Unix és Linux

A Unix rövid története

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

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

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

A Unixról 1974-ben jelent meg az első részletes publikáció. A Bell Laboratórium anyavállalata, az AT&T akkoriban ki volt tiltva a számítógépek piacáról, így a Unixot nem árulhatta pénzért. Semmi sem indokolta, hogy az operációs rendszert – forráskódjával együtt – ne adják oda bármelyik egyetemnek. Így is tettek. A másik körülmény, hogy a Unix PDP-11-en futott. Az egyetemeken akkoriban szinte nem is volt más gép, mint PDP-11. És ezen a ponton a DEC cég is alaposan besegített a Unix világsikerébe: a PDP-11 operációs rendszere borzalmas volt. Ezek után könnyű kitalálni, hogy mi történt: egyetemek százain kezdték

használni a Unixot, a forráskód birtokában pedig megindult a véget nem érő buherálás.

A következő mérföldkövet az amerikai kormány rakta le azzal, hogy részekre bontotta az AT&T vállalatot, az utódok számára pedig megnyitotta az utat a számítástechnikai ipar felé. Hamarosan megjelent az első kereskedelmi Unix változat, System III néven, egy évvel később megjelent a System V nevű javított változat. A System V több kiadást is megért, ezek Release 2, 3 és 4 néven ismertek.

A számtalan Unix-buheráló egyetem közül kiemelkedik a Berkeley Egyetem. ők számos ponton javítottak az AT&T-től valaha ingyen kapott Unixon, gyorsabb fájlrendszert írtak, beépítették a hálózatkezelést (ami TCP/IP néven legalább olyan elterjedt a hálózati protokollok között, mint a C a programozási nyelvek terén), valamit számos segédprogramot (csh>, vi, fordítóprogramok stb.) Az egyetem kereskedelmi változatú szoftvercsomaggá formálta az általuk fejlesztett Unix változatot és BSD (Berkeley Software Distribution) néven terjeszteni kezdte.

Amint látható, a nyolcvanas évek végére a Unix-világot két különböző, egymástól sok dologban eltérő Unix uralta. Ez elsősorban azzal a szomorú következménnyel járt, hogy a System V Unix alatt írt C programra a BSD-s rendszerek fordítói fityiszt mutatnak, és fordítva.

Nagyjából minden nagyobb számítógépgyártó a két elterjedt rendszer (System V, BSD) valamelyikére alapozva készítette el saját Unix változatát. Mivel az inkompatibilitás senkinek nem tesz jót, ezért az IEEE létrehozott egy UNIX-szabványt, Portable Operaring System (rövidítve: POS) néven. Hogy Unixosabban hangozzék, a rövidítéshez hozzábiggyesztették az IX betűket: így lett a dologból POSIX. E szabvány azt írja le, hogy egy "hordozható" operációs rendszernek hogyan is kell kinéznie. Minden nagy gyártó elismeri a POSIX jelentőségét, operációs rendszereiben támogatja is azt. Mindazonáltal ez nem akadályozza meg őket abban, hogy olyan kiegészítéseket, nem szabványos interfészeket és egyéb szolgáltatásokat építsenek be a termékükbe, amitől a különböző rendszerek továbbra is inkompatibilisek lesznek. A riválisok két szervezetbe tömörültek, a szögesdrótok egyik oldalán a DEC által vezetett Open Systems Foundationt (OSF) találjuk, velük szemben pedig az AT&T alapította Unix International (UI) szövetséget.

Vizsgáljuk meg egy kicsit közelebbről e nagy háborúskodás okát! Az ősidőkben, mikor néhány számítástechnikai mamutcég uralta a piacot, a számítógépek "önmagukkal" voltak kompatibilisek, azaz ha elromlott egy alkatrész, vagy a rendszert bővíteni akarták, a berendezéseket kizárólag az eredeti gyártótól lehetett beszerezni. A mamutok alaposan vissza is éltek azzal, hogy a vásárló még a portörlő rongyot is kénytelen tőlük vásárolni, egy-egy alkatrészért szemrebbenés

nélkül elkérték a piaci ár háromszorosát. (Néhány cég ma is folytatja ezt a gyakorlatot.)

A védekezést az jelentené, ha a különböző gyártók számítógépei hajlandók lennének együttműködni, azaz eltérõ típusú gépekből is lehetne rendszert építeni. Ez egyrészt nagyobb szabadságot adna a felhasználóknak a vásárlásnál, másrészt esélyt biztosítana a gyártóknak, hogy berendezéseiket minél szélesebb körben vásárolják. Egy ilyen nyitott rendszerbe utólag bármelyik cég szoftverét vagy számítógépét be lehet illeszteni: ezt az álmot hívják "nyílt rendszereknek" (Open Systems).

Nyilvánvaló, hogy a nyílt rendszerek kulcsfontosságú kérdése a minden gyártó által elfogadott (és be is tartott) szabvány. A kilencvenes évek elején létrehoztak egy gyártófüggetlen szervezetet a nyílt rendszerek szabványainak elkészítésére és a "szabványnak megfelel" minősítések kiadására. Ez a szervezet az X/Open, az általuk készített operációs rendszer szabvány neve pedig Core OS API, más néven Common API.

Ha egy operációs rendszer átmegy az X/Open minősítő tesztjein, megkapja a UNIX 95 minősítést. A "levizsgáztatott" operációs rendszerre írt programok forráskódja hordozható, azaz buherálás nélkül le lehet fordítani egy másik gyártó ugyancsak UNIX 95 kompatibilis operációs rendszerén.

Nézzük, mit lát a UNIX 95-ből az egyszerű felhasználó! Hmmm... egy ugyanolyan mezítábas parancssort, mint amilyet húsz évvel ezelőtt. A szabványban ugyanis nem szerepel a grafikus felhasználói felület, amelynek egységesítéséért ugyancsak ádáz küzdelem folyik. Mivel a UNIX alatt futó alkalmazások (például a nagy adatbázis-kezelő rendszerek) karakteres alapon működnek, jóval kevesebb gyártót érint a grafikus felhasználói felület ügye.

A Linux rövid története

Az elnevezés szigorúan véve a Linux kernelt jelenti (az operációs rendszer magja), általánosabban: a kernel és a GNU rendszerkönyvtárak és alkalmazások együttesét; még tágabb értelemben a különféle disztribúciókat. Pontos elnevezéssel élve GNU/Linux, de röviden Linux. A disztribúció nem más, mint egy telepítésre és felhasználásra kész GNU/Linux operációs rendszer, valamint válogatott felhasználói programok gyűjteménye.

1991: Linus Torvalds másodéves a Helsinki egyetemen, és a 386-os processzor védett módú (protected mode) üzemmódját, taszkváltó képességével kísérletezik. 1991. augusztus 25.: Linus bejelentése a minix levelezési listán.

> „Üdv minden Minix-felhasználónak odakinn! Készítek egy (ingyenes) operációs rendszert (csak hobbi, nem lesz olyan nagy és profi, mint a

> GNU a 386- (486) AT-klónokhoz. Április óta kotyvasztom, és már kezd elkészülni. (...) Mostanában ültetttem át a bash (1.08) és a gcc (1.40) programokat, és úgy tűnik, működnek a dolgok. Ez azt is jelenti, hogy pár hónapon belül valami használhatót fogok kapni, és kíváncsi lennék, milyen képességeket szeretnének az emberek. Minden javaslatot szívesen veszek, azt viszont nem ígérem, hogy meg is csinálom öket :-)
>
> Linus (torvalds@kruuna.helsinki.fi)
>
> Ui.: Igen! Nincs benne Minix-kód és többszálú fájlrendszerrel rendelkezik. Nem hordozható (a 386 feladatváltást használja stb.), és lehet, hogy soha nem is fog az AT-merevlemezeken kívül bármi mást támogatni, minthogy nekem csak ez van :-(.”

A Linux alapjai

A könyvtárszerkezetben találjuk az első látványos különbséget. Nincsenek lemezegységek (A:-tól Z:-ig), hanem egyetlen könyvtárfa van, legyen akárhány fizikai háttértár eszközünk. Ha egy újabb eszközt, mondjuk egy pen-drive-ot bedugunk a számítógépbe, meg kell adni, hogy a meglévő könyvtárfa melyik végpontjára csatlakozzon (pl. /mnt/pendrive). Ezután az eszköz tartalmát a megadott alkönyvtáron (mappán) keresztül lehet elérni.

Még a Unix tervezése során lefektettek egy, azóta nagyon hasznosnak bizonyult alapelvet, miszerint minden fájl (az „igazi” fájlokon kívül a mappák és a háttértárak, illetve egyéb perifériák is). Ez utóbbiak a /dev mappán keresztül érhetők el. Pl. egy átlagos otthoni PC-ben van két IDE-vezérő, mindegyikre rákötheto két IDE-eszköz (merevlemez v. CD/DVD; egy „mester” és egy „rabszolga”), és ezek a /dev/hda, /dev/hdb, /dev/hdc, /dev/hdd fájlokon keresztül érhetők el. A hd nyilván a hard disk (merevlemez) rövidítése. Ezek partícióit pedig a következő neveken keresztül érhetjük el: /dev/hda1, ..., /dev/hdd4 stb., értelemszerűen. A SATA v. SCSI merevlemezeket pedig a /dev/sda stb. fájneveken keresztül.

A jogosultsági rendszer igen egyszerű, de roppant hatékony. Előzetesen annyit kell tudni, hogy a felhasználók csoportokat alkotnak a Unix/Linux környezetben. Egy tipikus jogosultságra vonatkozó rész pl. így nézhet ki: „-rwxr-x---”.

Ennek értelmezéséhez ezt a jelsorozatot részekre osztjuk: -|rwx|r-x|---. Az r az olvasási jogot (readable) jelenti, a w az írási jogot (writeable) az x pedig a futtatási (executable). Az első hármas csoport mutatja a fájl tulajdonosának jogait – jelen esetben olvashatja a fájl tartalmát, írhat bele, és futtathatja. A második hármas a tulajdonos csoporttársainak jogait mutatja, esetünkben ők olvashatják a fájlt és futtathatják, de a tartalmát nem változtathatják meg, mert nincs írási joguk rá. A harmadik hármas csoport pedig a „bárki más” jogait adja meg, jelen

esetben semmilyen joga nincs másoknak. A legelső kötőjel azt jelenti, hogy a bejegyzés egy közönséges – reguláris – fájl. Alkönyvtár (mappa) esetében itt d mint directory áll. A b vagy a c jel azt jelenti, hogy az adott fájl egy blokkos vagy karakteres eszköz, periféria (blokkos pl. a lemezegység, karakteres pl. az egér vagy a billentyűzet). Az itt álló l a link rövidítése, ami egy máshol található (akár esetenként nem található) fájlra mutató hivatkozás.

A fájlrendszer nem más, mint annak módszere, hogy egy lemezegységen milyen rendben tároljuk a fájlokat. Különféle igények és adottságok esetén különféle megoldások születtek az idők folyamán. Feltehetően mindenki hallott már a FAT fájlrendszerről és az NTFS-ről. A Linux tipikusnak mondható fájlrendszerei a ReiserFS, az EXT2 vagy az EXT3. A CD-ROM-ok szabványos fájlrendszere az ISO 9660.

A FAT fájlrendszer alapelve, hogy a könyvtárbejegyzés tartalmazza a fájl neve mellett azt is, hogy a lemez hányadik blokkjában található a fájl eleje. Az ún. FAT-táblában (fájlok helyfoglalási táblázata, file allocation table) pedig minden egyes lemezblokknak megfelel egy rovat. Ha a fájl mondjuk az 1456. blokkban kezdődik, akkor a FAT 1456. rovata tartalmazza a fájl második blokkjának a sorszámát stb., a fájl utolsó blokkja esetében pedig a megfelelő FAT-rovatban egy ezt jelző érték található. Más fájlrendszerek más logikára épülnek, és más igényeket (is) kielégítenek, pl. jogosultság kezelése, naplózó fájlrendszerek stb.

A rendszerindítás folyamata (PC és Linux operációs rendszer esetén, merevlemezről) vázlatosan:

A gép bekapcsolása után egy „igazi” memóriában (ROM, read-only-memory, azaz amely a gép kikapcsolt állapotában is megőrzi tartalmát;) található program indul el, amelyik a bekapcsolás utáni feléledéshez szükséges vizsgálatokat, alapbeállításokat stb. elvégzi. Ez oly módon történik, hogy a processzor a bekapcsoláskor úgy „éledd föl”, hogy az utasításszámláló regiszterében (ami a következő végrehajtandó gépi utasítás memóriacímét tartalmazza) egy gyárilag beállított kezdőérték jelenik meg. Ezen a memóriacímen kell kezdődjön az említett program. Ez a POST (power on self test), a bekapcsolás utáni önellenőrzés.

Megállapítja a CMOS memória tartalma alapján, hogy melyik eszközről kell rendszerindítást csinálni, és beolvassa az adott merevlemez legelső szektorát, az ún. MBR-t (master boot record). Ebben a partíciós tábla adatai mellett egy gépi kódú betöltőprogram található, amely betölti és elindítja azt a célprogramot, ami lehetővé teszi, hogy válasszunk esetleg többféle indítható operációs rendszer, vagy változat közül (boot manager, pl. lilo). Ez „tudja” a kernel pontos helyét a merevlemezen, betölti azt, és átadja neki a vezérlést. A tömörített kernel kicsomagolja saját magát, majd elvégzi az indításkor szükséges tevékenységeket (memóriaméret megállapítása, PCI-eszközök és ISA buszok inicializálása, hálózati

protokollok betöltése, IDE-vezérlők és merevlemezek feltérképezése, esetleg a kernel modulok betöltése stb.). Ezután a kernel elindítja az első programot, az init-et. Ez a /etc/inittab-ban foglaltak alapján jár el, az alapértelmezett üzemállapot (futási szint, run level) eléréséhez szükséges programok elindításával. Végül elindítja a felhasználói bejelentkezéseket váró getty programokat a /etc/inittab-ban megadott eszközökön.

Egy Linux operációs rendszert futtató gép számos szolgáltatást nyújt(hat) a külvilágnak, az internetre csatlakozó más gépeknek. Ha egy szolgáltatást gyakran vehetnek igénybe, akkor az azt kiszolgáló program folyamatosan a memóriában van, és folyamatosan figyeli, hogy a számára rendszeresített kapukon (port) érkezik-e hozzá kérés, ha igen, azt kiszolgálja. Az ilyen, állandóan a memóriában lévő és futó programokat nevezzük démonoknak. Olyan szolgáltatások esetében, amelyeket ritkán vesznek igénybe, ez fölösleges memóriapocsékolás lenne, ezért a ritkábban használatos szolgáltatások esetében lehetséges az, hogy az összes ilyent kiszolgáló programok helyett csak egyetlen egy fusson a memóriában (inetd, inet démon). Ez egy táblázat alapján figyeli az összes szükséges kaput, és ha valamelyikre egy adatcsomag érkezik, akkor elindítja az adott szolgáltatásért felelős programot. Az kiszolgálja a kérést, majd kilép, az inetd pedig figyel tovább. Ez esetben a tárhelyfoglalás csökkentéséért némi többlet futásidővel fizetünk.

A bejelentkezéshez egy felhasználói név és a hozzá tartozó jelszó megadása szükséges. Maga a jelszó nem tárolódik a rendszerben, hanem csak egy olyan kód (árnyékjelszó, shadow password), amelynek alapján az eredeti jelszó nem, illetve csak nagyon(!) nehezen állítható vissza.

IRODALOM

[1] http://wiki.hup.hu/index.php/A_UNIX_t%C3%B6rt%C3%A9nete

[2] http://www.mimi.hu/linux/unix_tortenete.html

[3] http://hu.wikipedia.org/wiki/Unix

[4] http://eblokk.inf.elte.hu/I/2003/docs/unix/tortenet.htm

[5] http://www.bsd.hu/?q=unix-tortenelem

[6] http://wiki.hup.hu/index.php/Linux_T%C3%B6rt%C3%A9net

[7] http://debian.inf.elte.hu/linux_doksi/node28.htm

[8] http://www.szabilinux.hu/Lintort/lintort/node5.html

[9] http://www.szabilinux.hu/Lintort/lintort/node5.html

[10] http://www.szabilinux.hu/forditasok/SAG/sag-hu.html