartalom/NWS/2/8/index.htm>

Csoportmunka szoftverek

A témában különböző fejlesztőktől, gyártóktól nagyon sokféle alkalmazást találhatunk.

A fizetős kategória, és az Open Source megoldások összehasonlításakor (mint már megszoktuk), nem kell szégyenkezniük a nyílt forrású megvalósításoknak. Csoportmunka sokféle lehet, most azokat az integrált programrendszereket említjük, amelyek a lehető legtöbb részét felölelik a területnek.

Fizetős kategória

Novell Groupwise: régóta fejlesztett, kiforrott megoldás, a Novelltől megszokott minőségben, megfelelő támogatottsággal.

Lotus Notes és Domino: az IBM szintén nem kezdő ezen a területen, a Lotus megvásárlása után remek szoftveregyüttest alakított ki a márkanév köré. Mindaz elmondható a csoportmunka megoldásairól, amit a Novellnél említettünk. Csoportmunka megoldások többféle kialakítása között választhatunk a Lotus termékek használata során.

Természetesen ezenkívül még nagyon sok programot találhatunk, ha a zsebünket égeti a krajcár, de ezeket azért említettük, mert feltételezzük, hogy a legtöbben hallottak már ezekről a szoftverekről.



Nyílt forrású megoldások

Az előbbiekkal versenyző alkalmazásoknál az a helyzet, mint az angol szótárak esetében (minél nagyobb, annál több jelentése lehet egy szónak, kifejezésnek), minél tovább keresgélünk, annál többet találunk.

A mi kiválasztási szempontjaink között előkelő helyen szerepeltek az alábbiak: lehetőleg minél jobb magyar nyelvi támogatással rendelkezzen, legyen kiforrott (amennyire ez lehetséges), legyen elterjedt és minél nagyobb részt valósítson meg a csoportmunka feladatokból.

Néhány név: PHP Groupware, Eridu, eGroupWare, stb.
(Ennél sokkal több használható megoldást találhatunk).

A továbbiakban az eGroupWare megoldással foglalkozunk, mert ez tűnt az egyik legjobb rendszernek a fenti szempontokat is figyelembe véve, valamint a legtöbb konkrét tapasztalatot is ezzel szereztük. Mivel nem tudtuk a programok töredékét sem megvizsgálni, nem lehet kizárni, hogy ennél sokkal jobbat is találhat valaki, nekünk a kívánt célokra ez megfelelt.

A program bemutatását egy konkrét megvalósítás tapasztalataival ötvöztük, az adott helyen több éve működik, ma is folyamatosan használják.

E-GroupWare

Az E-GroupWare egy nyílt forráskódú csoportmunka szoftver. Sajnos a korlátozott terjedelem nem teszi lehetővé az egész rendszer mélyreható ismertetését, de még az egész alkalmazasegyüttes részletes telepítésének leírását sem, így megpróbálunk a fontosabb dolgokra, a felmerülő problémák megoldási lehetőségeinek ismertetésére koncentrálni.

Fontosabb információk a rendszerrel kapcsolatban

Az eGroupWare egésze nyílt forrású és ingyenes (GPL), ennek ellenére (vagy éppen ezért) felveszi a versenyt a fizetős megoldásokkal, nyugodtan használható üzleti környezetben is, (a fejlesztők törekvése, hogy ez legyen a legjobb nyílt csoportmunka rendszer). Az ingyenesség itt is a szoftverre vonatkozik, a vele való foglalkozást, hozzáértést már meg kell fizetni (persze, ha valaki tud olyan embert aki felebarátságból, ingyen gondoz eGroupWare rendszereket, legyen olyan jó és szóljon :)). Ma már elérhető hozzá magyar nyelvű támogatás is (pénzért, többféle megvalósításban, de még mindig sokkal olcsóbban, mint a kereskedelmi rendszerek esetében).

Az eGroupWare egy csoportmunka szerver, amely a legtöbb csoportmunka területet lefedi. Futtatható a legtöbb operációs rendszeren és platformon (Linux, MAC, Windows, stb). Kapcsolódhatunk hozzá más csoportmunka kliensekkel is (Kontact, Evolution, Outlook), sőt elérhető mobilról vagy PDA-ról, a SyncML szinkronizációval. A rendszer nemzetközi (ez azt jelenti, hogy folyamatosan dolgoznak a nyelvi verziókon, a magyar nyelvi megvalósítás egészen előrehaladott a főbb modulok esetén. Jelenleg 25 nyelven dolgoznak, az angolhoz képest van egy kis lemaradás, de a hindi vagy a perzsa nyelvi megvalósításhoz képest sokkal jobban állunk :)).

A szerver futtatható Apache vagy IIS webserveren, és a legtöbb adatbázisszerverrel együttműködik. Az általunk legtöbbet citált megvalósítás együttműködő partnerei az Apache és a MySQL :). A klienseken a legtöbb

böngésző használható, így, ha megfelelőek a beállítások, bárholnan dolgozhatunk a rendszerrel.

A rendszer honlapján minden lényeges információ megtalálható:

<http://www.egroupware.org/>.

A HUP-on is található egy jól használható összefoglaló: <http://hup.hu/node/27605>.

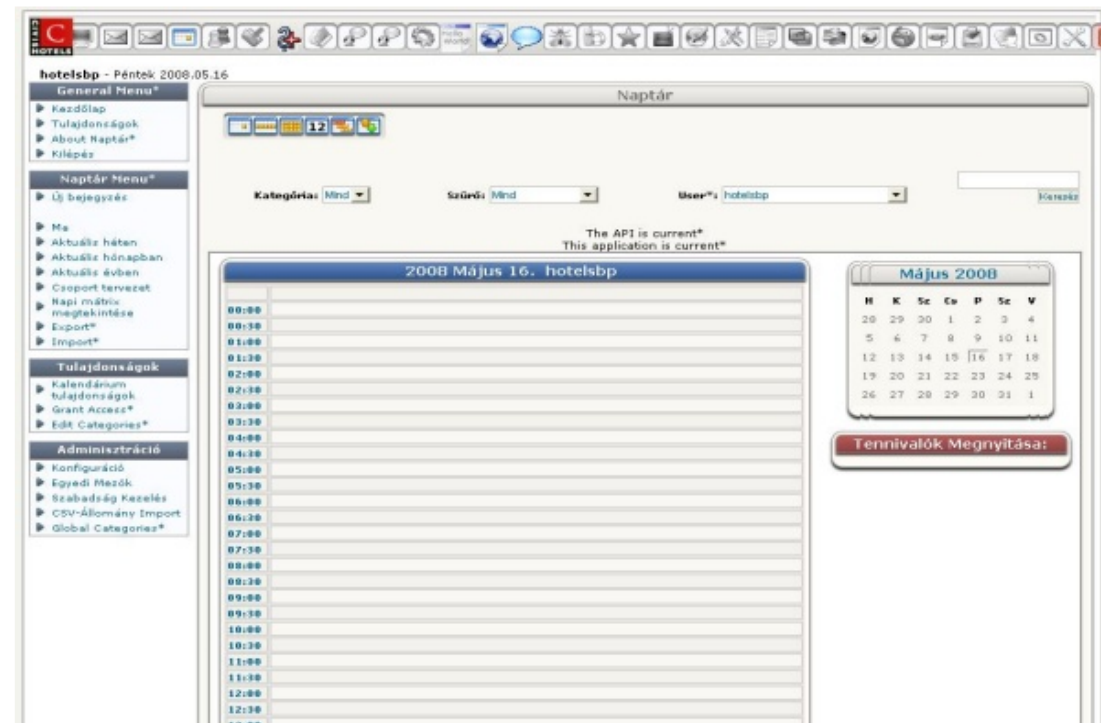
Ma már nyomtatott felhasználói kézikönyvet is rendelhetünk hozzá baráti áron (ennyit a nem eléggé dokumentált nyílt rendszerekről).

Mivel a fentebb jelzett helyeken részletesen megtalálhatók a rendszer moduljai, itt csak jelzésszerűen felsoroljuk a lehetőségeket, és később konkrét alkalmazásuk során némelyikre részletesebben is kitérünk.

Általános csoportmunka alkalmazások:

Naptár:

A többi alkalmazással együttműködő, remek naptár megvalósítás, többféle



nézettel, hivatkozásokkal. Adatcsere lehetőségek, csoportkezelés, stb.

Címjegyzék:

Az elhelyezhető adattartalom több mint elégséges, alapja SQL vagy LDAP. Itt is képezhetők kategóriák, van lehetőség adatok exportálására, importálására.

Teendők (Infolog):

Nagyon hasznos alkalmazás, bár az egész rendszer nem CRM célokra lett kifejlesztve, ennek a modulnak köszönhetően alapvető CRM feladatokat is elláthatunk az eGroupWare használatával. A korábban említett cégnél is főleg ilyen célokra használják.

Tennivalók, feljegyzések, telefonhívások nyomon követésére használható. Mint a legtöbb modulban, itt is létrehozhatók új kategóriák. A saját kategóriákat egy *-gal jelöli a rendszer.

FelaMiMail:

Nos ez egy remekül használható, IMAP levelező, jelenleg az eGroupWare standard e-mail alkalmazása.

További csoportmunka lehetőségek:

ProjectManager, Resources Manager, FileManager, SiteMgr, Timesheet, Tracker,

Teljes Neve	Cégnév	Üzleti Város	Üzleti Telefon	Mobil Telefon	Üzleti Email	Cselekedetek
	AVP Hungary KHE	Debrecen	52/311-940			[Icons]
	AVP Hungary KHE	Debrecen	52/311-940			[Icons]
	Dunia LLC & Partners Worldwide	San Remo				[Icons]
	West-line	Donyeck	8(062) 300 34 43			[Icons]
	Kartago Tours	Debrecen	52/533267		debrecen@kartagotours.hu	[Icons]
	Favonit tours	Donyeck	8 (062) 207 4271		favorittour@mail.ru	[Icons]
	Vetőmag Szövetség Szakmaközi Szervezet és Terméktanács	Budapest	061-332-5755	30-904-1360	nemethne.vetomag@hu	[Icons]
	Idriska tour	Kijev	+38 (044) 417-8185	8 800 301 1080	sales@idriska-tour.com	[Icons]
	Travel Point		8067 636168		travelpoint@yandex.ru	[Icons]
Kopodkoba	Abarem	Kneb	(+38044)		zsenia@avant-kiyev.com.ua	[Icons]

Wiki, KnowledgeBase, Workflow engine, NewsAdmin, Gallery2

Ezenkívül tartalmaz még sok egyéb, adminisztrációs, fejlesztői, stb segédeszközt és alkalmazást. Az alábbi ábrán a jelenlegi összes modul bekapcsolt (érdemes megnézni az ikonsort).

Most pedig lássuk a konkrétumokat: mit tegyünk, ha szeretnénk egy ilyen szervert üzembe állítani. Mindenekelőtt érdemes tisztázni az elvárásokat, sok marhaságot meg lehet csinálni az eGroupWare segítségével is, de a leendő felhasználók elképzeléseit vegyük komolyan, és fésüljük át a borzas igényeket. Lehet végül azt kell mondanunk nekik, beszélgessenek sokat, és cseréljenek információt gyakran, a teakonyhában.

Amennyiben úgy látjuk, hogy az elképzelések megoldhatóak az eGroupWare-rel, bátran vágjunk bele. Azt viszont ne felejtjük el, hogy a rendszer nyílt ugyan, de az eszement felhasználói elvárások applikálása nem kis feladat, ha magunkra vagyunk hagyva, a fejlesztés befolyásolására pedig nem sok lehetőségünk lesz a saját hozzáadott munkánk mértékén túl.

Amennyiben úgy tűnik nem érdemes foglalkozni a dologgal, bátran ajánljuk, hogy kérjenek árajánlatot egy pici és puha cégtől.

Jelen esetben egy inkább CRM jellegű dologra volt szükség, amit talán jobban teljesítene egy valódi CRM rendszer, ebből is van sok (pl.: SugarCRM), de az eGW a projekt indulásakor a többiekénél jobb magyar nyelvi támogatással rendelkezett, ezért erre esett a választás.

Ennél a cégnél főleg a naptár, teendők és címjegyzék modulokat használják.

Elég baj van egy ilyen rendszer beindítása és üzemeltetése során, így senkinek sem hiányzik a hardverekkel való bohóckodás. Ezért lehetőleg a hardveren ne spóroljunk, ha már a szoftveren megtettük.

A példának hozott rendszer alá ezért egy akkoriban (egészen pontosan 3-4 éve) jónak számító konfigurációt, egy Senorg rack szerveret tettünk, mára elavult, de még eddig semmi bajunk sem volt vele. Érdemes a költségkeretnek megfelelő adatvédelmi megoldásokat alkalmazni (Cluster, RAID, vagy rendszeres mentés, kinek mire van pénze).

Ha megvan a gép az operációs rendszerrel túl sok bajunk nem lesz, ízlés szerint

választhatunk, a példarendszer Debian-on fut. Ehhez kell az említett web szerver, és adatbázis szerver, jelen esetben Apache és MySQL.

Ezután telepíthetjük az eGW-t a következő helyről:

<http://www.egroupware.org/download>.

Aki telepítés nélkül szeretné kipróbálni a rendszert, erre is van lehetőség:

<http://egw-demo.stylite.de/currentversion/login.php>

Érdekes az adat és üzembiztonságról megfelelően gondoskodni, ebben az esetben a szerver egy tűzfal mögött van, ami csak meghatározott hálózatokról engedi a belépést (persze így le kellett mondani az internetkávézók munkahelyként való használatáról). A rendszeres mentés egy fizikailag másutt lévő gépre történik.

Ha mindezekkel készen vagyunk jöhet a rendszer finomhangolása. Ennél a cégnél a jogosultsági rendszer kialakítása volt a leginkább munkaigényes feladat, szerencsére az eGW támogatja a különböző csoportok képzését, és az adatállományok szükség szerint elkülöníthetők.

A felhasználóknak nem kell minden modult használniuk, sőt nem is kell tudniuk, hogy ezek léteznek, akár minden felhasználónak lehet más a menürendszer.

A különböző csoportok kialakításának lehetősége megkönnyíti a hasonló profilú felhasználói fiókok létrehozását.

Figyeljünk arra, hogy a beállításoknál a csoportokba rendezett felhasználók esetén, az alkalmazásokat a csoportnál adjuk meg.

A rendszer eléréséhez csak a szerver címét és a bejelentkezési adatokat kell ismernünk.

A cím böngészőbe való beírása után a bejelentkezési ablak jelenik meg, ennek kitöltése után a rendszer kezdőoldalán találjuk magunkat.

A különböző böngészők eltérően jeleníthetik meg az oldalt, egy alábbi képen látható lista például az Operában nem így jelenik meg, és a többszörös kijelölést sem alkalmazhatjuk a beviteli részen.

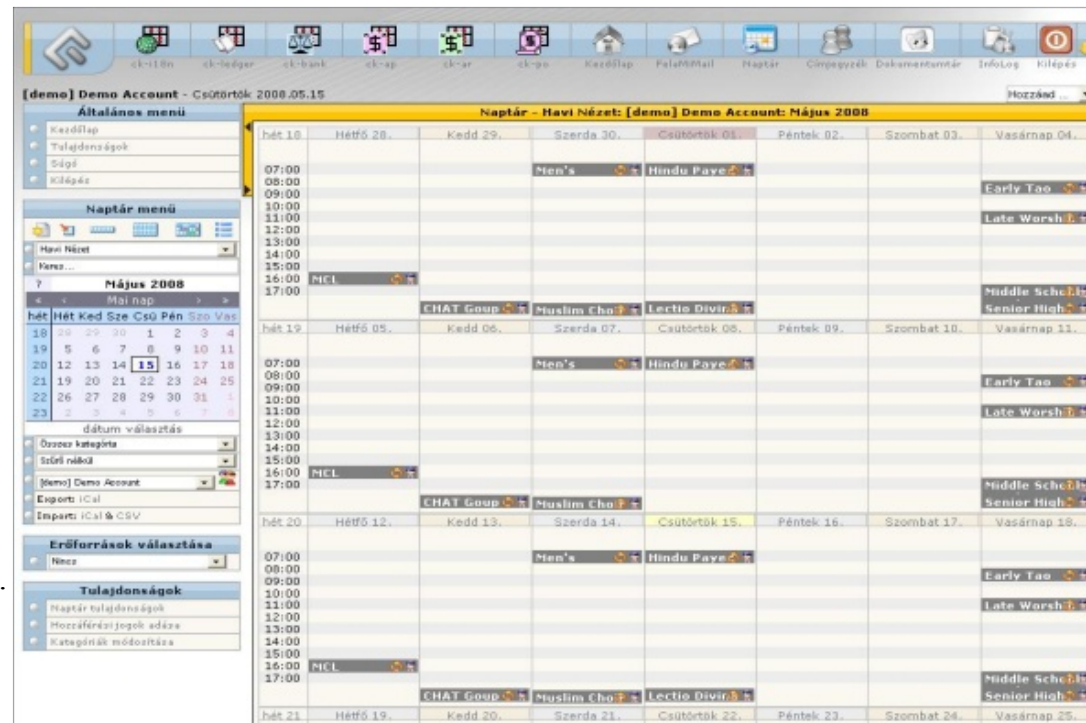
Az alapbeállítások is jól használhatóak, de ha már megismertük a rendszert

egészen érdekes dolgokat tudunk létrehozni.

A kinézet nagymértékben testre szabható. A felhasználók és csoportok teljesen elkülöníthetők. Egy felhasználó több csoportnak is tagja lehet, de nem kötelező a csoportba rendelés.

A modulokkal is szabadon bánhatunk, elég csak a valóban használt modulok megjelenítése, és a jogosultsági rendszer használatával kombinálva a legtöbb igény kielégíthető.

Egy kis teszteszabás eredménye látható az alábbi képen (a cikk végén említett cég munkája).



Az adatbázist érdemes lehetőleg teljes körűen kitölteni, később az adatok megjelenítésénél sokféle módon szűrhető az adattartalom, de ami nem került be a rendszerbe, az nem is lesz elérhető (különleges felhasználói igények esete, megtörtént :)).

A rendszer alap szűrői mellett létrehozhatunk sajátokat is.

Az adatbázisban kereshetünk, és van lehetőségünk az oldalkép nyomtatóra

optimalizálására is.

Az adatokat (megfelelő jogosultság esetén), a legtöbb nézetben megtekinthetjük, módosíthatjuk és törölhetjük. A jogosultságok finomhangolásával létrehozhatunk olyan felhasználókat is, akiknek csak megtekintési joguk van, ez tovább korlátozható, csak bizonyos adatokra. Egyeseknél kifejezetten ajánlott!

További lehetőség az adatok publikus és privát kezelése.

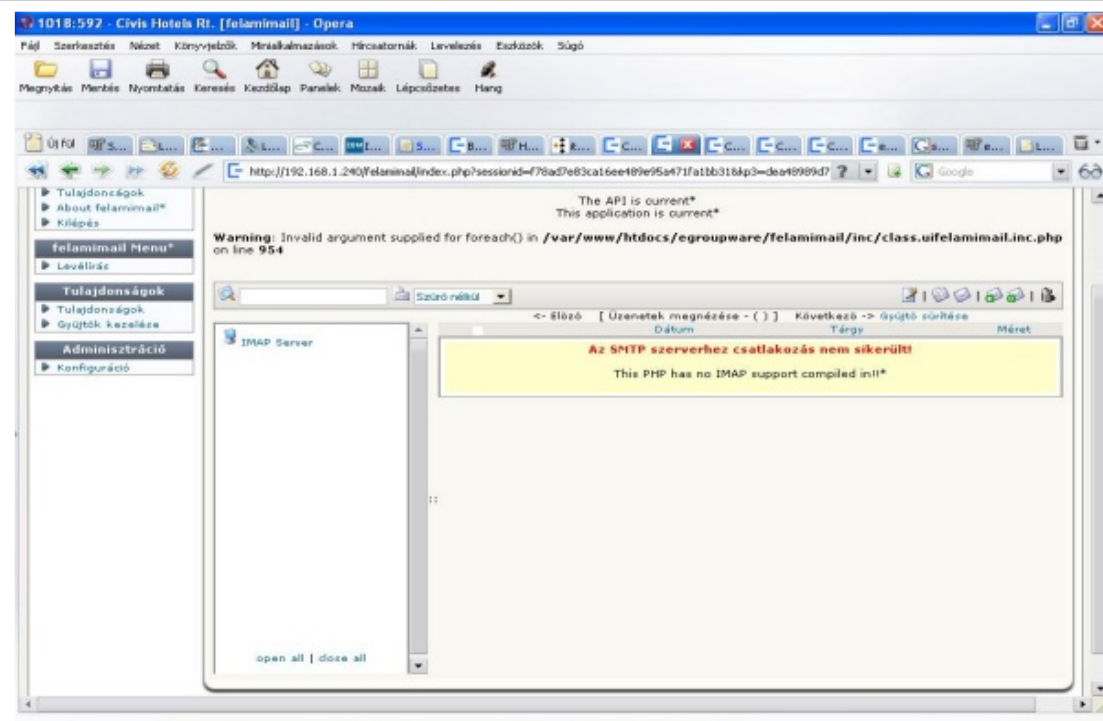
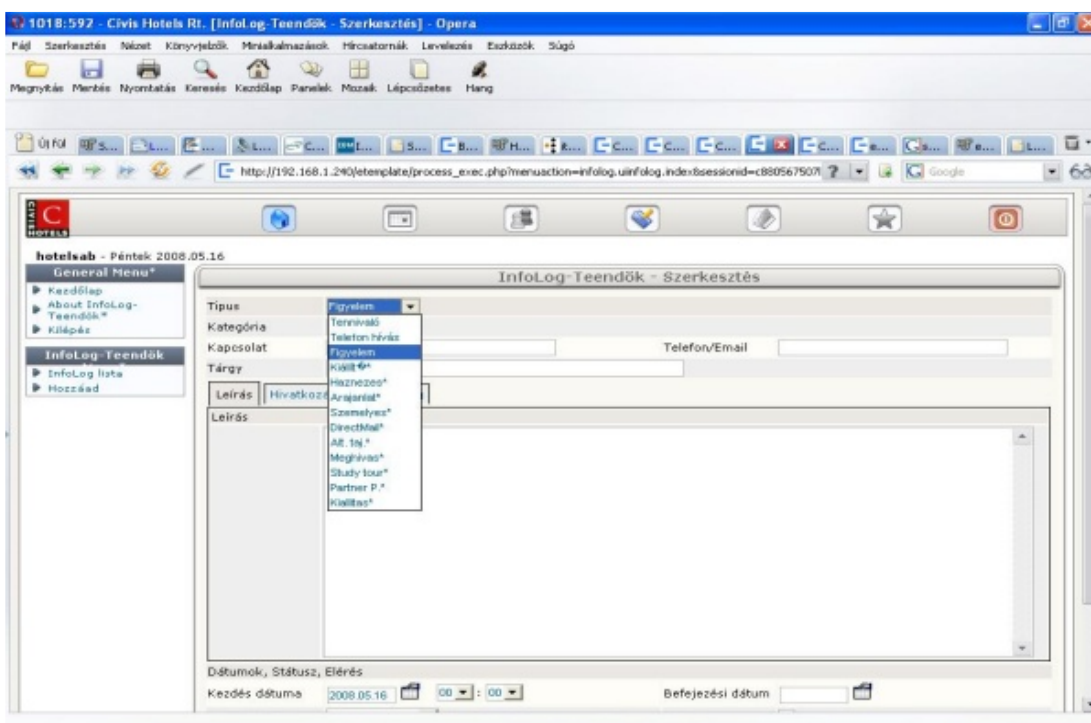
A legtöbb modulban a meglévőkhöz kívül új kategóriák, szűrők, stb hozhatók létre. Ezeket bizonyos esetekben a rendszer eltérően kezelheti, de a lehetőség nagyon hasznos.

Összefoglalva, az eGroupWare egy igen hasznos alkalmazáseggyüttes, nyilván nem lesz mindenki számára megfelelő, de ki ismer olyan rendszert, amit mindenki kiválóan tart?

Ezenfelül jóval kedvezőbb áron hozzáférhető, mint más rendszerek, hasonló funkcionalitással, és ezen felül van még néhány nem elhanyagolható előnye. A szerver oldal reszelgetése, frissítése során belefuthatunk ilyen vagy hasonló hibaüzenetekbe, de nem ez a jellemző (ebben az esetben sem a rendszer hibázott, hanem az adott sysadmin).

Ugye mennyivel barátságosabb és informatívabb hibaüzenet ez, mint....

Egy ilyen esetben a zárt kódú rendszereknél van lehetőségünk felhívni az adott



céges support számot és vagy segít vagy nem (inkább az utóbbi).

Ebben az esetben egy nem is telepített funkciót próbáltunk meghívni, így tudjuk mi okozza a problémát, de ha ez nem világos, még mindig nézegethetünk állományokat, forrást, stb, de a leginkább célravezető, a hozzáértők segítségének kérése, fórumokon, támogatási oldalakon, stb. Az esetek legnagyobb részében biztosan sokkal gyorsabban kapunk segítséget, mint zárt rendszer esetén. A másik kiváló dolog, amire lehetőségünk van, az a pénzkereset ezzel az ingyenes rendszerrel és a felhalmozott tudásunkkal.

Amit az alábbi oldalon elkövetnek, az bárki számára lehetséges.

<http://www.osbusiness.hu/>

Ha valakinek megtetszett a rendszer, és érez erőt magában a fejlesztéshez, honosításhoz, a nyílt rendszereknél megszokott módon csatlakozhat a fejlesztőkhöz, de akár sajátot is létrehozhat a licenc-nek megfelelően. Mindenkinek sok sikert, és bogármentes üzemeltetést kívánunk, az eGW használata, fejlesztése során, és reméljük a projekt a jövőben is töretlenül fejlődik tovább.

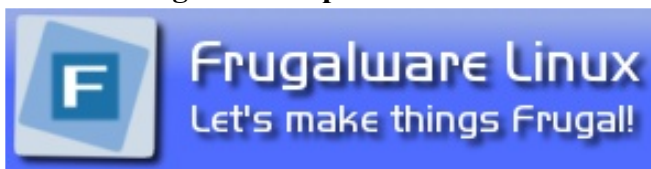
Szőke József

A cikkhez tartozó fórum címe:

http://www.flosszine.org/csoportmunka_open_source_kornyezetben_eGroupware

A Frugalware Linux múltja, jelene és jövője

Azt gondolhatnánk, hogy egy disztribúció születése különleges esemény, pedig nem az. Elég csak a népszerű DistroWatch.com oldal híreit figyelni, hetente



születnek új disztribúciók. A nyilvántartott disztribúciók száma 400 körül van. Örületes szám. Persze hozzá kell tenni, hogy sajnálatos módon a

legtöbb disztribúció hirtelen lelkesedéssel, mindenfajta racionális tervezés nélkül, általában egy másik disztribúción alapulva indul el. A lelkesedés tart pár hónapig/hétig/napig aztán eltűnik a sülylesztőben.

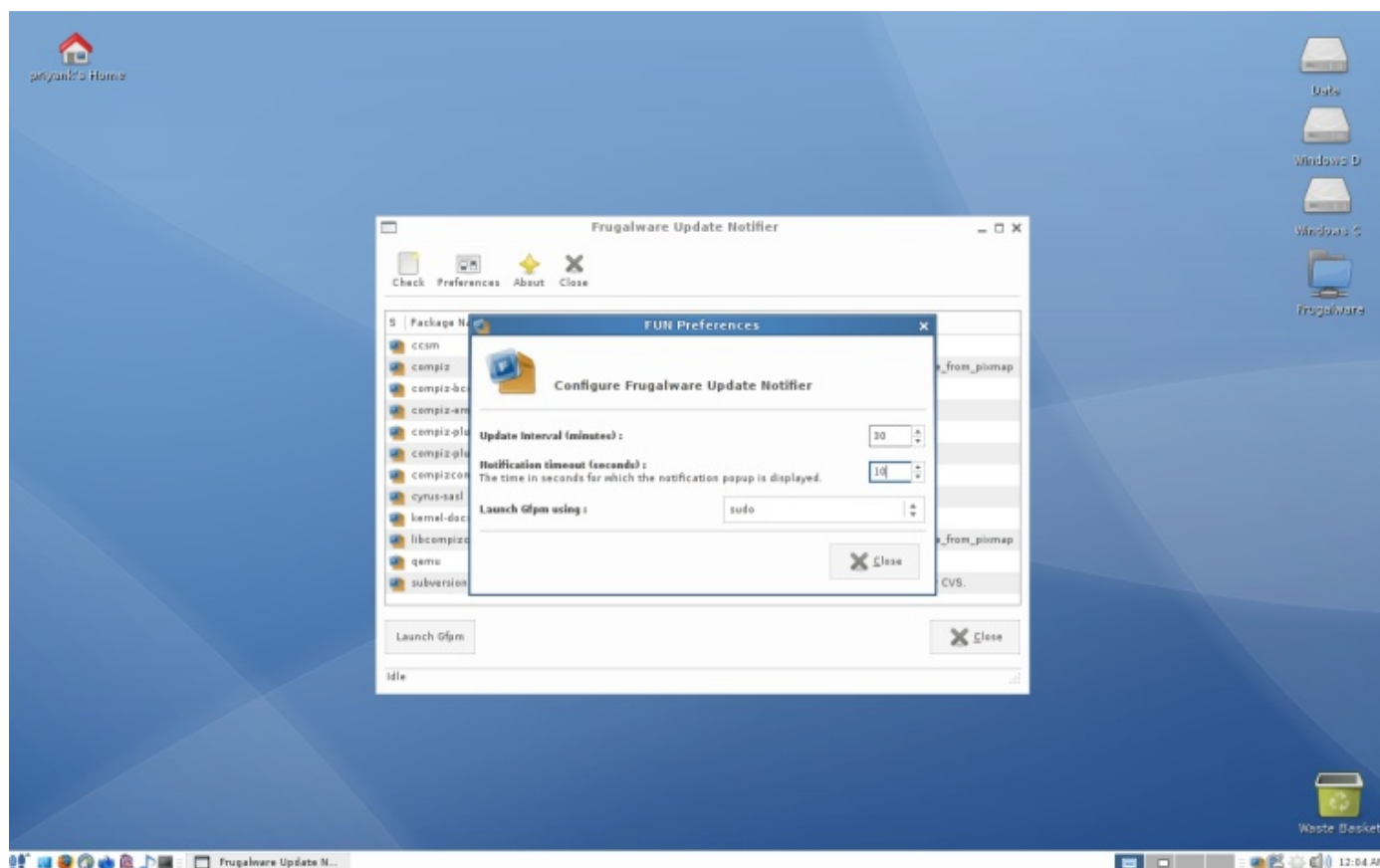
Természetesen ezeket leszámítva is számos terjesztésből válogathat a tisztelt felhasználó. És ez így is van rendjén. Nincs egyetlen univerzális operációs rendszer (mint ahogyan azt néhány fanatikus szeretné elhitetni), hanem adott célokra keresünk eszközt. Hiszen az operációs rendszer a felhasználó számára csak egy eszköz – hogy aztán azt munkára, szórakozásra, vagy mire használják, az már kevésbé igaztja a gyártókat.

Valószínű, hogy olyan helyen ahol fontos, hogy legyen telefonos ügyfélszolgálat és néhány órán belül a gyártó emberei a helyszínre érkezve orvosolják a problémákat nem célszerű közösségi alapon fejlesztett disztribúciót telepíteni, hiszen ezeket az emberek szabadidejükben fejlesztik. És fordítva, egy közösségi portál kiszolgálására tökéletesen meg fog felelni egy közösségi alapon fejlesztett rendszer: egy cég által fejlesztett fizetős rendszer költségét nehéz volna előállítani, egyéb problémákról nem is beszélve.

Nézzük meg egy kicsit közelebbről, hogy hogy megy ez Magyarországon. Itt már nem olyan mozgalmas a színpad. Az időrendi sorrendeket elhanyagolva tüzetes keresés után is talán három disztribúciót találtunk: UHU-Linux, blackPanther, Frugalware. Tekintve, hogy ez a cikk ez utóbbiról szól, az előző kettőről legyen elég annyi, hogy ezek mögött profit-orientált cég áll. Nem feltétlenül akarom őket negatív színben feltüntetni, de nyilvánvalóan céljuk a profit és nem a

felhasználók elégedettsége.

Ezzel szemben a Frugalware – Magyarországon elsőként – megkísérelt megvalósítani egy közösségi alapon fejlesztett rendszert. Hogy ez mennyire sikerült azt nem az én tisztem megállapítani, bár kétségtelenül az már önmagában elég negatív fényt vetne a fejlesztésre, ha kiderülne, hogy a fejlesztők se használják (ami egyébként nem igaz). Másik oldalról pedig valószínűleg nehéz kudarcként elkönyvelni egy olyan projektet amely külföldön is számottevő népszerűséget ért el, például jelenleg négy nyelven van közösségi IRC csatornánk, 6 nyelvű közösségi fórumunk van (leszámítva a létrehozott, de nem aktív fórumokat), a rendszer telepítője 13 nyelven beszél, és a disztribúciót a világ 11 pontján 8 országban tükrözik. Ugyan a rendszer fejlesztésének kezdetén (az első évben) nem használtunk verziókezelő rendszert, de a jelenleg regisztrált patch-ek (változtatások) száma – talán ezzel jól lehet mérni a fejlesztés mértékét – meghaladja a 32 ezret.

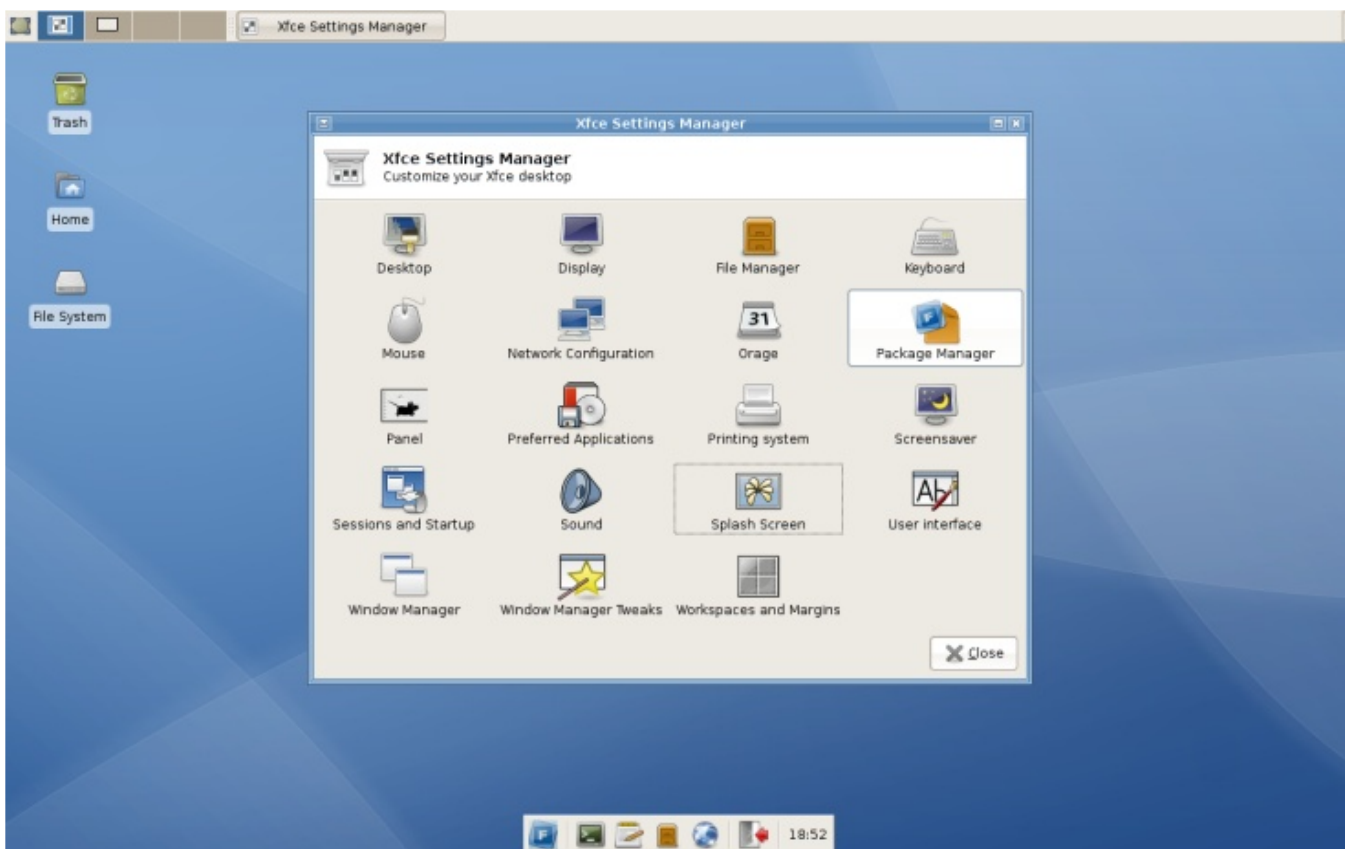


Ha valaki megkérdezi, hogy „jó-e a Frugalware?”, akkor erre a kérdésre nagyon egyszerű válaszolni: Jó. Persze az a válasz nem lesz hiteles. A kérdésre úgy lehet hiteles választ kapni, hogy letöltjük a telepítőt, kipróbáljuk, és magunk mondunk véleményt. Egy óra idő-ráfordítással már lehet eredményeket elérni. Az „ízére” ráérezni talán egy hét alatt, vagy inkább több.

A tisztelt Olvasó valószínűleg kíváncsi arra, hogy mégis miben más a Frugalware Linux mint egy másik disztribúció. Egy mondatban nehéz erre válaszolni, mivel a rendszer készítésekor nem húztunk meg határokat: jelenleg is használják kiszolgálóként és felhasználói munkaállomásként, magánemberek, cégek, kutatóintézetek, egyetemek, és még lehetne sorolni. Mégis, néhány szempont, amit igyekszünk szem előtt tartani, a sorrend és teljesség igénye nélkül:

Nyíltság

Ez azt jelenti, hogy a rendszer készítéséhez használt minden eszköz publikusan elérhető, ha valaki szeretne módosításokat készíteni privát használatra azt minden



gond nélkül megteheti.

Közösség

Mivel a rendszert nem pénzért fejlesztjük (bár van rá példa, hogy egy-egy funkció elkészítését valaki szponzorálja), nem a profit lebeg legfőbb célként a fejünk előtt, hanem az hogy számunkra és a felhasználók számára is egy használható rendszert készítsünk. Akárhogyan is értelmezzük a használhatóság fogalmát, az informatika gyorsan változó világában naprakésznek lenni bizony komoly kihívás.

Nemzetköziség

Korán rájöttünk, hogy sok olyan ember lehet akit érdekelhet a „termékünk”, de nem beszél magyarul, így a rendszer fejlesztése angol nyelven történik, majd ezt fordítják le lelkes önkéntesek sok nyelvre. Ebbe beleértendő a honlap, a (két)heti hírlevelek, a telepítő, a csomagkezelő, a dokumentáció, és még sok más.

Megbízhatóság

A Frugalware idén fogja ünnepelni negyedik születésnapját. Régen sokan kérdezték tőlünk, hogy: „Az szép, hogy ezt csináljátok, de mi lesz akkor ha én is felteszem, megszeretem és ti jövőre abbahagyjátok a fejlesztést?” Régen erre azt válaszoltuk, hogy ott a dokumentáció, ha tényleg annyira fontos a rendszer akkor csak pénz kérdése valakit fizetni aki a dokumentáció alapján felveszi a fejlesztés fonalát

(ismerős lehet a probléma „Vendor lock-in” néven). Az előző válasz most is igaz, de az idő előrehaladásával már van egy sokkal hangzatosabb válaszunk is: „Ha négy év alatt ilyen szép eredményeket érünk el, mennyire valószínű, hogy egyik napról a másikra minden fejlesztő megőrül és abbahagyja a fejlesztést?”

Kompatibilitás

A programok készítésénél örök kérdés, hogy hogyan lehet új funkciókat úgy megvalósítani, hogy kompatibilisek maradjuk az előző verziókkal. Erre a problémára mi azt találtuk ki, hogy fél évente adunk ki stabil kiadást. Az, hogy ebbe mi kerül bele és mi nem, az attól függ, hogy kinek mit volt ideje stabil állapotra elkészíteni a stabil kiadás időpontjára. Ezek között a kiadások között a frissítés gördülékeny és dokumentált. Így aztán a korábbi verziókkal való kompatibilitással csak fél évig kell nekünk is törődni, de a felhasználóknak sem kell bosszankodniuk.

A Frugalware első stabil kiadásának a 0.1 verziót adtuk. Ez két dolgot jelent. Egyrészt ez volt az első kiadás amit az elszántaknak már javasoltunk mindennapi használatra. Előtte a Frugalware csak egy érdekes projekt volt amiből majd lesz valami. A másik, amit jelent az, hogy tisztában voltunk azzal, hogy a felhasználók többségének az igényeit nem elégíti ki a rendszer, ezért nem indultunk 1.0-val. Később rájöttünk, hogy a verzióknak nincs különösebb jelentősége: ahogy a fentiekből már kiderült, nem biztos előre hogy egy adott verzióba pontosan mi kerül bele, így aztán nem a bekerült funkciók alapján kerül meghatározásra a verzió. Nincs nagyobb vagy kisebb ugrás, csak mindig egyel nagyobb, ahogy a fél év elteltével a következő kiadáshoz érünk. Azt szoktuk mondani, hogy aki attól fél, hogy 1.0 előtt nem használható a rendszer az 0.8 helyett vegye 8.0-nak a verziót, és akkor megnyugodhat. Ha valaki mégis meg szeretné határozni, hogy mióta használható széles körben a rendszer akkor azt kell megnézni, hogy milyen funkciókat követel meg a rendszertől.

Néhány példa:

- OpenOffice.org csomag bekerülése a rendszerbe: 0.2
- GNOME támogatás: 0.2
- Zárt alkalmazások támogatása (Skype, Opera, stb.): 0.3
- Biztonsági frissítések: 0.5

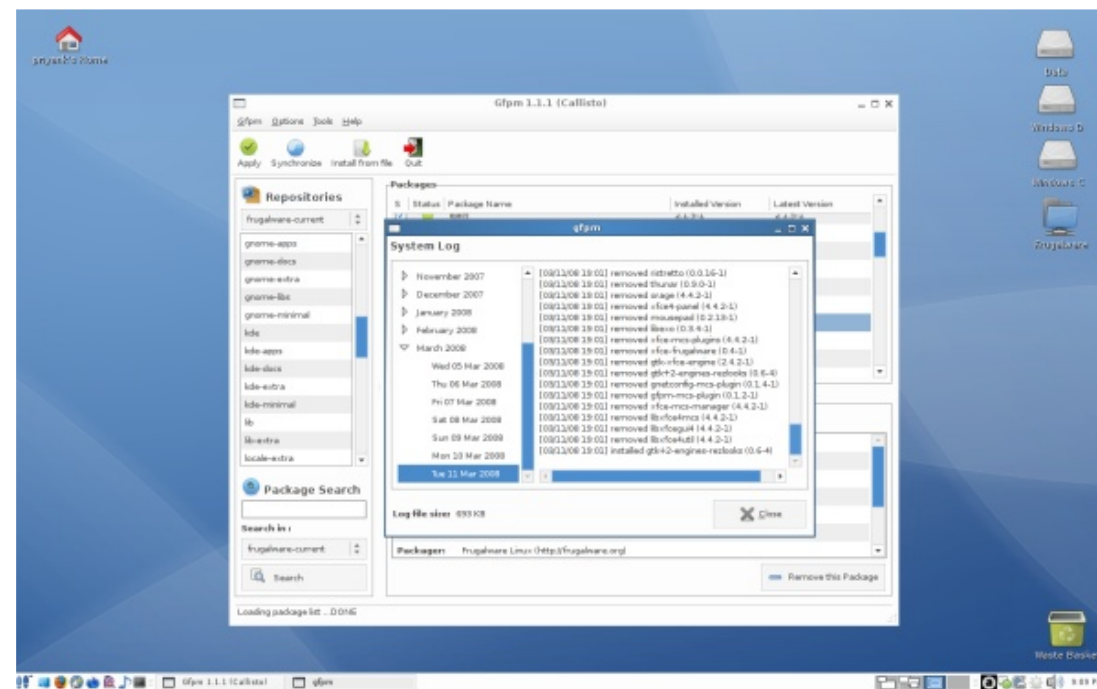
Azt gondolom, hogy néhány extrém esettől eltekintve a 0.8-as – jelenlegi aktuális stabil – kiadásunk az ehhez hasonló igényeknek bőven megfelel.

Ha sikerült felkelteni az érdeklődést az Olvasóban, a fentiekből következően bárkitől szívesen látunk hozzájárulásokat a projekt fejlesztéséhez: legyen az fordítás, tesztelés, programozás, csomagolás, vagy pénzügyi adomány. A csomagolásról ejtenék még néhány keresetlen szót. Ha tüzetesen keresünk, biztosan találni még ma is olyan nyílt forráskódú projektet amit valaki napi szinten használna, de jelenleg nem található meg a rendszerben. Ekkor több lehetőségünk van. Első az az, hogy letöltjük a program forráskódját, feltelepítjük és örülünk. És imádkozunk. Hogy ne legyen benne biztonsági hiba, mivel nyilván nincs időnk arra, hogy minden forrásból feltett projekt honlapját naponta megnézzük és megbizonyosodjunk arról, hogy nem kell-e frissíteni, hogy egy biztonságos

Hivatkozások:

1. <http://frugalware.org/hu>
2. <http://distrowatch.com/>
3. <http://uhulinux.hu>
4. <http://www.blackpanther.hu>
5. http://en.wikipedia.org/wiki/Vendor_lock-in

rendszer boldog tulajdonosai lehessünk. Ez azért annyira nem biztató. Következő lehetőség, hogy a hibajelentő rendszerben nyitunk egy „Feature request”-et, és megkérjük a fejlesztőket, hogy kerüljön be a rendszer által



támogatott csomagok közé kedvenc programunk. Innentől már csak várni kell: egyrészt, hogy ténylegesen valamelyik fejlesztő vállalja a program karbantartását, másrészt, hogy megijöjjön a friss ropogós következő stabil kiadás amelyben már szerepel támogatott programként kívánalmunk tárgya. Az utolsó lehetőség pedig, hogy – ha némi programozói vénával is meg vagyunk áldva – magunk készítjük el a Frugalware csomagot az adott programból és azt elküldjük a fejlesztőknek. Ha sokszor csinálunk ilyet akkor azt fogjuk észrevenni, hogy megkérdezik tőlünk, hogy nem szeretnénk-e fejlesztők lenni, ami elég hízelgő szokott ahhoz lenni, hogy ne utasítsuk vissza. Legalábbis ilyesmire még nem volt példa.

Nincs más hátra mint hogy búcsúzzam, és figyelmükbe ajánlom projektünk weblapját, amin minden egyéb információ megtalálható, a levelezési listáktól kezdve az IRC csatornáig, a hibabejelentőtől a fórumig, a dokumentációtól a biztonsági figyelmeztetőig, és még lehetne sorolni.

A cikkhez tartozó fórum címe:

http://www.flosszine.org/a_frugalware_linux_multja_jelene_es_jovoje

Vajna Miklós,
Frugalware Linux projektvezető

JAKE 2 a függetlenség szobra

Amennyiben egy tetszőleges Linux rendszert játéklatformként kell elemeznem, elkerülhetetlen megemlíteni a Quake vonal néhány tagját. Hiszen John Carmack legendás (és platformfüggetlen) sorozata önálló életében ugyanúgy maradandót alkotott, mint a megannyi más FPS alapkőveként szolgáló id-féle játékmotorok.

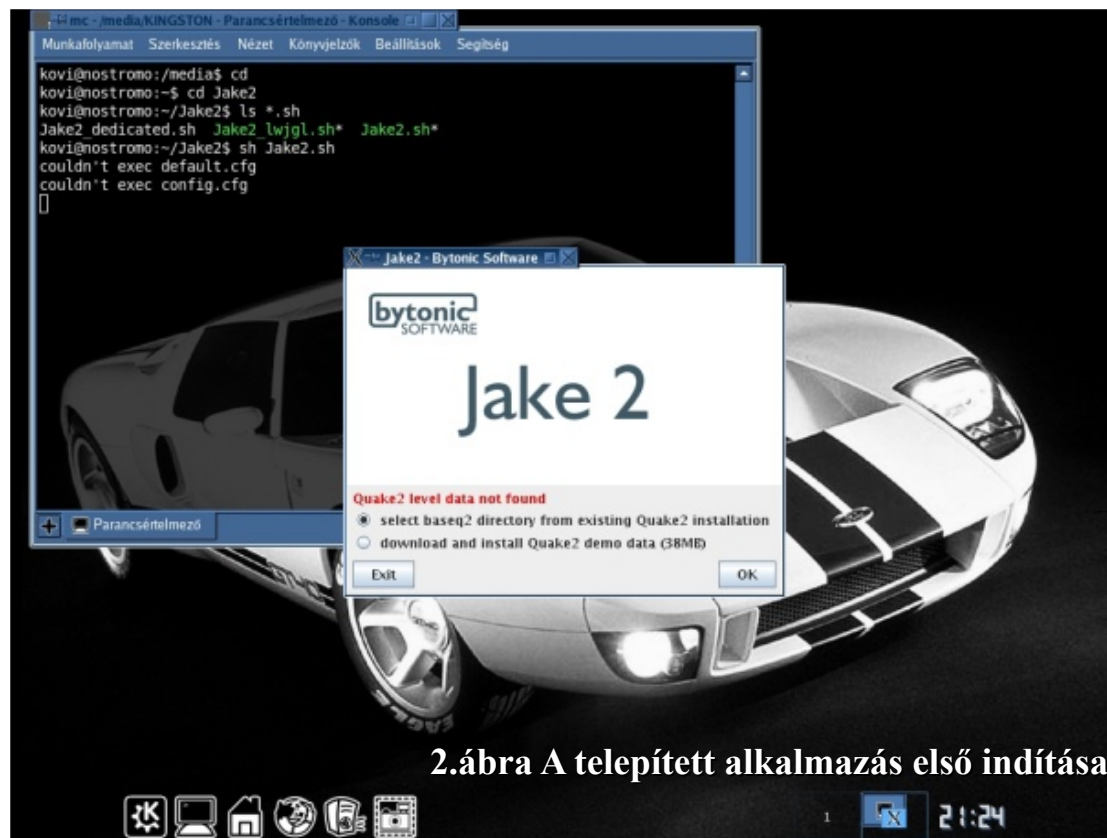
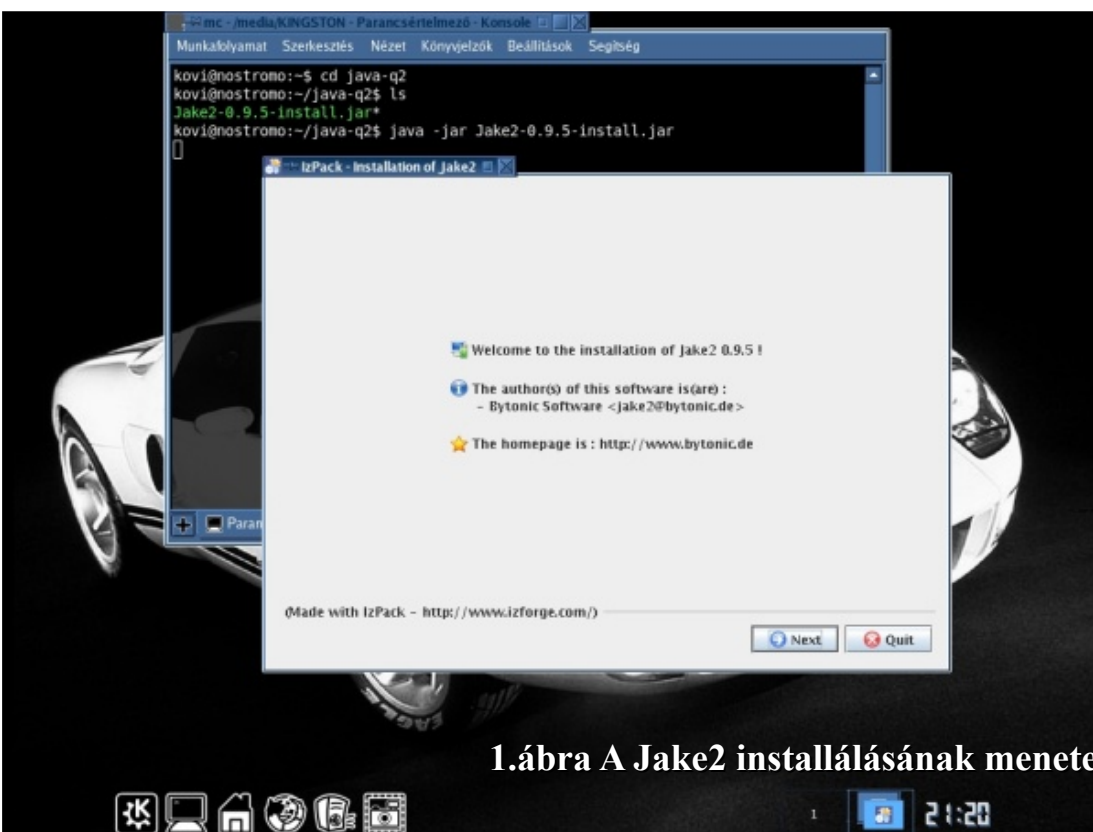
A függetlenség (hó)napja

Amikor egy „lerágott csont” jellegű témát próbálok új kontösben bemutatni, kissé hálátlan feladatra vállalkozom. Hiszen a születendő írásból kisülhet jó is, rossz is: bizonyosan akad, aki kedvét fogja lelteni a „nosztalgiazásban”, de bőven

lesz rá eset, hogy az Olvasó átugorja a témát. Azért mégsem reménytelen a helyzetem, mivel a Quake sorozat második részét egy német csapat nemrég teljes egészében platformfüggetlenre alakította. Ennek megfelelően az átírt kód lehetőségeivel foglalkozom néhány szó erejéig.

Egy vonás, mindenképp felett

PC-k, Mac-ek, PDA-k, okostelefonok. Mi bennük a közös? Furcsa kérdés - de igen, minden különbözőségük ellenére akad néhány közös vonásuk. Csakúgy, mint a Windows, Un*x, Linux, BeOS, OS X rendszereknek is. A mostani cikk szempontjából az a legfontosabb egyezőség, hogy ezeknek az eszközöknek és rendszerszoftvereknek általában meg van a saját lehetőségük arra, hogy Java kódokat futtassanak a megfelelő értelmező jelenlétében. Nem kell zseninek lenni ahhoz, hogy emiatt a Java nyelvet a platformfüggetlenség egyik legfőbb



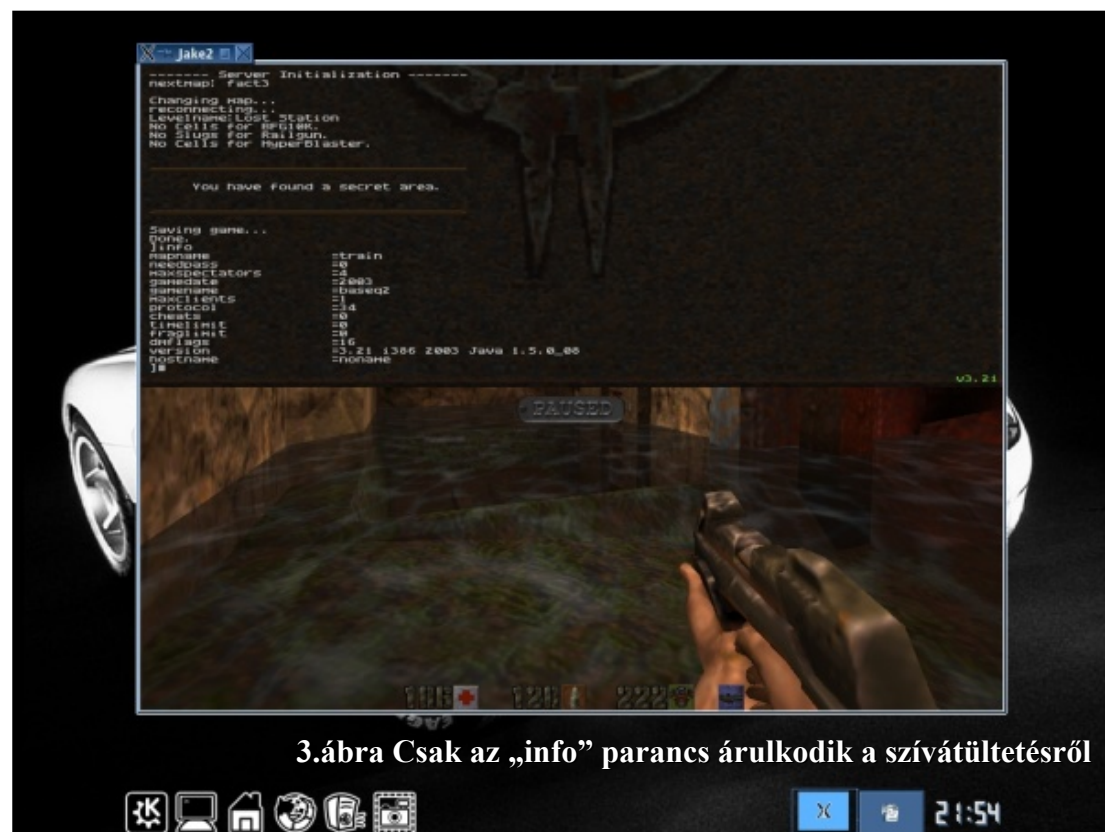
megtestesítőjeként láthassa a végfelhasználó és programozó egyaránt.

Ezen a programnyelven számtalan (kis)alkalmazás íródott az idők folyamán, de a játékszoftverek alkotóit eddig valahogy nem igazán hozta lázba a technika ilyen irányú lehetősége. Hiszen egy nyitott forráskóddal rendelkező játékprogram sokkal inkább úgy költözött egyik platformról a másikra, hogy a forráskódját felkészítették az adott rendszerre, majd az idegen környezetben lefordították. Lényegre törő és életképes megoldás ez, de két-három ponton azért „sántíthat”: minden egyes átültetés újabb és újabb munkát igényel, miközben a portolás végeredménye nem feltétlenül lesz hajszálra azonos és az előző változattal teljesen kompatibilis szoftver. Amellett sem mehetek el szó nélkül, hogy platformként (is) szakértő egyéneknek igen nehéz találni: az olyan szakemberek, akik egy több Mbyte méretű kódot csípőből átírnak mondjuk BeOS-ra, nem teremnek minden bokrban. Aztán ott van például az SDL keresztplatformos megoldása, amely hosszú ideje komoly alapot biztosít a függetlenséget kereső fejlesztőknek, de nem mondhatnám, hogy akár a PDA gépeken bármilyen módon ki lenne használva ez a réteg. Röviden lezárva ezt a fejezetet, a lényegre térek: mára már teljesen kiforrott állapotú a Jake2, ami nem más, mint a Java nyelven átírt Quake2!

A nagybetűs PÉLDA

Az id Software az egyik legfőbb olyan játékfejlesztő csapat, akik minden munkájukhoz készítene néhány alternatív rendszerhez kötődő binárist. John Carmack gárdája gyakorlatilag belekényszeríti a függetlenség e modelljébe a konkurenciát is - így nekünk, Linux felhasználóknak többszörösen is a kedvünkben jár. De miben többek még az id-s srácok, mint a többi piaci résztvevő? Egy tekintetben mindenképp kilógnak átlagból: miután egy aktuális játékuk már nem hordoz számottevő kereskedelmi potenciált, a fő kódját megnyitják a nagyközönség felé. Ezáltal a projektjük rövid időn belül olyan rendszereken is futni fog, amire a fejlesztők eredetileg nem is gondoltak. Nem kell messzire menni a példáért: eredetileg a Quake2 nem volt felkészítve BeOS, OS X (és a jóég tudja még, milyen) rendszerekre. Ehhez képest nehéz olyan környezetet mondanom, ahol mára nem érhető el valamilyen „helyi verziója”, a szakértő programozóknak hála. Talán a telefonokon és tenyérgepeken nem fut,

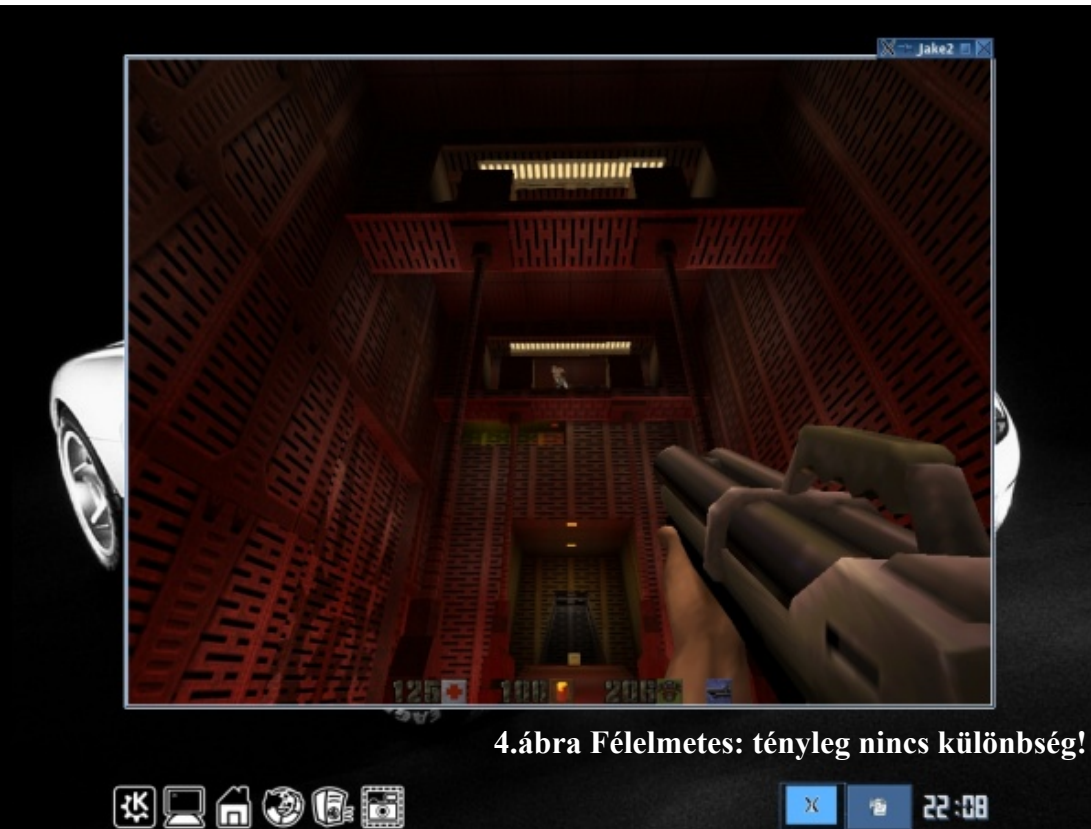
valamint azért biztosan akad egy-két olyan platform is, ahol még nem hallottak ilyen nevű programról. Azonban az utóbbi mondatomat már nyugodtam múlt időbe is temethetem, mivel három programozó a Quake2 megnyitott C alapú kódját átírta Java programnyelvi alapra. Nem merek (nem is tudok)



3.ábra Csak az „info” parancs árulkodik a szívátültetésről

belegondolni abba, mennyi munkát igényelhetett az átalakítás, de minden elismerésem a német illetőségű, Bytonic névre keresztelt triónak. Nézzük, mit remélhetünk az általuk átírt Quake2-től! Először is az eredeti verzióval teljesen kompatibilis szoftvert, aminek futásáról a Sun „felségjelű” Java Runtime Enviornment környezet gondoskodik (talán nehéz elsőre komolyan venni az itt írtakat, de a képi megjelenítés, a hangképzés, a belső világ és a fizika felsorolásából semmi sem mutat különbséget a C-ben írt anyakódhoz képest). Másodszor pedig az egyes félelmeknek és várakozásoknak ellenmondó és kiemelkedő sebességet! Bár nincs különösebb értelme futási teljesítményt mérni (hiszen mára ez egy átlagosnál gyengébb vason is meghaladja a 4-500 képkocka

/ másodperc értéket), de ugyanabban a hardware környezetben a Jake2 képes jóval nagyobb teljesítményt is elérni, mint az eredeti változat...



4.ábra Félelmetes: tényleg nincs különbség!

Felélesztés

Linuxon több módja is van a Quake2 használatba vételének: az id Software és az Icculus anyagaival nem foglalkoznák: ezeket számtalan portálon és fel lehet lelteni. Most aktuálisan, íme a Java - ba öntött technika felélesztése. Hihetetlenül egyszerű dolgunk akad: a Jake2 honlapjáról ([[=>]]<http://www.bytonic.de/html/jake2.html>) töltsük le a Jake2-verzió-install.jar állományt, majd ellenőrizzük, hogy a Linuxon lévő JRE komponens milyen verziószámú (a Jake2 futtatásához legalább v1.4 azonosítójú JRE kell). Ezek után a letöltött fájl mellett állva adjuk ki a java - jar Jake2-verzió-install.jar parancsot (ebben az esetben nem szükséges root jogkör)! Grafikus telepítő fogja köszönteni a felhasználót, mely először a

licenc tudnivalóit olvastatja el, majd a telepítés módjára kérdez rá. Miután minden összetevő a helyére került, az installált alkalmazást a telepítés útján lévő Jake2.sh szkripttel lehetséges elindítani (sh Jake2.sh). Természetesen kérhető a 3D képi megjelenítés is, ekkor a rendszerünkben jelen kell lennie a Lightweight Java Game Library komponensnek (ezt egyébként tartalmazza az előbbieken letöltött telepítő): az OpenGL leképezés használatát sh Jake2_lwjgl.sh parancssal lehet bekapcsolni. Az átírt kód képes dedikált szerver üzemmódra is, ehhez a módhoz a Jake2_dedicated.sh állomány tartozik. Indítás után a szoftver rákérdez az eredeti Quake2 adatcsomagok elérési útjára, ami akár a telepítő CD-n lévő /Install/Data/baseq2 mappa is lehet... Egyszerű, igaz?

Konklúzió

Talán feltűnik, hogy igyekszem a Quake2 születésének, történetének (vagy éppen sajátosságainak) újbóli, sokszori bemutatását mellőzni. Nem véletlenül: aki végigolvasta ezt a rövid írást, az valószínűleg a játékrovat célzott rétegének tagja, nekik ezen a ponton nem tudnék újat mondani (ne feledjük, az eredeti játékprogram idestova 12 éves). Végezetül a porterek munkájának minőségéhez kanyarodnék vissza egy gondolat erejéig. Remélem nem követek el hibát, amikor szerénytelenül kijelentem: a Quake vonal erősen a kezemre áll. Nem kell túlzottan erőlködnöm, hogy a második részt cirka négy óra alatt végigjátsszam a legnehezebb módban - bár ez messze nem rekordidő a kategóriában (hiszen akad, aki mondjuk az első epizód bármely összetett világát teljesíti szűk negyed óra alatt), de én ilyen háttérrel nem látok különbséget a két változat világában és játékelményében sem. Csupán a megfelelő konzolparancs árulkodik a „szívátültetésről”. Nem kérdéses tehát, hogy profi munkáról van szó: engem azonnal levett a lábamról!

Kovács Zsolt

A cikkhez tartozó fórum címe:

http://www.flosszine.org/jake2_a_fuggetlenseg_szobra



NAGY RÓBERT (OpenBSD)

-FLOSSzine: Miért pont OpenBSD?

-Nagy Róbert: Évekkel ezelőtt Linux felhasználóként kezdtem, majd ezután egy rendezvényen megismerkedtem a FreeBSD-vel az egyik ottani rendszergazda által. Ezután otthon ezt a rendszert használtam de nem voltam megelégedve vele, mivel nem nyújtotta azt a kényelmet és egyszerűséget

amit egy operációs rendszertől elvárnék. Ezért került arra a sor, hogy kipróbáljam az OpenBSD rendszert. Azóta is OpenBSD felhasználó vagyok. Mivel hiányoztak olyan dolgok a rendszerből amire nekem szükségem volt, ezért egyre több és több diff-et küldtem be a fejlesztőknek, segítettem a hibák javításában, ezért 2004 áprilisában megkaptam a commit jogomat és azóta is örömmel teszek meg bármit ami segíthet ezen operációs rendszeren, vagy a felhasználóin és fejlesztőin.

-FLOSSzine: Mióta fejlesztesz az OpenBSD égisze alatt és mi tartozik a te felelősségi körödbe?

-Nagy Róbert: 2003 vége fele kapcsolódtam be a fejlesztésbe de, mint ahogy azt az előbb már írtam csak 2004-ben lettem hivatalosan is fejlesztő. Viszonylag elég sok mindennel foglalkozom, de főleg olyan dolgokkal amit örömmel csinállok és érdekel, de természetesen vannak olyan dolgok is amiket valakinek meg kell csinálni. Ilyen például a landisk és amd64 csomagok rendszeres készítése. Ezen kívül a kiadásokhoz készült csomagokat is többek között én készítem elő. A munkáimat meg lehet találni mind a kernelben, userlandben és a ports treeben is. Mindenki azért vállal

felelősséget amit írt, tehát például az általam írt driverek én álám tartoznak és engem hordanak le ha valami rossz, vagy ha nem reagálok egy hibajavításra.

-FLOSSzine: Mit gondolsz arról az állításról, hogy az OpenBSD a földkerekség legbiztonságosabb operációs rendszere?

-Nagy Róbert: Ez relatív. A jelenleg civil szférában elérhető operációs rendszerek közül szerintem igen. Természetesen a hadseregek által fejlesztett operációs rendszerekkel nincs értelme összehasonlítani, mivel ezek célszoftverek nem pedig egy általános felhasználásra kitalált operációs rendszer. Szerintem a mi elveink és fejlesztési módszereink nagyon is hozzátesznek ahhoz, hogy elmondhassuk magunkról azt, hogy ez a világ egyik legbiztonságosabb operációs rendszere. Bizonyítja ezt az, hogy olyan helyeken használják ahol igen nagy mértékben számít a biztonság és a megbízhatóság. Szívesen mondanék neveket, de sajnos ezt nem tehetem meg.

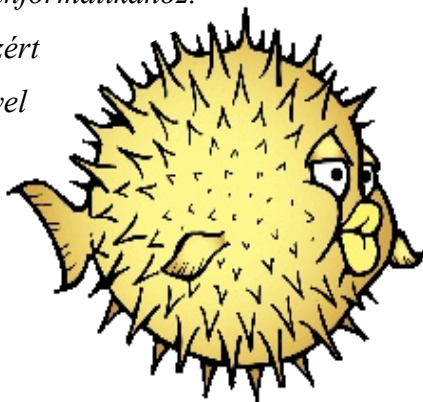
-FLOSSzine: Milyen hangulat uralkodik egy Hackathon-on? Hogyan jellemeznéd? Jól értesültünk, hogy ilyen szervezője már Te is voltál (ráadásul szép hazánkban)?

-Nagy Róbert: Első sorban azért rendezzük meg a hackathonokat, hogy a fejlesztők 1 hétig egy helyen lehessenek, megkönnyítve ezzel a kommunikációt és a közös munkát. Ezen kívül így a legkönnyebb az új fejlesztőket bemutatni a csapatnak és természetesen pár alkohol tartalmú ital elfogyasztása után mindenki feloldódik. Eddig mind a 6 hackathonon nagyon jól érezte magát mindenki és nagyon sok mindent megcsináltunk amit másképp hónapok munkájával tudtunk volna elérni. Budapesten eddig két hackathont szerveztem amelyek ugyanúgy folytak le mint a többi, csak végre rájöttek a fejlesztőtársaim, hogy a magyar konyhánál és lányoknál nincs jobb a világon :)

-FLOSSzine: Mi a véleményed Theo de Raadt-ról, ill. volt már szerencséd személyesen is találkozni vele?

-Nagy Róbert: Természetesen már nagyon sokszor találkozunk és szerintem azon felül, hogy nála jobb projektvezetőt találni nem lehetne egy nagyon okos és széles látókörű ember, akitől nagyon sok mindent lehet tanulni és akire fel lehet nézni. Akárhányszor elmegyünk sörözni vagy vacsorázni mindig ki tudunk kapcsolódni és normális emberek módjára olyan témákról beszélgetni amiknek semmi köze az informatikához.

Szerintem az emberek azért vannak olyan véleménnyel róla, amilyenről vannak, mert egyáltalán nem ismerik, de szerintem ezt Theo is akarja.



OpenBSD

-FLOSSzine: Ha 48 óráig Theo lehetnél, mire fordítanád a rendelkezésedre álló időt?

-Nagy Róbert: Úgy viselkednék Theo-val ahogy ő szokott, csak hogy átérezze :)

-FLOSSzine: Tudsz még aktív magyar OpenBSD fejlesztőkről?

-Nagy Róbert: Sajnos nem. Néha szoktak magyarok küldözgetni diffeket, de ennyi amiről tudok. Illetve az egyik OpenBSD fejlesztő magyar származású. A szülei 1956-ban mentek ki az USA-ba és jelenleg Washington-ban élnek. A neve Mike Erdely, bár mondjuk ez már lerövidített, mivel eredetileg Erdélyi a neve. Sajnos ő már nem beszél magyarul, de amikor a budapesti hackathonra jött, nagyon izgatott volt és csakúgy mint a többiek, imádta a várost.

-FLOSSzine: Szerinted mire van a legnagyobb igény OpenBSD-s körökben? Mi az, ami hiányzik, hogy elterjedtebb, elfogadottabb rendszerré váljon?

-Nagy Róbert: Jelenleg a legnagyobb igény a mostani igényeknek megfelelő SMP és thread támogatásra van szükség. Nekünk nem az a célunk, hogy elterjedtebb vagy elfogadottabb rendszer legyen. Mi ezt az operációs rendszert úgy mond magunknak és a jelenlegi felhasználóinknak fejlesztjük, hobbiból.

Nekünk nem célunk az, hogy piacvezető operációs rendszert adjunk ki a kezeink közül.

-FLOSSzine: Mit javasolnál azon ifjú titánoknak, akik szívesen támogatnák szakértelmükkel a projektet, de valamiért ezt eddig nem tette meg?

-Nagy Róbert: Csak azt tudom javasolni, hogy telepítsék fel és használják, ismerkedjenek meg vele, és látni fogják a saját szemeikkel, hogy miért érdemes ezzel foglalkozni. Természetesen ezt erőltetni nem lehet és nem is kell. Ha valaki szeretné csinálni, senki nem fog az útjába állni. Ez egy életstílus amit más fejlesztői csapatban nem lehet megtapasztalni.

A cikkhez tartozó fórum címe:

http://www.flosszine.org/interju_nagy_roberttel_openbsd

MICKÓ GÁBOR (Hungarian Unix Portal)



-FLOSSzine: Mióta létezik a portál és napi hány olvasóval rendelkezik, ebből hány visszatérő?

-Micskó Gábor: A portál a mai formájában 2001.

augusztus 11., szombat – 14:47 óta létezik. Akkor

postáztam az első bejegyzést (<http://hup.hu/node/2>) az

oldalra. Az oldal korábban is működött már (<http://hup.hu/old/index.html>), de akkor egy kicsit más volt a profilja.

A HUP története elég hosszú. Aki kíváncsi rá, elolvashatja a sztorit az oldal FAQ-jában (<http://hup.hu/faq>). Az oldal napi 15-20 ezer egyedi látogatást tudhat magáénak (a Medián adatai alapján). Az olvasók 61%-a rendszeresen visszatérő olvasó (Google Analytics).

-FLOSSzine: Hányan csináljátok és a civil életben mivel foglalkoztok?

-Micskó Gábor: A HUP mögött ketten dolgozunk aktívan: Nagy Attila (bra) és jómagam. Attila foglalkozik az oldalt futtató hardverrel és operációs rendszerrel, én pedig az oldal alatt dolgozó motorral és magával az oldal karbantartásával ütöm el a szabadidőmet.

Civil életben Attila és én is számítástechnikával foglalkozunk, számítástechnikából élünk. Újabban van egy segítők, aki megpróbálja az oldalt kereskedelmileg is

működőképessé tenni. Ha így vesszük, akkor hárman dolgozunk a HUP mögött. Persze nem szabad megfeledkezni az alkalmi segítőkéről sem, akik például a HUP póló projektben vettek részt, vagy éppen logót rajzoltak, modult írtak a webmotorhoz...

-FLOSSzine: Köztudottan rengeteg szabadidőt fektettél abba, ami mára már kinőtte magát a legnagyobb magyar FLOSS portállá. Érezted már úgy, hogy a befektetett energia kárba veszett és leginkább abbahagynád?

-Micskó Gábor: Komolyan sosem fordult meg a fejemben, hogy befejezzem az oldal karbantartását. Néha elgondokoztam azon, hogy vajon el tudnám-e ütni az időmet jobban, hasznosabban, de rájöttem, hogy a szakma miatt – legalábbis amíg ebben a szakmában tevékenykedek – el kell olvasnom a szaksajtót. Viszont ha már elolvasom, „nem kerül sokba”, ha írok róla néhány sort.

-FLOSSzine: A nemrégiben végbement változásoknak (non-profitról profitorientáltra váltás) milyen a visszhangja? A látogatottsági statisztikák hogyan változtak ezáltal?

-Micskó Gábor: A változtatások előtt sokat hezitáltam. Ez időben kifejezve körülbelül fél évet jelent. A fél év alatt beszéltem számos emberrel, azokkal, akik a kezdetek óta az oldal körül vannak. Elmondtam, hogy mire készülök. Szinte kivétel nélkül arra bátorítottak, hogy induljak el arra, amerre most a HUP tart. Az olvasók is pozitívan fogadták a bejelentést. A többség megértette, hogy a lépés nem az oldal ellen történt, hanem annak fennmaradása mellett.

-FLOSSzine: Tervben vannak-e nagyobb változtatások, amelyek az oldal szerkezetét, tartalmát lényegesen befolyásolhatják?

-Micskó Gábor: Nagyobb változtatások az oldal életében belátható időn belül nem várhatók. Tesszük a dolgunkat úgy, ahogy eddig tettük. Hogy a jövő mit hoz, azt nem tudom. Egy biztos, az oldal marad az elkövetkező években olyannak, amilyenek megszoktuk.



-FLOSSzine: Nemrégiben még be lehetett szerezni HUP.hu -s pólókat (2 színben is). Terveztek ilyesmit a közeljövőben?

-Micskó Gábor: A HUP póló projekt a közösség kérésére jött létre. A közösség kitalálta, hogy lenne ilyesmire igény. Szerencsénkre a Kiskapu akkor felvállalta a projekt összes nyűgjét a tervezéstől kezdve a gyártáson át a kipoztázásig. Nekem erre sem időm, sem energiám nem lett volna. A pólók elfogytak, de komolyabb igény nem jelentkezett azóta rá. Hogy lesz-e a jövőben HUP póló? Ez nyilvánvalóan attól függ, hogy lesz-e rá komolyabb igény. Ha ilyen igény jelentkezik, akkor megfontoljuk a HUP póló projekt „újrátöltését”.

-FLOSSzine: Mit kérnél azon olvasóinktól akik adott esetben a HUP-ot is rendszeresen látogatják (pl. mi szerzők is :), hogy az oldal erősebb közösséggé forrja ki magát?

Elég gyakran eltűnnek érdekes figurák pl. a flame-eknek köszönhetően. Nem gondolod, hogy időnként erősebb moderátori beavatkozás szükségeltetne?

-Micskó Gábor: Igazából nincs kifejezett kérésem. Az oldal „az ami”. Akik szeretik, azok pont azért szeretik, mert „az ami”. Ha változtatnánk rajta, akkor már más lenne, más közönséggel. Nem hiszem, hogy változtatni kellene rajta. Erősebb moderálás? Nem hiszem, hogy szükséges lenne. Az oldal pont attól „működik”, hogy csak néhány szabályt kell betartani (lásd: FAQ) . Persze van, akiknek ez is problémát okoz. Olykor elkél egy kis kézi rásegítés a téma megfelelő kerékvágásba tereléséhez, de ez elég ritka. Szerintem jól van ez így, ahogy jelenleg van :)

A cikkhez tartozó fórum címe:

http://www.flosszine.org/interju_micsko_gaborral_hup

Integrált Fejlesztői Környezetek Linux alatt

Ebben a cikkemben arról szeretnék beszélni, hogy ha valaki programfejlesztésre adja a fejét, akkor Linux alatt milyen lehetőségei vannak. A cikkben a C/C++ fejlesztői környezetekről lesz szó, és azok közül is elsősorban az általam használt és kedvelt fejlesztői környezetekről fogok beszélni. Igazából ez a cikk egy rövid bemutató lesz mindegyikről, részben a terjedelem szűkössége miatt, részben pedig mert szerintem mindegyik környezet önmagában is megérdemelne egy különálló cikket.

Mielőtt rátérnék a fejlesztői környezetek bemutatására, tisztáznék néhány fogalmat. Integrált Fejlesztői Környezet, rövidítve IDE (Integrated development environment). Az IDE olyan alkalmazások gyűjteménye, melynek segítségével további alkalmazásokat hozhatunk létre.

Főbb összetevői:

-Szövegszerkesztő: a forráskód

szerkesztésére szolgál

-Fordító: A forráskód fordítására szolgál

-Fordítási és automatizálási eszközök

-Hibakereső, nyomkövető eszközök

-Hibakereső és nyomkövető (debugger):

Olyan eszköz melynek segítségével a forráskód megtekinthető futás közben.

Töréspontok (berakpoint) segítségével vizsgálható, hogy futás közben melyik

programrészlet hajtódik végre, melyik elágazásba lép a program, és különböző változóknak mi az értékük.

NetBeans

Az első IDE, amit be szeretnék mutatni, a NetBeans. Nagyjából egy évvel ezelőtt kezdem el használni, és meglepően könnyű és gyors volt áttérni rá. Amit tudni kell a NetBeansről, hogy Java-ban írták, tehát kicsit erőforrásigényes. Többfajta nyelvet támogat, elsősorban Java, de külön bővítményekkel támogatja még a PHP-t, C/C++-t, Ruby-t és még többet is. Használják weblapfejlesztéshez, támogatja az AJAX

technológiát. Java alapú mobil alkalmazások fejlesztésére is alkalmas. Valamint UML diagrammok is készíthetők vele. Támogatja a CVS és az SVN verziókövető rendszereket.

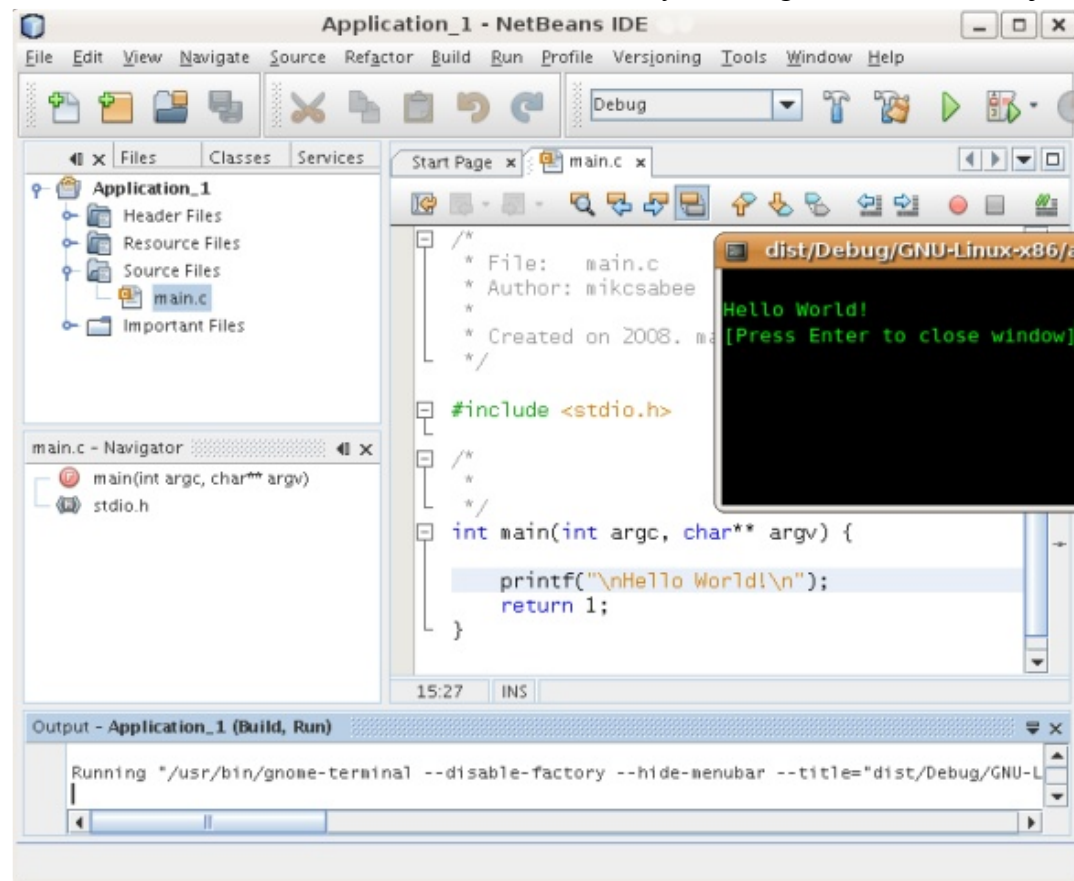
A fejlesztés projektekben zajlik. Minden projekthez tartozik négy külön alkönyvtár amelyekbe külön lehet kategorizálni a fejlécfájlokat, erőforrásokat, forráskódokat és különböző fontosabb állományokat, például makefile-okat.

A NetBeans környezetet talán öt fő részre lehetne osztani.

Felső menüsor: Itt érhető el a NetBeans legtöbb funkciója, a fájl megnyitástól a verziókövetésen át a fordításig szinte minden.

Projekt navigátor: Ez az ablak

alapértelmezetten a menüsor alatt balra található. Itt van lehetőség kezelni az állományokat, megnézni az osztályokat, esetleg különböző szervizszolgáltatásokat



indítani.

Fájl navigátor: Az aktuálisan megnyitott fájlról ad részletes információkat, például a benne található függvények, argumentumlistával.

Szövegszerkesztő.

Kimenet: Alul a fordítási opciók és az esetleges fordítási hibák tekinthetők meg benne.

Eclipse

Az Eclipse eredetileg egy nyílt forráskódú közösség. Ennek a közösségnek a fő projektje az a platformfüggetlen szoftverkeretrendszer, melyet jellemzően különböző IDE-k készítésére használtak fel. Az Eclipse közösségnek több mint 60 nyílt forráskódú projektje van, melyek több csoportba sorolhatóak, ezek közül a legfontosabbak, a Enterprise fejlesztés, beágyazott és mobil eszközök fejlesztése, Internetes alkalmazások, szoftver menedzsment. Maga az Eclipse IDE több nyelvet támogat, melyek közül a legfontosabbak: Java, C/C++, PHP, COBOL. Az Eclipse IDE nemigazán hasonlít a NetBeans felületére megoldásra. A szövegszerkesztő természetesen ugyan úgy megtalálható benne, de Projekt böngészőt kicsit máshogyan képzeltek el, különböző fordítási állományok és makefile-k is menedzselhetők innen. Felület alsó részén található a fordítási kimenet, különböző hibaüzenetek, fordítási opciók, stb. A jobboldali menüben pedig egy kivonatot láthatunk az aktuálisan szerkesztett forráskódokról. Az Eclipse hibakereső lehetőségeit, kimondottan fejlettnak találtam.

Ultimate++

Az Ultimate++ -t keresztplatformos programfejlesztésre tervezték. Ennek

megfelelően rengeteg olyan függvénytarat mellékeltek hozzá ami ezt elősegíti (GUI, SQL, hálózatok). Kizárólag csak a C++ programozási nyelvet támogatja. Az Ultimate++ -ot a 602-es verzió óta használok, és a cikkhez a képernyőképet a 2008.2.beta verzióból készítettem. Mindenkinek, aki ki szeretné próbálni ezt a verziót ajánlom, mert a 2007-es verzióban könnyen bele lehet szaladni néhány

programhibába (ajánlom mindenkinek a figyelmébe a changelog-ot).

A felület felépítése meglehetősen egyszerű. A szövegszerkesztőn kívül bal oldalt fenn egy csomagkezelő található, ahol az előre elkészített, vagy más forrásból származó csomagok tarthatók karban. A baloldali mező alsó részén az aktuális projekt fájllai találhatóak. A szövegszerkesztő alatt fordítás kimenete tekinthető meg.

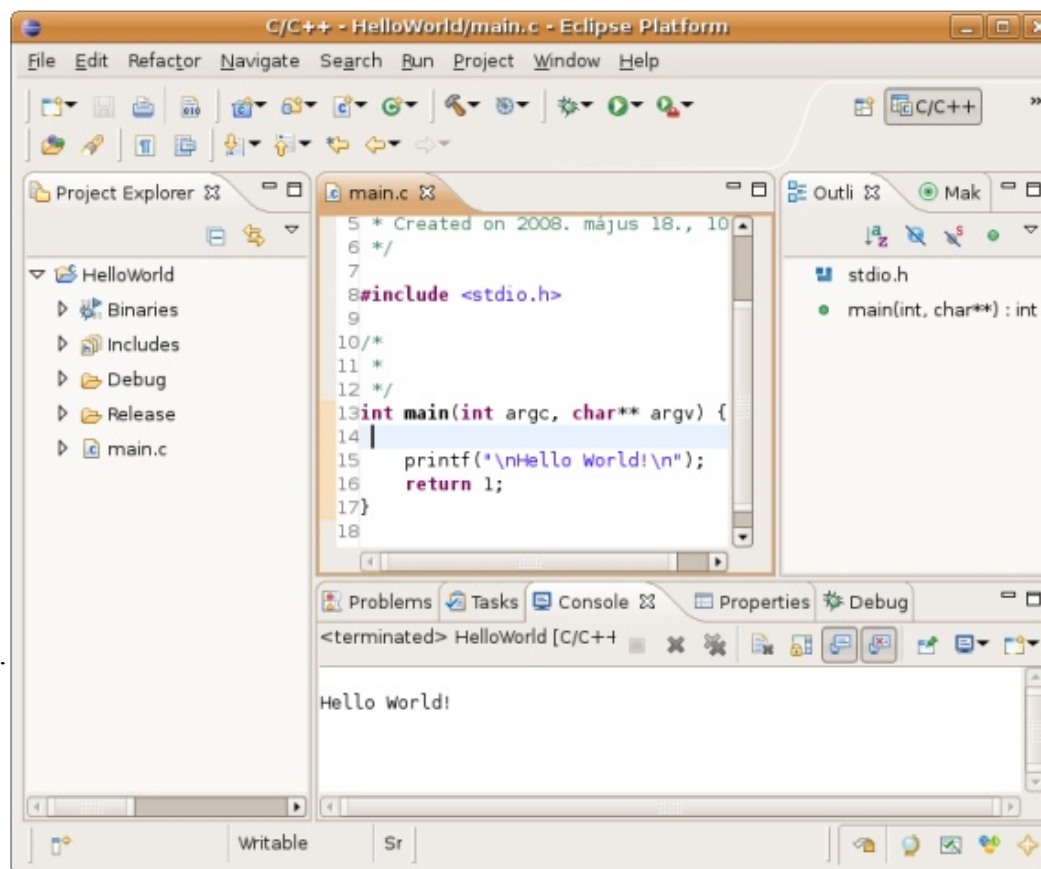
Vélemények

A cikkben megemlített IDE-k nagyon funkció gazdagok, korszerű kódkiegészítést támogató szövegszerkesztőkkel. Mindhárom IDE a GCC-t használja a programfordításhoz Linux alatt. Személy szerint én mostanában a NetBeans-t szoktam használni fejlesztéshez, de amikor keresztplatformos GUI-s alkalmazásokat írok, akkor természetesen visszatérek az Ultimate++ -hoz.

Biztosan sokan furcsállják, hogy kihagytam az EMACS-ot, de bevallom őszintén, hogy azt még sohasem használtam.

Miklós Csaba (Mikcsabee)

2008. május 25.



Hivatkozások:

NetBeans: <http://www.netbeans.org/>

Eclipse: <http://www.eclipse.org/>

Ultimate++: <http://www.ultimatepp.org/>

A cikkhez tartozó fórum címe:

http://www.flosszine.org/integralt_fejlesztői_kornyezetek_linux_alatt

Hello world!

(Programozás Linux környezetben)

Cikksorozatunk bevezető részében megismerkedünk a Linux környezetben történő programozás alapjaival. Különösebb előtanulmányokra nem lesz szükség, mindössze egy működő GNU/Linux operációs rendszer fusson a gépünkön illetőleg nem árt egy kis C programozási alapismeret, bár nem létszükséglet ha valamilyen más programozási nyelven már van minimális gyakorlatunk, legyen az akár a gwbasic is.

1 Az alapok

Miután terveink szerint eme cikkeket a teljesen kezdők is olvasni fogják ezért teljesen az alapokról indulunk. Tapasztaltabb programozó kollégáink se adják fel a reményt, a későbbiekben nekik is lesz sok csemege az anyagok közt, akár többszálú webserver házi készítése is.

Néhány alapvető dolgot az utószóban írok le, például a cd parancs használatát, a TAB-bal való navigációt, a man page-ek használatát, satöbbi. Ha valakinek ezek újdonságot jelentenek, ugorjon a cikk végére, ott megtalálja a szükséges információkat.

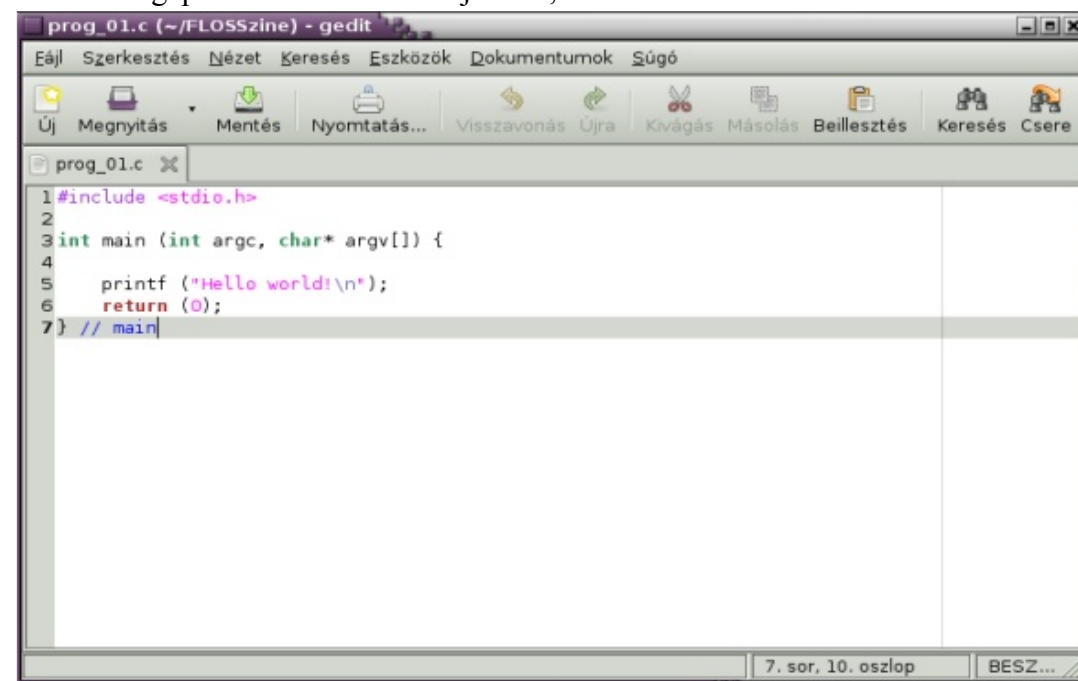
1.1 Az operációs rendszer

Az operációs rendszer feladata nem más, mint hogy a számítógép (azaz a hardver) erőforrásait hatékonyan elossza az alkalmazások (programok) közt és ezeket a folyamatokat egyéb segédprogramok segítségével kezelhetővé tegye az adminisztrátorok illetve a felhasználói számára. Maguk az erőforrások lehetnek

például a billentyűzet, a display (pongyolán fogalmazva a monitor), a hálózat, a nyomtató, a floppy diszk, a merevlemez, stb. Az alkalmazások lehetnek az operációs rendszer részei, de lehetnek felhasználói programok is, mint például az a kis program amit ebben a cikkben fogunk elkészíteni.

Maga az operációs rendszer nem igényel grafikus felületet, az teljesen más kérdés, hogy manapság az összes asztali (azaz desktop) gépre írt operációs rendszer (még a Linux is) szép grafikus felülettel jelentkezik be, egérgattintással intézhetünk szinte mindent, de egy jól felépített operációs rendszer működéséhez nincs szükség grafikus felületre. Egy jó darabig mi is konzolos - azaz szöveges képernyős - alkalmazásokat fogunk írni, megjegyezzük, sok olyan grafikus alkalmazás van amely valójában csak egy burkoló program (GUI azaz grafikus felület, interface) egy konzolos alkalmazáshoz.

Az, hogy melyik Linux disztribúciót válasszuk munkánkhoz egyéni ízlés kérdése, ha már van a gépünkön akkor használjuk azt, ha most ismerkedünk vele akkor személy



```
prog_01.c (~/FLOSSzine) - gedit
Eájl Szerkesztés Nézet Keresés Eszközök Dokumentumok Súgó
Új Megnyitás Mentés Nyomtatás... Visszavonás Újra Kivágás Másolás Beillesztés Keresés Csere
prog_01.c
1 #include <stdio.h>
2
3 int main (int argc, char* argv[]) {
4
5     printf ("Hello world!\n");
6     return (0);
7 } // main
7. sor, 10. oszlop BESZ...
```

szerint (de ez nagyon egyéni ízlés kérdése) az Ubuntut javaslom (ubuntu.com) illetve bátrabbaknak, tapasztaltabbaknak a Debiánt (debian.org). Ha egy adott program telepítésére hivatkozok a cikkben, akkor a debian alapú rendszerekre értendő amit írok, más csomagkezelők esetén az adott rendszer csomagkezelőjével intézzük a telepítést.

1.2 A szükséges programok

Munkánkhoz szükség lesz egy editorra azaz szövegszerkesztőre. Itt kell egy fontos különbségre felhívunk a figyelmet. Ezen esetekben text editorokról, azaz szövegszerkesztőkről beszélünk és nem pedig word processorokról amit a magyarban szintén szövegszerkesztőnek hívunk de valójában nem az. A text editor az szimplán szöveget kezel mindenféle formázások nélkül, a word processor pedig a szöveg formázására helyezi a hangsúlyt, nem szimpla szövegállomány a végeredménye, jellemző képviselőik az OpenOffice.org Writer programja illetve az MS Word a Windows világában. Az editorokra példa a linuxos világban rengeteg van, a Windows környezetben a notepad a képviselőjük.

Szükségünk lesz a C fordítóra, ez a gcc. Általában alaptól minden rendszeren fent van, ha ezt nem tudjuk biztosra akkor amúgy is grafikus felület van előttünk, vagy nyitunk egy terminál ablakot és root-ként beírjuk, hogy apt-get install gnome-devel ami feltelepít minden olyan csomagot is ami a későbbiekben lesz szükséges, vagy a synaptic grafikus csomagkezelőben jelöljük ki és telepítjük.

Nem árt, ha a man page-ek is fent vannak a gépen, apt-get install manpages manpages-dev, így bármikor segítséget kaphatunk a használható parancsokról, függvényekről. Személy szerint letöltöttem a gépemre a teljes libc leírást <http://www.gnu.org/software/libc/manual/>, így egy böngésző ablakban bármikor teljes használati utasítást találok a használni kívánt függvényhez.

Szükségünk lesz egy terminál programra, ezt vagy rögtön a desktopon (asztalon) találjuk meg vagy pedig az Alkalmazások/Kellékek menüpont alatt, feltételezve, hogy grafikus felület van előttünk.

1.3 Egy program futtatása

Annak, hogy azelőtt foglalkozunk egy program futtatásának módjával mielőtt azt egyáltalán megírtuk volna, oka van. A szabályosan installált programokat az operációs rendszer külön könyvtárakba helyezi, ezek pl. a /usr/local/bin, ahova csak root jogokkal lehet bevenni fájlokat viszont benne van a PATH-ban, azaz bárhol (tehát bármely alkönyvtárban) is vagyunk, a program nevének begépelésére azt megkeresi a rendszer és elindítja. Ha mi magunk írunk programot, annak a futtatható állománya ott lesz helyben abban a könyvtárban ahol éppen fordítottuk. Legyen a programunk neve prog1_01. Ha ezt a nevet begépeljük, azt az üzenetet fogjuk kapni a rendszertől, hogy nem létezik ilyen program, holott ott van az orrunk előtt. Biztonsági okokból ugyanis nem lehet futtatni közvetlenül a név begépelésével ilyen helyi programokat, a futtatás helyes módja a ./prog1_01 begépelése, így már el fog indulni.

A UNIX rendszerek nem különösebben foglalkoznak a fájlnevekben található kiterjesztésekkel, túlzottan nem igényli a rendszer a használatukat. Jelen pillanatban ez minket annyiban érint, hogy nem kell .com vagy .exe vagy más kiterjesztést adni a futtatható programok nevéhez, anélkül is elindítja őket a rendszer.

2 A program

Akkor ugorjunk is a közepébe. Indítsuk el kedvenc szövegszerkesztőnket az Alkalmazások/Kellékek menüből a Szövegszerkesztő programot, majd gépeljük be az alábbi picit programot:

```
#include <stdio.h>

int main (int argc, char* argv[]) {

    printf ("Hello world!\n");
    return (0);
} // main
```

Ha ezzel megvagyunk, elmentettük prog_01.c néven, akkor nyitunk egy terminál ablakot, a cd parancs segítségével elbattyogunk abba a könyvtárba ahova ezt a forrásfájlt mentettük akkor gépeljük be a következő parancsot:

```
gcc -Wall -o prog_01 prog_01.c
```

Különösebb üzenetet nem kapunk ha mindent jól csináltunk csak visszakapjuk a kurzort. Esetleg egy ilyen becsúszhat:

```
prog_01.c:7:10: warning: no newline at end of file
melyen könnyen segíthetünk, egy soremelést (ENTER) rakunk az utolsó sor végére.
```

Gépeljük be a programunk nevét:

```
./prog_01
```

nyomjunk ENTER-t és láss csodát:

```
Hello world!
```

Megvolt a sikerélmény, nézzük végig tételesen, hogy vajh mit is csináltunk!

2.1 A fordítás

Fogjuk meg a könnyebbik végén a dolgot, a fordítási parancsnál. A gcc teljesen egyértelmű, ez a fordítóprogram neve. Megadtunk neki néhány paramétert, ezeket nézzük sorra. A -Wall az egy kapcsoló, azaz opciókat adunk meg vele. A W opciók

a Warning-okat állítják, azaz meg tudjuk határozni, hogy a fordítóprogram mely figyelmeztetéseket írja ki üzenet gyanánt. Az all az összeset kiírja. Az -o kapcsolót követnie kell egy fájlnévnek, az output filenévnek, tehát itt azt adjuk meg, hogy milyen néven mentse el a lefordított programot a fordítóprogram, ez lesz a prog_01. Szorgalmi feladatként próbáljuk ki, hogy mit csinál a fordító ezen opció használata nélkül. Az utolsó paraméter a fájlnev, azaz annak a fájlnek a neve amit le szeretnénk fordítani.

Mint látjuk ez valóban roppant egyszerű és fogjuk ezt még egyszerűbbé tenni annak ellenére, hogy a bonyolultabb szerkezetű programok esetén bonyolódik a fordítási parancs is.

2.2 A forráskód

A forráskód első sorai vagy a GPL licence szoktak lenni, vagy a header include-ok. Ahhoz, hogy a fordító elő tudja állítani a tárgykódot, szükséges a programban használt könyvtári függvények header állományait “megmutatni” neki, ez történik az include direktívák használatával. Jelen esetben csak a standard I/O függvénykönyvtárból csemegéztünk, ezt igazából nem is kellett volna meghivatkozni mert automatikusan megkeresi a fordítóprogram, de mi nem slamposkodunk és szépen szabályosan beemeljük.

A C program minden esetben tartalmaz egy main() függvényt, melynek visszatérési típusa int. Figyeljük meg, hogy kilépés előtt 0 (nulla) értékkel térünk vissza. Szintén szokjuk meg, hogy a main függvénynek van visszatérési értéke és azt adjuk is meg a kilépéskor, ugyanis léteznek olyan rendszerek is ahol ennek elmulasztása kellemetlen, rejtett hibákhoz vezethet. A main függvény argumentumlistájával most ne foglalkozzunk, annyit említek meg most róluk, hogy

ezeken keresztül lehet elérni a híváskor adott paraméterlistát.

A program érdemi része az üdvözlőszöveg kiírása, akik nem C nyelven programoztak eddig azok érdeklődéssel nézik a szöveg végén a `\n` vezérlőkaraktert, ez a `newline` jele, azaz sort emel a kiíratott szöveg végén. Magával a `printf()` függvénnyel és rokonaival egy későbbi cikkben bővebben fogunk foglalkozni, rendkívül sokoldalú és hasznos jószág, ám a hibákat ez a függvénycsalád általában `segfault`-tal jutalmazza.

3 Utószó

Ha idáig eljutott a Kedves Olvasó, akkor szerencsésen lefordította és futtatta élete első linuxos programját. Ígéretemhez híven adok pár apró tanácsot ami könnyebbé teszi a munkát.

3.1 A `cd` és `mkdir` parancsok

Neve a `change directory`-ből származik, könyvtárváltásra használjuk. Ha megnyitunk egy terminál ablakot, akkor a saját könyvtárunkba jutunk, általában azt látjuk a terminál ablakban, hogy `felhasználóév@gépnev:~/ $` vagy esetleg `felhasználóév@gépnev:/home/felhasználónév$`. Ez a saját könyvtárunk, a gépnevet és felhasználónevet az operációs rendszer installálásakor adtuk meg. Az első esetben a tilde `~` karakter a HOME (azaz saját) könyvtár jele. Hozzunk létre egy könyvtárat az `mkdir munkakönyvtar` beírásával. Ekkor a `munkakönyvtar` nevű könyvtárat létrehoztuk. Belépni ebbe a `cd munkakönyvtar` paranccsal lehet. Azonban ha hosszú könyvtárneveink vannak akkor érdemes használnunk a Linux parancsértelmezők (shell-ek) egyik hihetetlenül kényelmes szolgáltatását, a TAB-ra történő kiegészítést. Kezdjük el begépelni a `cd munk` parancsot majd itt nyomjuk meg a TAB billentyűt. A shell ki fogja egészíteni nekünk a teljes könyvtárnevet. Értelmszerűen ha az előbb beléptünk már ebbe a

könyvtárba akkor lépünk vissza egy szinttel feljebb a `cd ..` paranccsal, hogy meg is találjuk a `munkakönyvtar`-at mert ha benne vagyunk akkor nem fog menni. Jelzem a `cd` parancs után mindenképp kell a szóköz, azaz a `cd.` parancskiadásra hibaüzenetet kapunk.

3.2 A man page-ek

Ha a man page-eket böngésszük, akkor vagy egy terminál ablakot nyitunk hozzá vagy egy új lapot a meglévőn belül, hogy mindig kéznél legyen. Ha például az `mkdir` parancs használatára vagyunk kíváncsiak, akkor begépeljük, hogy man `mkdir` és megkapjuk a használati utasítást hozzá. Régi linuxos mondás, hogy a kezdő felhasználó ott adja fel az ismerkedést a linuxszal, hogy nem tud kilépni a manpage-ből mert a less szövegnézegetőből egyáltalán nem kézenfekvő módon a `q` megnyomásával (a `quit`-ből származóan) kell kilépni. Ha feltelepítjük a `man2html` csomagot, akkor a webes böngésző ablakában is nézegethetünk manpage-eket elég kényelmesen.

3.3 Az `mc`

A DOS-os időkből (aki dolgozott vele) mindenki emlékszik a híres Norton Commanderre. Nos ennek van klónja, a Midnight Commander, becsületes nevén `mc`. Telepítése a szokásos, root-ként `apt-get install mc`. Használata evidens. Van beépített editora, kisebb programjainkhoz tökéletesen elegendő egyelőre az is, bátran használjuk.

Vomberg István

A cikkhez tartozó fórum címe:

http://www.flosszine.org/hello_world_programozas_linux_kornyezetben

Linksys NSLU2, az energiatakarékos mindenés

Régóta szerettem volna itthon is egy szervert üzembe állítani, amely számos feladatot látna el. Például a fejlesztésekhez szolgálna adatbázissal, CVS vagy SVN háttérrel, kezelné a VoIP hívásaimat, felvenne rádióadásokat, esetleg néha segíthetne a letöltésben, illetve a Wifi routeremhez sem ártana egy Radius szerver. Sok megoldást végigpörgettem gondolatban, mire kikötöttem a Linksys NSLU2-nél.

Az ötlet

Egy non-stop otthoni szervernek illik csendesnek lennie, keveset fogyasztania, viszont nem kell, hogy nagyon erős legyen. Első körben egy kiöregedett Pentium 1-es gépre gondoltam, de az mind méretben, mind fogyasztásban elég jelentős volt, illetve a zajszint sem lett volna egy leányálom. Fogyasztását tekintve több, mint 100 watt lett volna monitor nélkül, ami havi 720 órával számolva, mintegy 72 kilowatt (durván: 2300 forint a villanyszámlából).

A Linksys NSLU2 ehhez képest 10 wattot fogyaszt, de ha még a tápegység veszteségét is beleszámítjuk, akkor se jelentős. Mindezt pedig egy teljesértékű otthoni szervert kapunk, amennyiben pendrive-ot vagy 2.5"-es USB-s merevlemez házát használunk. 3.5"-es ház esetén még annak a tápegysége is hozzáadódik. Ha pendrive-ot használunk, akkor pedig teljesen hangtalan az eszköz. Jó eséllyel tehát havi 3-400 forintért lehet nonstop szerverünk.

[illegible]

Az előzmények

A Linksys az NSLU2-t eredetileg NAS (hálózati tároló) eszköznek szánta, elsősorban kisirodák és otthonok kiszolgálására. Az eszközön két db USB 2.0-ás aljzat és egy 10/100-as hálózati csatló kapott helyet. Az eszköz a szokásos Linksys webes felülettel konfigurálható.

Minthogy azonban az NSLU2 belseje szabványosnak mondható ARM architektúra, így megfelelő operációs rendszert telepítve nemcsak hálózati tárolónk, de sok egyéb más is lehet.

Debiannal például a lehetőségeink határtalanok. Azaz majdnem határtalanok, ugyanis az eszköznek nincs se konzol kimenete, se konzol bemenete, így normál esetben csak hálózaton kommunikálhatunk vele. Persze aki igazán barkács lélek, az készíthet hozzá 4x40 karakteres LCD kijelzőt és a fiókban porosodó USB billentyűzetet is munkára foghatjuk.

Szóba került a barkács lélek: a régebbi (2006. május előtti) Linksys NSLU2-k üzemi frekvenciája csupán 133 Mhz, de némi forrasztással ez 266 Mhz-re emelhető,

míg 2006 május utáni eszközök már gyárilag 266 Mhz-el érkeznek. Az eszköz memóriája 32 Mbyte. Akinek ez, vagy az USB portok száma nem megfelelő (további 3db elérhető), vagy szabványos soros konzolt is szeretne, annak csak annyit mondanék: megoldható, de az eszköz megbontásával ugrik a garancia.



Türelem Linuxos szerverterem

A Debian telepítése erre az eszközre inkább türelem, semmint ügyesség kérdése. Az eszközre többféleképpen lehet az induló image-t felrakni, én az upslug Debian csomagot választottam, mert az tűnt a legegyszerűbbnek. Ez nem csinál mást, minthogy egy recovery módban lévő NSLU2-re feltölt egy firmware-t. Hogy kerül az NSLU2 recovery módba? A készülék hátulján találunk egy reset gombot. Az újabb eszközöknél pár másodpercig nyomva tartva a Power LED hol zöld, hol sárga színben villog. Az ügyesség csupán itt kell, hogy a megfelelő pillanatban engedjük el a reset gombot. Ezután már csak az upslug-ot futtatjuk, amely a gyári firmware helyére beírja a Debian telepítő firmware-t. Ha ez megtörtént, akkor az eszköz újraindul. Ekkor csatlakoztassuk gyorsan az USB-s eszközünket – USB-s merevlemezkeret vagy min. 2 gigabájtos pendrive. Ha az NSLU2 sikeresen elindult, akkor lépünk be ssh-val. Ehhez tudnunk kell az eszköz IP címét. Ha dhcp-t használtunk, akkor némileg nehezebb a dolgunk, de egy ping -b és az eszköz MAC címe segít. Ha korábban a doboz fix IP-vel rendelkezett, úgy első körben azon az IP-n keressük. Ha gyári beállításokkal rendelkező NSLU2-t telepítünk, akkor a 192.168.1.77-es IP-n keressük. A felhasználónév installer, a jelszó pedig install. Innentől teljesen szabványos Debian telepítővel van dolgunk. Minthogy azonban lowmem módban vagyunk, így a modulok összeválogatásakor az ext3-at, a partman-ext3-at és az USB storage-t ajánlott betölteni. Ha mindent jól csináltunk, akkor particionáláskor már látszik az USB-s eszközünk. Fontos, hogy minimum 256 megabájt swap területet hozunk létre a kicsi memória miatt és minimum 1 gigabájt háttértár ajánlott.

Ezután már tényleg nincs sok hátra. A tasksel-ben válasszuk ki, hogy miket szeretnénk használni. Később persze azt apt-al is telepíthetünk.

Ha a csomagok elkezdtek csorogni a netről, nyugodtan nézzünk valami hasznos elfoglaltság után. Például ebédeljünk meg, vagy vigyük le a kutyát sétálni, mert az alaprendszer telepítése minimum 1 óra. A telepítés végén a telepítő a végleges kernelt ismét beleírja a flashbe. Fontos momentum, hogy a gyári – Linksys - flash visszaállítható. Ehhez a korábban már említett upslug-ra és a Linksys honlapjáról letölthető firmware-re van szükség. Fontos: a firmware frissítéshez ne használjuk a Linksys frissítő programját.

Egy új rendszer született

A rendszer a flash memória frissítése után újraindul és pár perc múlva be is tudunk lépni ssh-n. Innen már elkezdhetjük testre szabni: fix ip, apt-get, stb. Néhány egyszerű programot lefuttatva -lspci, lsusb, lshw - hamar kiderül, hogy a kűtyü

egyszerű, mint egy fakocka, de mégis sok mindenre jó.

Az USB-s rack sebességét nézve az eszköz 2-3 megabyte/másodpercet tud, függetlenül attól, hogy milyen merevlemez van benne. Továbbá jó tudni, hogy ne akarjunk rajta ki-betömöríteni, illetve programot fordítani. Amennyiben mégis, úgy a screen mindenképp kötelező alkalmazás.

Megérte?

Hogy mire elég ez a kis vas? Nálam elég sokmindenre: jelenleg fut rajta: syslog-ng (ez főleg a Linksys SPA941 miatt), ssh, samba, cups, bind, apache, mysql, postgresql, asterisk. Ezentúl néha megkérem, hogy streamripperrel rögzítsen netes rádióadásokat. Az eszköz terheltsége szinte nulla, csupán csak akkor lódul meg, ha például az asterisk-nek egyszerre 4 vonalon kell üzenetet rögzítenie. Az is kiderült, hogy ha az asterisknek nem kell hangkodeket átkódolnia, akkor kb. 10 szimultán beszélgetést is el tud vinni. Érdekes ötlet, hogy a VoIP szolgáltatótól kapott számomat kívülről tárcsázva utasíthatom az NSLU2-t arra, hogy rögzítse a kedvenc rádióm (radioseven.se) adását a már korábban említett streamripperrel.

Legyen szó akármilyen terhelésről, az NSLU2 alig észrevehetően melegszik. Amennyiben flash alapú háttértárral használjuk, úgy hangtalan is, de ebben az esetben az ext3 helyett célszerű flash-hez tervezett fájlrendszert használni, illetve a naplózást kikapcsolni. Fontos tudni továbbá, hogy nálam kb. 54 megabájt memóriát igényel az eszköz, vagyis swap mindenképp szükséges. Ezért nem ajánlom teljes mértékben a flash-es megoldást.

Mire lehet használni ezen kívül?

Lehetne például netrádióklienst faragni, ehhez csupán egy USB-s hangkártyára van szükség. Lehet Bluetooth vagy Wifi accesspointot fabrikálni belőle. Amennyiben nem szükséges 10-20 megabitnél gyorsabb hálózati kapcsolat, úgy akár kis tűzfal is lehet belőle egy plusz USB-s hálózati csatolóval. De nem ragoznám a lehetőségeket, mert előttem már nagyon sokan sok ötlettel álltak elő az NSLU2-Linux honlapon.

Bővebb információ:

<http://www.nslu2-linux.org/>

E-Medve

A cikkhez tartozó fórum címe:

http://www.flosszine.org/linksys_nslu2_az_energiatakarekos_mindenes

A GNU Screen terminálkezelő program

Bevezetés

A screen egyfajta ablakkezelő, ami a szöveges terminálon, konzolon képes további shelleteket és az azokban indított programokat párhuzamosan, "ablakokban" futtatni úgy, hogy a kapcsolat megszakadása esetén azok nemcsak hogy meg sem szakadnak, de az újrapcsolódás után ismét használhatóak lesznek. Ezzel alapvetően két problémát old meg: megvéd(het)i a fontos munkát a hálózati kapcsolat megszakadásától vagy akár az ablak véletlen bezárásától, illetve lehetővé teszi folyamatosan futó programok több helyről, akár egy időben több különböző személy általi elérését. És természetesen tökéletesen alkalmas időigényes fordítási feladatok "magukra hagyására" is, illetve olyan esetekre, mikor a program hagyományos módon történő háttérbe küldése (backgrounding) technikai vagy hibakeresési okokból nem nyújt kielégítő megoldást. A mindennapokban a következő feladatokra használom:

- otthon: konzolos irc kliens (irssi) és torrent kliens futtatása, szoftverek fordítása, folytonos irc/egyéb IM kapcsolat a desktop gépek állapotától, tartózkodási helyemtől és operációs

rendszerektől függetlenül

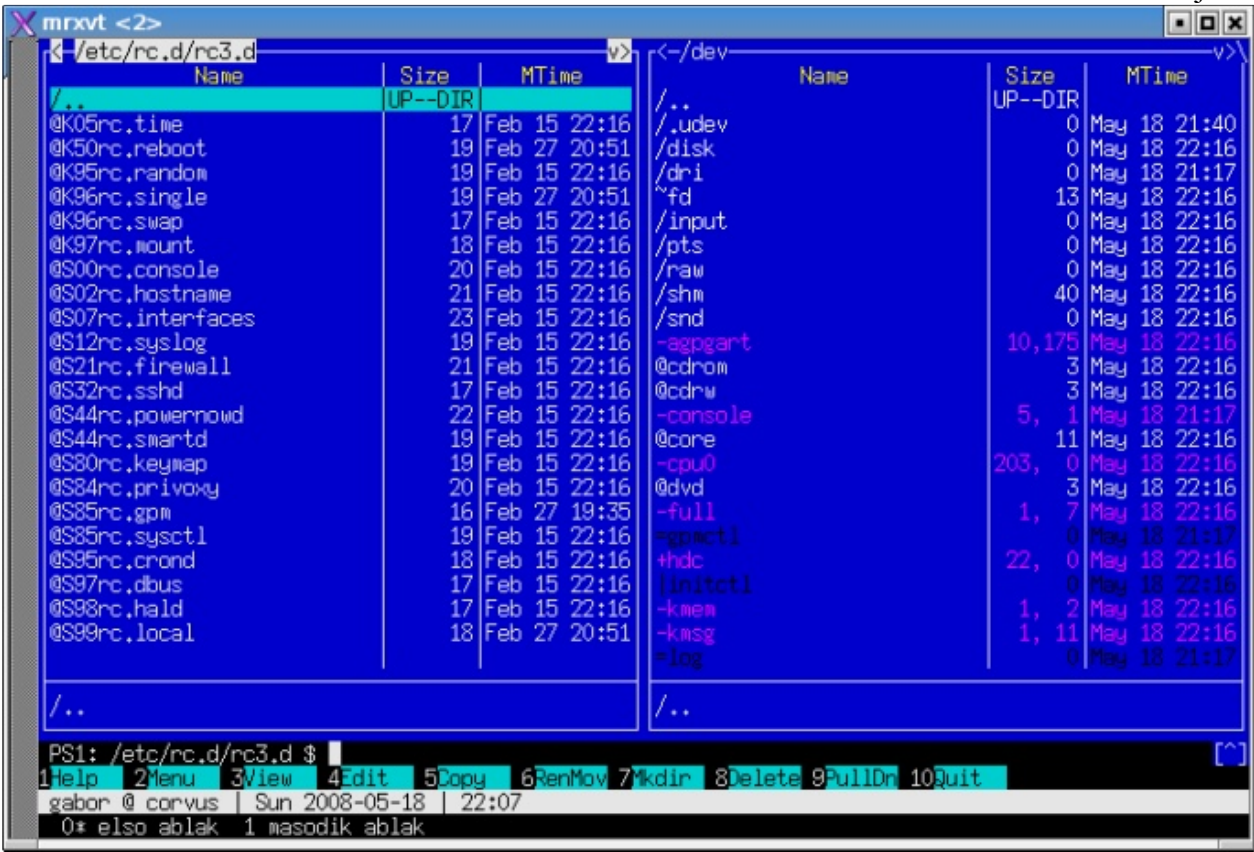
- munkában: tetszőleges számú shell session egyidejű kezelése, a helyi hálózattal és a lokális géppel kapcsolatos problémák kivédése kritikus feladatok végzése közben (pl. Unix rendszerfrissítések), gyors ablakváltás, a "miért nincsenek tabok a putty-ban" dilemma feloldása, futó processzek hibakeresése.

Ez a több mint 20 éves, GPL licenc alatt terjesztett, közkedvelt program meglehetősen kiforrott kódnak számít. A kiadások már kevés új funkciót tartalmaznak (a legfrissebb 4.0.3-as verziót 2 éve adták ki, szintén 2 évvel a 4.0.2 után). A vi editorhoz hasonlóan a screen "tanulási görbéje" is fordított exponenciális trendet mutat: ha egyszer sikerült a legtöbb opciót ízlés szerint beállítani, többet nemigen kell vele bajlódni - csak ne felejsük el jól elmenteni a konfigurációs fájlt. Ezzel az írással szeretném bemutatni a saját tapasztalataimat, illetve a

rendszeradminisztráció során a leginkább hasznosnak talált trükköket. Javaslom, hogy a cikk olvasása közben rögtön próbáljátok is ki a leírtakat, hogy testközeli legyen az élmény! A cikk végén leírtam az első lépéseket, amelyekkel megismerhetitek a használathoz szükséges legalapvetőbb műveleteket.

A screen fejlesztése

A projektet 1987-ben indította a berlini egyetemen dolgozó Oliver Laumann, a kód akkor még MIT-szerű licenc mellett volt elérhető. A



kiadásokat 1991 óta Jürgen Weigert és Michael Schröder, az erlangeni egyetem munkatársai már GPL v2 licensszel tartják karban, integrálva Laumann és egyidejűleg forkolva Wayne Davison hasonló projektjét.

A screen 2003 óta az FSF GNU Project, illetve a GNU Savannah szoftverfejlesztő adatbázis része.

A kiadások tarball formájában, ftp-n jelennek meg; a CVS fa egy ideje szintén publikus. Két levelezőlista is rendelkezésre áll.

Operációs rendszerek

Természetesen az összes valódi szabad Unix jellegű operációs rendszerben használhatjuk a screen-t, így a Linux disztribúciók és a BSD változatok egyaránt kínálnak bináris csomagot belőle. A kereskedelmi Unixokon részben hivatalosan is támogatott a bináris telepítés, például az AIX-hez az IBM bináris és forrás-RPM-eket is letölthetővé tesz, igaz, csupán egy jóval régebbi verziót. Szükség esetén gcc-vel is gyerekjáték lefordítani a screen-t.

A screen Cygwin segítségével Windows alatt is használható, bár vannak ismert problémái.

Maga a lefordított állomány csupán egy binárist, mellette egy man oldalt és néhány konfigurációs fájl tartalmaz. A screen telepíthető userként is, de a megfelelő biztonsági szint és jó néhány ehhez kapcsolódó funkció csak akkor érhető el, ha rootként, "globálisan" telepítjük.

A screen alapértelmezésben az autoconf-ot, a gcc-t és a GNU make-et szeretné használni a fordításhoz. A megjelenítéshez az ncurses library szükséges, emellett opcionálisan a PAM-ot is igényli függőségként.

Beállítások

Globális konfigurációs fájl: `/etc/screenrc`

User-szintű konfigurációs fájl: `$HOME/.screenrc`

Parancssori paraméterek

Online parancsok: `Ctrl-a:[paraméter] [opció] [[érték]]`

A screen session-kezelése

Az egyes munkamenetek kezelése, csatlakozás es leválasztás

A manuál részleteiből csak a leghasznosabbakat kiragadva:

`screen vi foobar.txt` screen indítása, az első ablakban egy vi editorral megnyitva a "foobar.txt" nevű fájlt

`screen -ls` listázza az adott környezetben elérhető screen-eket és állapotukat (csatlakoztatott, leválasztott vagy döglött)

`screen -s valami` létrehozza az új screen-t a megadott névvel (itt: 'valami') - felismerhetővé teszi a sessiont

`screen -x valami` kapcsolódik a megadott screen-hez

`screen -DR` ha van screen session, kapcsolódik hozzá, ha nincs, létrehoz egyet

`screen -m -d` screen indítása a rendszerrel együtt

A screen session neve

A screen session nevére hivatkozva kapcsolódhatunk hozzá, de mint az '-ls' opció kimenetében látható, a screen voltaképpen a PID-et is hozzáfűzi a névhez, így nem keletkezik konfliktus egyazon név ismételt megadásakor sem, sőt kapcsolódáskor a screen jelzi, ha a név önmagában nem egyértelműsíti a munkamenetet!

Multiuser mód, jelszavas védelem, ACL-ek

A screen alkalmas, sőt véleményem szerint kifejezetten ajánlott többfelhasználós üzemre, ugyanis - feltéve, hogy a munka menete előre megbeszélt szabályok szerint halad ("ki üti le az Entert"), - így tetszőleges számú felhasználó közvetlenül követheti az eseményeket. Ez remek lehetőséget ad oktatásra is. A munkamenetet elsődlegesen a multiuser paranccsal tehetjük hozzáférhetővé, a finomhangolás - a fájlok jogosultságaihoz hasonló rendszerben - az `acladd`, `aclchg` és `acldel` parancsokkal történik.

Munkamenetünket a 'Ctrl-a x' billentyűkombinációval zárolhatjuk, de ez szinte semmi védelmet nem ad, tulajdonképpen csak leválasztja és megadott (alkalmi) jelszóval védi a screen-t az aktuális felületen, melyhez lokálisan bárhonnan (pl. másik terminálból) kapcsolódhatunk. Egyetlen esetet tudok elképzelni, mikor tényleges védelmet jelenthet: ha a gépnek csak szöveges konzolja van és más nem ismeri a lokális jelszavakat.

Az ablak több részre osztása (split)

Csak függőlegesen lehet osztani az ablakot. Az újonnan létrehozott területen külön el kell indítani egy programot (shellt), enélkül inaktív.

Ctrl-a s létrehoz egy új területet (kettéosztja az ablakot), amiben még nem fut semmi, shell sem!

Ctrl-a Tab vált az osztott ablak részei között

Ctrl-a c létrehoz egy új shellt az üres területen

Ctrl-a x bezárja az adott területet az osztott ablakban

Ctrl-a Q minden más területet (osztott ablakrészt) bezár az adott ablakban

A legfontosabb screen billentyűkombinációk

A default értékek - értelemszerűen a teljesség igénye nélkül...

Ctrl-a c create új ablak nyitása

Ctrl-a n next következő ablak

Ctrl-a p previous előző ablak

Ctrl-a k kill ablak "elpusztítása" (pl. nem reagáló ssh session esetén)

Ctrl-a d detach lekapcsolódás a screen-ről

Ctrl-a A ablak átnevezése

Ctrl-a w windows ablaklista megjelenítése

Ctrl-a l terminál újrarajzolása (mint a Ctrl-l a shellben)

Ctrl-a : parancsbeviteli mód

Ctrl-a ? help help; opciók megjelenítése (nagyon jól tud jönni!)

A legfontosabb ajánlott screenrc beállítások

Sorvégi kommentek használhatók.

autodetach on automatikus lecsatlakozás a munkamenet megszakadásakor
defscrollback 1024 az ablakonkénti visszagörgethető puffer 1024 sorra állítása
caption always " \$USER @ \$HOSTNAME | %D %Y-%m-%d | %c" 1.) alul állandó sáv megjelenítése login, gépnév, dátum és idő kijelzésével

hardstatus alwayslastline " %w" 2.) az előző sor alatt az aktív ablakok neveinek megjelenítése

screen -t ablak1 0 links www.netbsd.org indításkor a megadott című (itt "ablak1") ablak létrehozása a megadott sorszámmal, ebben egy links futtatása

Tippek, trükkök

- escape átdefiniálása (hasznos lehet például, ha screen-ben akarunk screen-t futtatni):

~/ .screenrc: escape ^Oo # itt az A betűt cseréltem ki O-ra

vagy

screen -e ^Oo

vagy

Ctrl-a :escape ^Oo

- scrollbar puffer elérése:

Esc (belépés a pufferbe); PageUp/PageDown (gördítés); Esc (kilépés))

- ssh-val egyenesen a másik gépen futó screen-be:

ssh user@host 'screen -DR'

- shell beállításai:

Ha a screen-be lépve a shellre vonatkozó felhasználói beállításaink nem működnek, az ENV változóval kell hivatkozni a profilra ('set ENV ~/.shellrc'), amiben a változókat, alias-okat stb. tároljuk.

- screen-ben vagyunk-e:

Leggyorsabban a 'Ctrl-a w' segítségével tudjuk megállapítani, illetve ha például scriptből szeretnénk tesztelni, alapértelmezésben a screen beállítja \$STY shell változót, mely az indító screen processz PID-jét, TTY-ját és a hostnevet tartalmazza.

Screen-ben használható programok

- Shell (bash, ksh, ssh, telnet...)
- IM, chat (irssi, bitchx, bitlbee...)
- Fájltvitel (ncftp, lftp...)
- Levelezés (mutt, pine...)
- Böngészés (links, elinks, lynx, w3m...)
- Fájlcserélés (rtorrent...)
- Fordítók, fájlkezelők és -szerkesztők, médialejátszók...

... egyszerűen minden konzolos program, amit nem lehet vagy nem célszerű a háttérben daemon módban használni.

Alternatív megoldások

DTach

Pusztán annyi funkcionalitást hordoz, amellyel egy adott munkamenetet a benne futó programmal magára hagyhatunk, majd újra csatlakozhatunk hozzá. Fapados egyszerűsége miatt praktikus lehet abban az esetben, ha a screen sokoldalúságára nincs szükség, kizárólag a leválaszthatóság a kíváncsi.

Hivatkozások:

Hivatalos screen oldal [<http://www.gnu.org/software/screen/>]

GNU Screen - Wiki [<http://aperiodic.net/screen/>]

Screen Cygwin alatt [<http://cygwin.com/packages/screen/>]

Abram Hindle: GNU Screen [<http://churchturing.org/w/screen/>]

DTach [<http://dtach.sourceforge.net/>]

A Wikipedia VNC cikke [<http://en.wikipedia.org/wiki/Vnc>]

Grafikus megoldások

A számtalan grafikus lehetőség közül a VNC-t (x11vnc, RealVNC, TightVNC, UltraVNC...) emelném ki, mivel ez egy elterjedt, többplatformos szoftver, aminek van nyílt forrású verziója. Természetesen ez egy teljesen más megközelítés, kliens-szerver modellre épül, saját portokat használ és számottevő hatása van a az erőforrásokra és a hálózati forgalomra is. Különösen alkalmas grafikus munkaállomások vagy például Windows szerverek távoli adminisztrációjára. Unix platformokon az X sok más lehetőséget is kínál.

Egy képzeletbeli screen session

`screen -ls` Ellenőrizzük, hogy fut-e screen az adott gépen

`screen -S myscreen` Létrehozunk a saját screen-ünket 'myscreen' névvel

`Ctrl-a d` Leválunk (detach) a screen-ről

`screen -ls` Láthatjuk, hogy megjelent a 'myscreen' session, a név előtt a futó processz számával

`screen -x myscreen` Ismét belépünk a korábban otthagyott screenbe, névvel hivatkozva rá.

`Ctrl-a c` Nyitunk egy második ablakot.

`Ctrl-a A A` megjelenő promptban átírjuk az aktuális ablak nevét.

`Ctrl-a w` Listázzuk a meglévő ablakainkat - * jelöli azt, amelyikben éppen dolgozunk

`Ctrl-a n` Átlépünk a következő ablakba.

`Ctrl-a DD` Kilépünk az összes ablak egyidejű bezárásával ('power detach').

Lőcsei Gábor

A cikkhez tartozó fórum címe:

http://www.flosszine.org/a_gnu_screen_terminalkezelő_program

FreeDOS

Ebben a mai modern (sőt szép, új) világban még mindig vannak olyan elvetemültek, akik képesek DOS-t használni. Az ő tevékenységüket hívatott segíteni a FreeDOS projekt (freedos.org).

A kereskedelmi DOS-okat már régóta nem fejlesztik, ugyanakkor nagyon sok helyen használnak még régi gépeket, régi DOS-os programokkal.

A gépek pusztulnak, a felhasználói igények fennmaradnak, az új gépeken a régi DOS nem, vagy csak sok kompromisszum árán futtatható.

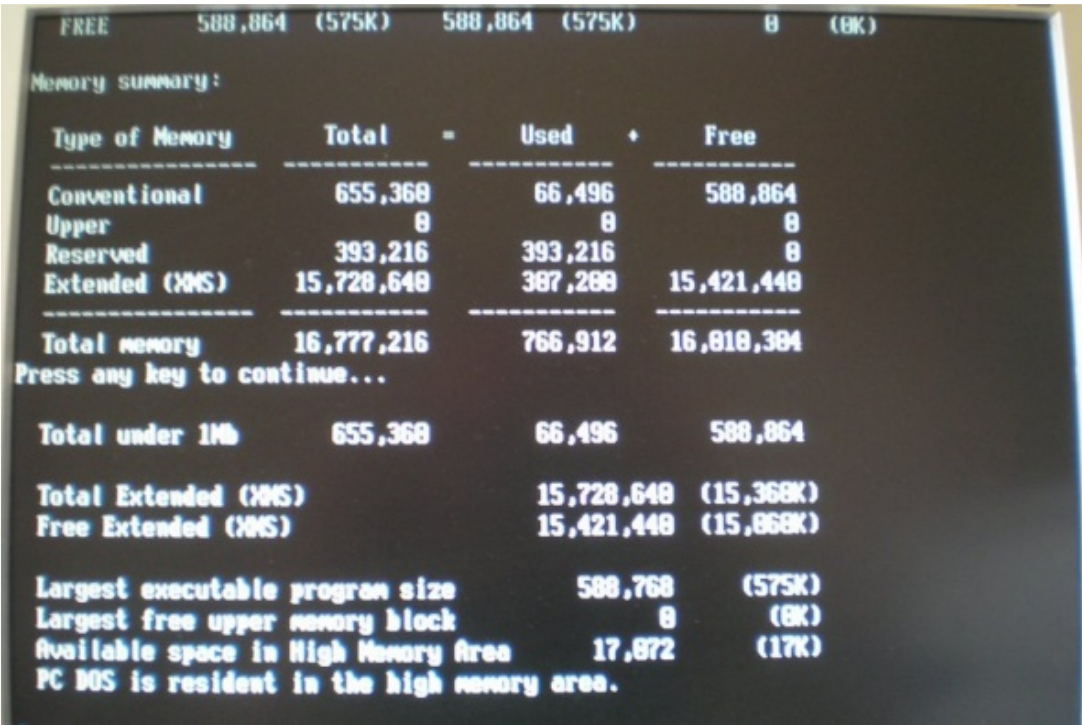
Az 1.0-ás verziószámú FreeDOS sok új márkás gépre felkerült (HP, DELL, stb), hogy később mi lett velük, az más kérdés.

Jelenleg a FreeDOS.org oldalról letölthető a legutolsó verzió, ezt vesszük szemügyre (nálunk is nagyon sok gépen futnak még DOS-os programok, így már többször nézegettük a régebbi verziókat is).

Korábban, főleg Novell környezetben, Clipperben írt alkalmazásokkal sokat kellett variálni, hogy egyes programok rendesen működjenek, néha túl kevés memória maradt a 640K-s alsó területen. A korábbi FreeDOS-oknál ez jelentette a leginkább problémás területet, túl nagy volt a rendszer által elfoglalt memória, így nem maradt elegendő hely a csodálatosan megalkotott professzionális kartotékcímke nyilvántartó programokat nyilvántartó rendszernek. Mára ez megváltozott, ma már sokkal jobb a helyzet.

A képernyő(! :)képeken egy kis összehasonlítás a memóriefoglaltsággal kapcsolatban (PC-DOS és FreeDOS). Természetesen az összetevőkkel sokat lehet játszózni, de nézzük az alaphelyzetet.

Mint látjuk nincs nagy különbség, mind a kettővel már el lehet boldogulni, és egy kis finomhangolással mindkét rendszernél szert lehetne tenni még egy kis többletmemóriára.



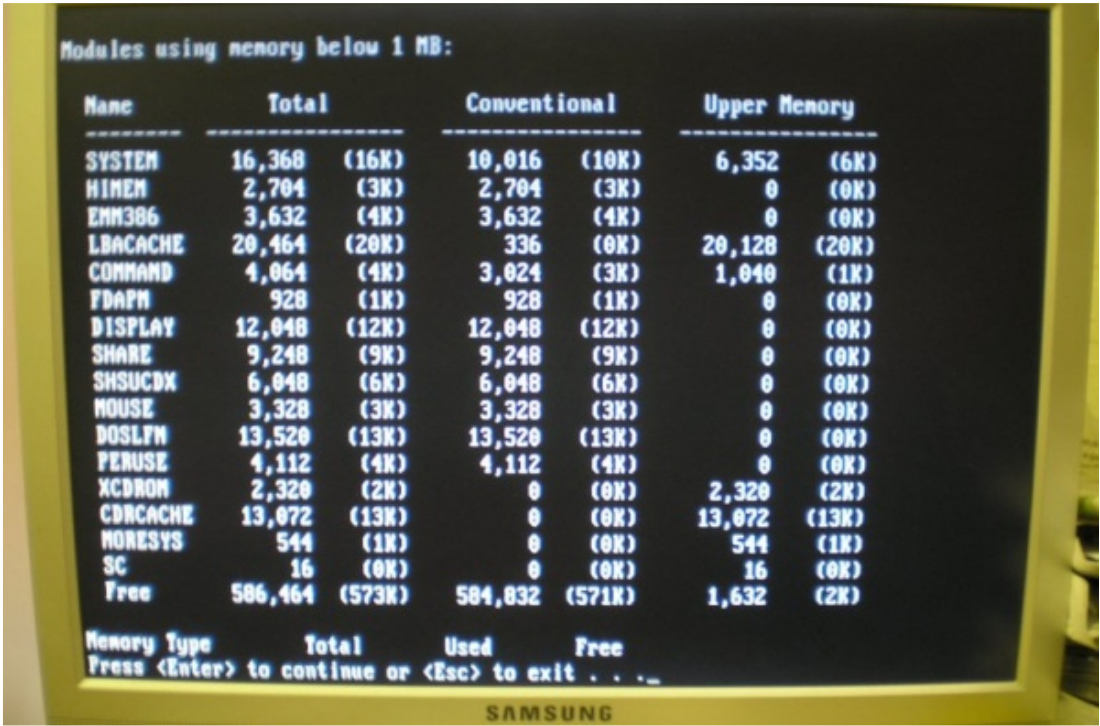
FREE 588,864 (575K) 588,864 (575K) 0 (0K)

Memory summary:

Type of Memory	Total	Used	Free
Conventional	655,368	66,496	588,864
Upper	0	0	0
Reserved	393,216	393,216	0
Extended (XMS)	15,728,648	387,288	15,421,440
Total memory	16,777,216	766,912	16,010,304

Press any key to continue...

Total under 1Mb	655,368	66,496	588,864
Total Extended (XMS)		15,728,648	(15,368K)
Free Extended (XMS)		15,421,440	(15,068K)
Largest executable program size		588,768	(575K)
Largest free upper memory block		0	(0K)
Available space in High Memory Area		17,872	(17K)
PC DOS is resident in the high memory area.			



Modules using memory below 1 MB:

Name	Total	Conventional	Upper Memory
SYSTEM	16,368 (16K)	10,016 (10K)	6,352 (6K)
HIMEM	2,704 (3K)	2,704 (3K)	0 (0K)
EMM386	3,632 (4K)	3,632 (4K)	0 (0K)
LBACACHE	20,464 (20K)	336 (0K)	20,128 (20K)
COMMAND	4,064 (4K)	3,024 (3K)	1,040 (1K)
FDPM	928 (1K)	928 (1K)	0 (0K)
DISPLAY	12,048 (12K)	12,048 (12K)	0 (0K)
SHARE	9,248 (9K)	9,248 (9K)	0 (0K)
SHSUCDX	6,048 (6K)	6,048 (6K)	0 (0K)
HOUSE	3,328 (3K)	3,328 (3K)	0 (0K)
DOSLPM	13,520 (13K)	13,520 (13K)	0 (0K)
PERUSE	4,112 (4K)	4,112 (4K)	0 (0K)
XCDROM	2,320 (2K)	0 (0K)	2,320 (2K)
CDRCACHE	13,072 (13K)	0 (0K)	13,072 (13K)
NORESYS	544 (1K)	0 (0K)	544 (1K)
SC	16 (0K)	0 (0K)	16 (0K)
Free	586,464 (573K)	584,832 (571K)	1,632 (2K)

Memory Type Total Used Free
Press <Enter> to continue or <Esc> to exit . . .

Aki a régi DOS operációs rendszerből próbál kiindulni a FreeDOS értékelése során, meg fog lepődni, ugyanis nem egy egyszerű operációs rendszert, hanem komoly alkalmazásokkal felvértezett komplett programegyüttest kapunk a telepítés után. Kis túlzással mondhatjuk, hogy az MS-DOS és a FreeDOS úgy aránylik egymáshoz, mint egy sima Win és egy komplett Linux disztribúció. Szerencsére itt nem kell kitérni arra, hogy a Win2007 WordXP-ből miért is hiányzik a zekcel, amit néhány önjelölt poweruser szokott hiányolni az OEM Win telepítés után.

Nos, lássunk néhány konkrétumot. Az alap Free-DOS funkcionálisan kiválthatja korábban meglévő DOS rendszereinket, vagyis használhatjuk a megszokott parancsokat, sőt lehetőségeinket jelentősen kitágítja az új operációs rendszer.

Azért néhány különbséggel rendelkezik a FreeDOS, az első ami szembetűnik, hogy az fdos könyvtárba pakolja magát, de ez nem olyan dolog, amivel ne lehetne megbarátkozni, vagy segíteni rajta.

Kicsit visszaszaladva, a

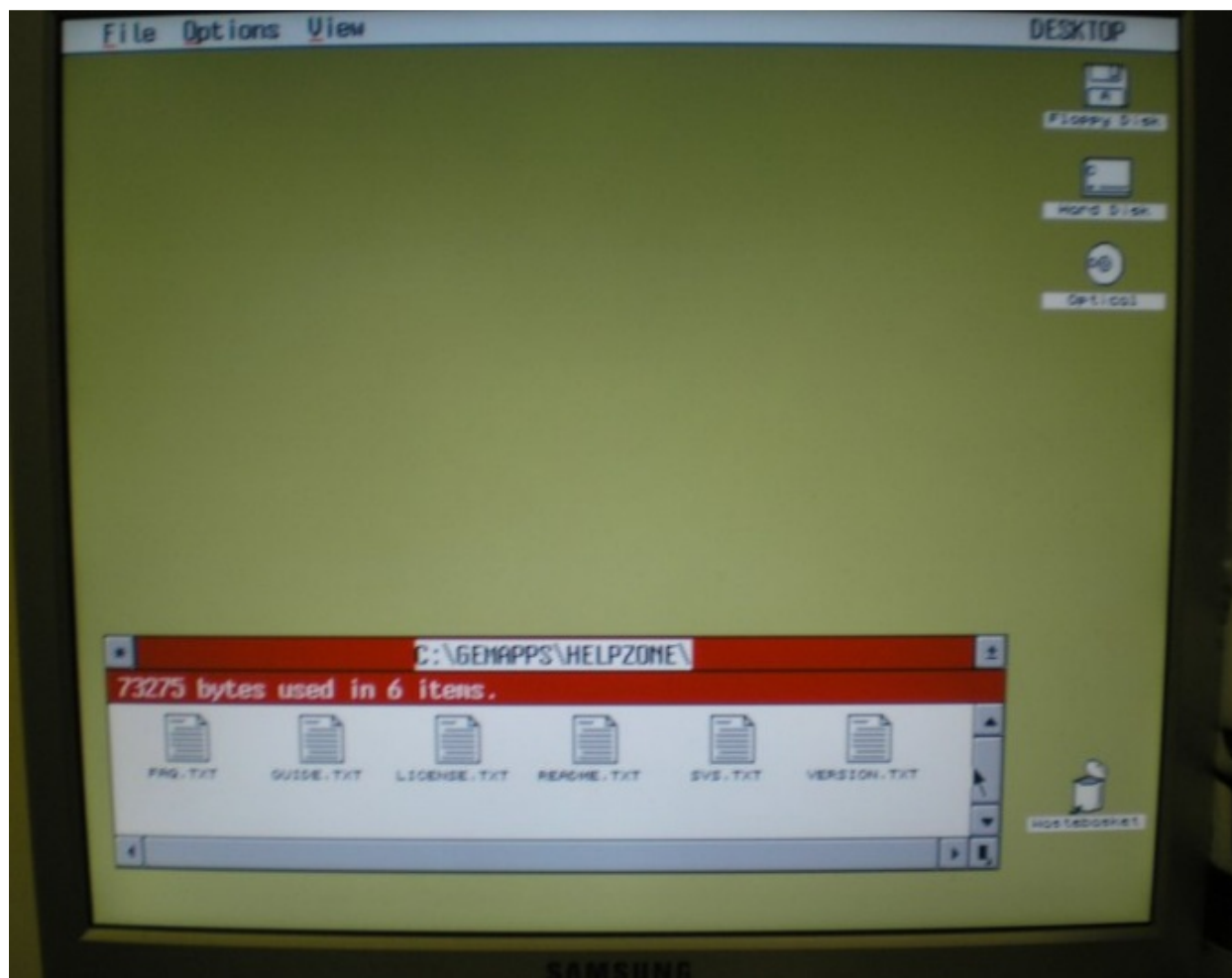
telepítésről csak annyit írnék, hogy választhatunk többféle mód közül, a rendszer felrakhatja magát a hard diskre, de indulhat live cd-ként is a CD/DVD meghajtóról, sőt tudunk floppy disk készletet is fabrikálni magunknak. Mivel rövid ismertetőről van szó, egyelőre nem mélyednék bele a telepítés folyamatába, eltart egy ideig mire mindent beállítunk és feltelepítünk, persze az elején választhatunk, (normális rendszereknél megszokott módon) hogy milyen környezetet szeretnénk magunknak.

Akkor nézzük mit kaphatunk a pénzünkért (hahaha), a teljesség igénye nélküli,

rövid 200 oldalas felsorolás következik :).

Csak hogy belevágjak a közepébe, van itt valaki az első búr háborúból, aki használt GEM-et? Nekik szól a következő szóösszetétel, OpenGEM, na ezt is kapunk. GUI-ból ennyi bőven elég, vissza az elejére.

A FreeDOS-sal kapott parancsértelmező több, mint remek, de mint tudjuk, mindig vannak olyan ex-leprások, akiknek semmi nem elég, ezért kapunk 4DOS-t is, aki akarja lecserélheti a command.com-ját. Szabad



választásokat a munkaasztalra!

Hogy elmondhassuk magunkról, valóban visszafelé haladunk, nézzük a kapott bootmanagereket. Mert ugye nem egyet kapunk, hogyan is venné az ki magát, sőt hová is vezetne ez?

Most tényleg csak felsorolás következik, mivel már látom, ahogy Örs összevonja a szemöldökét, ugyanis eredetileg max 1 A4-es oldalról volt szó (majd leveszem a betűméretet 1 pontra): BLACKOUT, BOOTMGR (na ez meg mi lehet?), Bootsplash (még jó, hogy nem bootcrash), GNU GRUB, Smart Boot Manager (SMBT MGR), XOSL.



Innentől valóban rövid lesznek, fejlesztőeszközök garmadája (na jó, ez enyhe túlzás), pl.: ARROW-ASM, FLAT ASSEMBLER, Free Basic, Free Pascal, NASM, OpenWatcom, PACIFIC-C, YASM MODULAR ASSEMBLER, stb mivel a felsorolás nem teljes. Editorok, pl.: EMACS, FED, VIM, SETEDIT, stb. GUI: az OpenGEM-en kívül kapunk oZone-t is. Netes cuccok: csak kiragadva néhányat, Arachne WWW browser, lynx, Pegasus (nyilván for DOS), SSH2DOS, Terminal, stb. Hanggal kapcsolatos dolgok: OpenCubic player, ABCMIDI, LAME, MIDI PLAYER, MPXPLAY, Normalize, stb. A Util szekcióban helyet kapott jó néhány tömörítő (szerencsére kicsomagolóval együtt), a már említett 4DOS, és még rengeteg hasznos segédprogram.

Mire mindent átnézünk, gondolom megérkezik a Free vagy Open CP/M, aztán kezdetünk mindent előlről.

Ja, a játékokat nem is említettem (a freeDOOM elvette az eszemet és az időmet), na akkor itt megemlíteném, hogy játékokat is kapunk.

Zárszóul csak annyit, milyen jó lett volna mindez 1987-ben. Az örök elégedetlenkedők pedig csinálhatnak maguknak a reklámszünetben egy sokkal jobbat.

Szőke József

A cikkhez tartozó fórum címe:

<http://www.flosszine.org/freedos>

MODSECURITY

Mi a ModSecurity?

A ModSecurity egy webapplikációs tűzfal (web application firewall, WAF), amely Apache 2.x.x webkiszolgálóval képes együttműködni, és modulként (security2_module) használható vele. Ha valaki ragaszkodik az 1.3.x-as Apache verzióhoz, az is beszerezhet egy régebbi verziójú ModSecurity-t. A program kétféle licenc alatt is elérhető, GPL v2 ill. a Breach Security nyújtotta kereskedelmi licenccel is.

Miért van rá szükség?

A web alkalmazásokat - pl. on-line áruházak, blogok, fórumok, stb. - számos támadás fenyegeti, pl. SQL befecskendezés, XSS, komment spam. Ha az említett alkalmazások 100% biztonságosak volnának, nem lenne szükség arra, hogy megvédjük őket, de sajnos a „Hello World!” programok kivételével aligha létezik teljesen biztonságos program. Itt jön képbe a ModSecurity, amely a http forgalmat még azelőtt átvizsgálja, mielőtt az elérné a (potenciálisan) sérülékeny web alkalmazást, és a támadásokat blokkolja, ill. képes ezeket naplózni is. Ez utóbbi azért fontos, mert a szokásos Apache naplózó eszközök nem adnak arra lehetőséget, hogy a POST adatokat elmentsük, pedig ezek nélkül nem tudjuk meg, hogy pontosan milyen adattal akasztotta ki a támadó az alkalmazást. A ModSecurity naplózása abban is a segítségünkre lehet, hogy finomítsuk a szabályainkat.

A modulra úgy is tekinthetünk, mint egy IDS alkalmazásra, amely a hagyományos behatolásérzékelő megoldásokkal szemben könnyen elbánik az SSL-titkosított forgalommal is, ugyanis a ModSecurity az SSL dekódolása után látja a http forgalmat. További előnye, hogy nem igényel speciális környezetet (pl. külön vas, OS, ...), hanem mindenhol elfut, ahol az Apache életképes.

Telepítés

Töltsük le a modult a <http://www.modsecurity.org/download/index.html> címről, a cikk írásakor az 2.5.2-es verzió a legfrissebb. A használatához szükség lesz még a lua, curl, libxml2, ill. az apr és apr-utils nevű komponensekre. A telepítés a szokásos módon történik:

```
tar zxvf modsecurity-apache_2.5.2.tar.gz
cd modsecurity-apache_2.5.2/apache2
./configure
make
su -c 'make install'
```

A csomagban található egy modsecurity.conf-minimal nevű fájl, amely egy minimális konfigurációt biztosít, ami elegendő ahhoz, hogy elinduljunk. Módosítsuk a httpd.conf állományt és adjuk hozzá az alábbi részletet. Ez egyrészt betölti a mod_security2 modult, ill. hozzáadja az említett, minimális szabályokat tartalmazó állományt.

```
LoadModule security2_module
    modules/mod_security2.so
<IfModule security2_module>
    Include conf/modsec.conf
</IfModule>
```

Ha nem akarunk magunk bajlódni a szabályok megírásával, a fenti web oldalról letölthetjük az ún. core rule-okat. Ez egy olyan ingyen elérhető szabálygyűjtemény, amelyet akár egy az egyben használhatunk (ehhez csak ki kell csomagolni a modsecurity-core-rules_2.5-1.6.0.tar.gz fájlt, (és hivatkozni rá a httpd.conf-ban), vagy jó kiinduló alapot ad ahhoz, hogy megírjuk a saját szabályainkat. A következő példákban a minimális ModSecurity konfigurációt veszem alapul, és ehhez adom hozzá az ismertetett szabályokat.

Csapjunk a közepébe!

A webkiszolgálók hajlamosak sok információt elárulni magukról a „Server:” fejlécben, amelyekkel egy hacker időt spórolhat meg magának. Az 1. táblázatban néhány web kiszolgáló által közölt adatokat láthatjuk.

Apache/2.2.8 (FreeBSD)	Népszerű portál
UltraBrutal Server/7.617	Hazai ISP
Oracle Application Server/10g (10.1.2) Digital River/2.14 (Unix) PHP/4.4.4 with Hardening-Patch OracleAS-Web-Cache-10g/10.1.2.0.0 (H;max-age=900+300;age=185;ecid=57987868597,0)	www.simtel.net
Microsoft-IIS/7.0	„Bill” Co.

1. Táblázat: Néhány web kiszolgáló „Server:” mezőben közölt adatai

Adjuk az alábbi sort a ModSecurity konfigurációjához:

```
SecServerSignature "Boa/1.2.3 (C64)"
```

Ennek hatására az Apache kiszolgálónk az alábbi információt mondja magáról:

```
Server: Boa/1.2.3 (C64)
```

Az ellen ... véd

A blogok, fórumok és wikik nagy problémája, hogy a spammerek egy halom linket helyeznek el bennük, hogy ezzel is előrébb kerülhessenek a keresőgépek találati listáján. Nyilván az nem túl jó ötlet, ha a html linkeket teljesen száműzzük, de azt megtehetjük, hogy azokat a kéréseket elutasítjuk, amelyek négy-nél több linket tartalmaznak. Írjuk az alábbi konfigurációs részletet a modsec.conf végére.

1. szabály:

```
SecRule ARGS|ARGS_NAMES "(http:\\/.*){4}" \
  "t:htmlEntityDecode,t:compressWhiteSpace,t:lowercase, \
  ctl:auditLogParts+=E,deny,log,auditlog,status:501, \
  msg:'Comment Spam',id:'950020',severity:'3'"
```

Ezzel azt adtuk meg, hogy (mind a POST, ill. GET kérések esetén) ha vagy a változók neveiben vagy azok értékeiben szerepel a „http:” sztring legalább négyszer, akkor végrehajtjuk a következő (t:-vel jelölt) transzformációkat: HTML dekódolás (pl. &#nnn, <), a fehér karaktereket szóközre cseréljük, és



az egymást követő szóközökből csak egyet hagyunk meg, ill. minden karaktert kisbetűvé konvertálunk. A „ctl” direktíva módosítja a megadott (jelen esetben auditLogParts) opciót. A „deny” hatására a kérést elutasítjuk, továbbá a kérés bekerül az auditlog-ba, ill. az error_log fájlba.

Ezután amikor egy olyan kommentet akartam elküldeni, amelyikben öt html link szerepelt, akkor a kiszolgáló 501-es hibát (Cannot process request) küldött vissza (a „status:501” miatt), és a POST kérésem el sem jutott a Wordpress blogba.

Az error_log-ba az alábbi

bejegyzés került, amelyből kiderül, hogy a /etc/httpd/conf/modsec.conf fájl 75. sorában akadt el a kérés:

```
[client 192.168.x.x] ModSecurity: Access denied with
code 501 (phase 2). Pattern match "(http:\\/.*){4}"
at ARGS:comment. [file "/etc/httpd/conf/modsec.conf"]
```

```
[line "75"] [id "950020"] [msg "Comment Spam"]
[severity "ERROR"] [hostname "teszt.aaaa.fu"] [uri
"/wordpress/wp-comments-post.php"] [unique_id
"m4tGCMCoAccAAAr7CaoAAAAA"]
```

Az audit log-ba bekerült az audit fejléc (A), a http kérés fejléce és így a megtámadott URL is (B), a kérés törzse (C), a válasz (F), az audit log trélere (H) – hasonló adatokkal, mint ami az error_log-ba kerül, végül pedig a bejegyzés végét jelző rész (Z).

```
--3819aa26-A--
[06/May/2008:19:59:20 +0200] m4tGCMCoAccAAAr7CaoAAAAA
192.168.1.22 42694 192.168.1.199 80
--3819aa26-B--
POST /wordpress/wp-comments-post.php HTTP/1.1
User-Agent: Opera/9.xx (X11; Linux i686; U; en)
Content-Length: 1162
...

--3819aa26-C--
author=ajkaal&email=sklskls%40ajkakal.lo&url=&comment=
%3Ca+href%3D%22http%3A%2F%2Fgroups.google.com%2Fgroup%
2FgrjqSdests-grooup-4%2Fweb%2Foutdoor-sex-
clips%22+rel%3D%22nofollow%22%3EOutdoor+sex+clips%3C%2
Fa%3E%3C...&submit=Submit+Comment&comment_post_ID=1

--3819aa26-F--
HTTP/1.1 501 Method Not Implemented
...

--3819aa26-H--
Message: Access denied with code 501 (phase 2).
Pattern match "(http:\\/.*){4}" at ARGS:comment.
[file "/etc/httpd/conf/modsec.conf"] [line "75"] [id
"950020"] [msg "Comment Spam"] [severity "ERROR"]
Action: Intercepted (phase 2)
Apache-Handler: type-map
Stopwatch: 1210096760342024 5133 (1115* 1705 -)
Producer: ModSecurity for Apache/2.5.2
(http://www.modsecurity.org/).
Server: Apache/2.2.8 (Unix) DAV/2

--3819aa26-Z--
```

Szintén gyakori támadás az SQL befecskendezés, amelynek során a támadó úgy

manipulálja a CGI input adatokat, hogy az így létrejött SQL parancs nem egészen azt fogja végrehajtani, vagy teljesen mást, mint amit szerettünk volna. Vegyük a következő SQL utasítást, aminek hatására pl. egy PHP script megkeresi Béla email címét.

```
SELECT email FROM users WHERE name='bela'
```

De mi a helyzet, ha egy támadó nem a logikusan várt „bela” sztringet adja át paraméterként, hanem az alábbi?

```
bela' OR 1=1 OR name=''
```

Ebben az esetben valójában a következő SQL utasítás hajtódik végre, amelynek hatására az összes felhasználó adatait kilistázza a program:

```
SELECT email FROM users WHERE name='bela' OR 1=1 OR name=''
```

Próbáljuk megfogni ezt a támadást az „OR 1=1” részletnél:

2. szabály:

```
SecRule ARGS|ARGS_NAMES "or (1\ {0,}\=\{0,}\1)" \
  "t:htmlEntityDecode,t:compressWhiteSpace,t:lowercase,deny,log,auditlog,status:501,msg:'SQL injection' \
  #1',id:'999002',severity:'3'"
```

A szabály kiválóan működik, csak az a baj, hogy a 2=2, 1<2, TRUE, stb. SQL kifejezésekre is működik a támadás. Igazából végtelen módon össze lehet állítani az igaz értéket adó részletet. Módosítsuk ezért az előbbi (2.) szabályt a következőre, amely – ha nem is az összeset – de a leggyakoribb próbálkozásokat képes hárítani:

```
SecRule ARGS|ARGS_NAMES "or (true|\d{1,}\ {0,}\[\\=\<\>]\ {0,}\d{1,})" \
  "t:htmlEntityDecode,t:compressWhiteSpace,t:lowercase,deny,log,auditlog,status:501,msg:'SQL injection' \
  #1',id:'999002',severity:'3'"
```

Ha azonban az alábbi inputot adja meg a hacker, ez sajnos át fog csúszni a ModSecurity szabályainkon:

```
' OR name IN(SELECT name FROM users) OR name='
```

Vegyük fel ezért a következő szabályt is, amely semlegesíti a „SELECT FROM” próbálkozást:

3. szabály:

```
SecRule ARGS|ARGS_NAMES "(select.+from)" \
  "t:htmlEntityDecode,t:compressWhiteSpace,t:lowercase,deny,log,auditlog, \
  status:501,msg:'SQL injection' \
  #2',id:'999003',severity:'3'"
```

Szintén az SQL befecskendezés körébe tartozik az a támadás, amikor a hacker úgy módosítja az SQL utasítást, hogy az egy extra műveletet (pl. INSERT, UPDATE, DELETE, DROP, stb.) is végrehajt. Vegyük a következő CGI adatot:

```
'x'; INSERT INTO users (email, name) VALUES ('gerzson@gerzson.hu','gerzson'); --
```

Ennek hatására egy olyan SQL utasítás áll össze, amely egy extra felhasználót is felvesz az adatbázisba:

```
SELECT email FROM users WHERE name='x'; INSERT INTO users (email, name) ('gerzson@gerzson.hu','gerzson'); --
```

Módosítsuk ezért a 3. szabályt!

```
SecRule ARGS|ARGS_NAMES "(((select|delete|.from)|insert into|update|drop (table|database))" \
  "t:htmlEntityDecode,t:compressWhiteSpace,t:lowercase,deny,log, \
  auditlog,status:501,msg:'SQL injection' \
  #2',id:'999003',severity:'3'"
```

A jelszó fájl elküldése vagy megtekintése egy további módszer, hogy felhasználói fiókok neveit kitaláljuk (és aztán nyers erő támadással megpróbáljunk gyenge jelszavakat kitalálni). Vegyük az alábbi PHP programot, amely megmutatja a megadott fájlt: <http://teszt.aaaa.fu/1.php?file=1.txt>. De mi van akkor, ha az /etc/passwd fájlt adja meg valaki paraméterként? Az 1.php, mivel a /etc/passwd fájlt mindenki olvashatja, örömmel megmutatja az



összes felhasználó adatait. Szerencsére az alábbi szabály segít rajtunk:

4. szabály:

```
SecRule REQUEST_FILENAME|ARGS|ARGS_NAMES  
"/etc/passwd" \\  
"t:htmlEntityDecode,t:compressWhiteSpace,t:lowercase,deny,log,auditlog, \\  
status:501,msg:'/etc/passwd retrieval',id:'999004',severity:'3'"
```

Gyakori eset, hogy a támadó a "../.." szekvenciával akar az adott könyvtáron kívüli állományokat elérni. Ezt az 5. szabállyal semlegesíthetjük.

5. szabály:

```
SecRule REQUEST_FILENAME|ARGS|ARGS_NAMES  
"..\\/{1,}" \\  
"t:htmlEntityDecode,t:compressWhiteSpace,t:lowercase,deny,log,auditlog, \\  
status:501,msg:'directory traversal',id:'999005',severity:'3'"
```

Egy időben arra lettem figyelmes, hogy az egyik web hosting felhasználó módosította az index oldalait, és azokba egy trójai letöltő program web címét illesztette be. Természetesen ezt egy számítógépes kártevő végezte el, nem a felhasználó. Azonban az utolsóként bemutatott szabály képes megvédeni a felhasználókat azoktól a gépünkön lévő weboldaltól, amelyekben szerepel a www.hacker.com hivatkozás.

6. szabály:

```
SecRule RESPONSE_BODY  
"(http|https)\\:\\7\\www\\.hacker\\.com" \\  
"t:htmlEntityDecode,t:lowercase,deny,phase:4,log,auditlog,status:501, \\  
msg:'hacker site',id:'999006',severity:'3'"
```

Záró szavak

A legjobb védekezés persze az lenne, ha az egyes web

alkalmazások golyóállóra lennének megírva, és maguk képesek lennének elhárítani az összes támadást. Sajnos azonban hibák mindig becsúszhatnak, arról nem is beszélve, hogy pl. egy web hosting szolgáltató esetében lehet egy-két kíváncsi ügyfél, akik több információt szeretnének begyűjteni, akár a rendszerről, akár a többi felhasználóról, mint amennyit ildomos. A ModSecurity ezek ellen is hatásos védelmet adhat.

A szabályokat érdemes tesztelni, nehogy fals pozitív hibával keserítsük meg a látogatóink életét. Pl. a 3. szabályban szereplő „update” kifejezés teljesen helyénvaló lehet a megfelelő kontextusban, ez a szabály azonban mégsem engedi a használatát. A ModSecurity képes ún. DetectionOnly módban is működni, amikor csak naplózza a sértéseket, de nem korlátoz. A bemutatott példák természetesen csak kedvcsinálóként szolgálnak. Bátorítom az olvasót, hogy írja meg a saját szabályait, vagy módosítsa, finomhangolja a cikkben szereplőket.

Sütő János

A cikkhez tartozó fórum címe:

<http://www.flosszine.org/modsecurity>



Védtelen véletlenek

(Debian/OpenSSL bug)

Kivonat

Az OpenSSL nevet az informatika berkein belül sokan ismerik, általánosságban elismerően nyilatkoznak tevékenységükről, becsülik az elért eredményeket. Mára



Kurt Roeckx neve sem csupán egy a sokból, bár talán az előbbihez hasonló jó hírnévről nem beszélhetünk. A két név erős összefonódása az utóbbi hetek cikkeiben

sem a véletlen műve, hiszen Kurt Roeckx –egyebek mellett– az OpenSSL szoftvercsomag karbantartója a Debian disztribúcióban, egyszersmind annak a patchnek a szerzője, mely a közelmúlt egyik legsúlyosabb biztonsági problémájához vezetett nem csupán a Debian, hanem az azon alapuló disztribúciók mindegyikében. A tények tények, a helyzet adott, a kérdés csupán annyi, mit tehetünk boszorkányüldözés helyett?

Kronológia

2006.04.19. Az események végzetes hibába torkolló láncolatát a Richard Kettlewell által bejelentett 363516-os sorszámú Debian hibajegye indítja el, amely egy az OpenSSL függvénykönyvtár Valgrind melletti használatával összefüggően felmerült problémára hívja fel a figyelmet.

2006.05.01. Kurt Roeckx, az openssl csomag karbantartója, az openssl-dev levelező listán egyeztet a OpenSSL fejlesztőivel, vázolja a felmerült problémát és ismerteti saját megoldási javaslatát.

2006.05.02. A hiba orvoslására szánt változtatás feltöltésre kerül a Debian saját verziókezelő rendszerébe. Egyelőre ez nem okoz semmilyen gondot, mivel hibás helyre kerül a változtatott állomány, így annak hatása a binárisban nem jelentkezik.

2006.09.17. A verziókezelő használatakor elkövetett botlás korrigálásra kerül, egyszersmind az ezúttal már valóban hibás OpenSSL csomag publikálásra kerül az unstable ágba.

2008.05.07. A sebezhetőséget okozó változtatás, a probléma szempontjából kritikus része visszavonásra kerül, a javított verzió ezzel elérhető az unstable ágból.

2008.05.13. A Debian biztonsági figyelmeztetést ad ki, melyben tájékoztatja felhasználóit a Luciano Bello által felfedezett hibáról és kitér az érintettség, valamint a súlyosság kérdésére is.

Helyzetelemzés

A személyi felelősség kérdése –még ha sokakban felmerül is– kevésbé fontos a jelen helyzet kialakulásának szempontjából. A jövőbeni hasonló események elkerüléséhez a folyamat kritikája visz közelebb. Próbáljunk tehát ezen a módon közelebb kerülni a probléma forrásához, illetve felvetni néhány kézenfekvő kérdést:

Hibabejelentés

A felhasználói elégedetlenséget kiváltó probléma az OpenSSL függvénykönyvtárhoz kapcsolódik, így azt a terjesztő felé jelezni csupán egy felesleges lépés iktatása a kommunikációs folyamatba, mely könnyen félreértésekhez vezethet. A javítás feladata, annak mikéntje és határideje egyaránt a fejlesztők kompetenciájába tartozik.

Hibaleírás

Amennyiben megtaláltuk a megfelelő fórumot (a konkrét esetben az openssl-dev levelező lista) ismeretlenként nem haszontalan rávilágítanunk, hogy a javítással milyen célt kívánunk elérni, illetve a problémára adott esetleges megoldási javaslatokat kizárólag saját céljainkra egyszeri esetben használnánk-e fel, avagy széles körben kívánjuk azt terjeszteni.

Javítás

Javító foltok készítése természetesen hasznos és az ehhez szükséges elszántság és szorgalom becsülendő, ugyanakkor ingoványos területekre vihet az ellenőrizetlen buzgóság.

Kommunikáció

A nyílt forrású rendszerek kommunikációjának jelentős része maga is nyílt, így a legrosszabb taktika amit választhatunk az elkendőzés, hiszen a hibák az avatott szemek számára egyértelműek és nehezen elfedhetőek.

Reakciókészség

Ha már nyakunkon a baj, egyebet nem tehetünk, mint hogy a problémát mielőbb javítjuk, a javítást minél szélesebb körben tesszük elérhetővé és minél pontosabb információval szolgálunk a felhasználók számára a következmények tekintetében.

Tanulságok

A fentiek némiképp utólagos okoskodásnak tűnhetnek, ugyanakkor tegyük fel magunknak a kérdést, mi az amit erősségünknek tartunk az open source közösségen belül és mi az amit hiányolunk a versenytársak viselkedési mintáiból. Előbbiek között felsorolható a gyors reagálóképesség és a nyíltság. Annak fényében azonban, hogy a Debian nem hozza nyilvánosságra, hogy Luciano Bello valójában mikor ad hírt felfedezéséről, ezen felül május 7. és 13.



közötti 6 napban semmit nem tesz, ezzel is tovább növelve az esetleges rendszerkorruptiók kockázatát, valamint a biztonsági figyelmeztetésének kiadását követően egy olyan fekete listával áll elő, melynek generálási módszere ismeretlen, elmondható, hogy azon közösség értékeiből, melynek meghatározó tagja, ebben a szituációban vajmi keveset tud felmutatni. Ellenben a másokkal szemben támasztott elvárásainknak, mint a kooperáció, együttműködési hajlam sem ékes példája, sajnos nem is első ízben, hogy házon belül próbálnak kezelni olyan problémákat, melyek megoldása szélesebb körben is igény lehetne.

*Jogos lehet a kérdés: **Akkor most remegnek a nyílt forrás oszlopai?***

Nem, az ideák szintjén semmiképp, ugyanakkor a való életben saját elveinktől és tapasztalati úton kikristályosodott módszereinktől való eltávolodás az általánosítás veszélyét hordozza, vagyis a szem előtt lévő hibákra figyelve alakíthatnak ki negatív véleményt azok, akiknek a nyílt forrás épp oly jó választás lehet, mint más egyéb megoldások.

Pfeiffer Szilárd

A cikkhez tartozó fórum címe:

http://www.flosszine.org/vedtelen_veletlenek_debian_openssl_bug



```
editor@flosszine.org:/dev/null$ sleep 5356800_
```