Informatika 15.

A Linux operációs rendszer

Nagy, Gábor

Informatika 15.: A Linux operációs rendszer

Nagy, Gábor
Lektor: Cseri, Tamás

Ez a modul a TÁMOP - 4.1.2-08/1/A-2009-0027 „Tananyagfejlesztéssel a GEO-ért” projekt keretében készült. A projektet az Európai Unió és a Magyar Állam 44 706 488 Ft összegben támogatta.

v 1.0

Publication date 2010

Kivonat

A Linux operációs rendszer rövid bemutatása

Tartalom

15. fejezet - A Linux operációs rendszer

1. 15.1 Bevezetés

Ez a modul megpróbál egy rövid, áttekintő képet adni a Linux operációs rendszerről, amelynek egy jelentős része számos más UNIX-alapú, vagy UNIX-szerű operációs rendszer esetében is felhasználható tudást jelenthet.

A rövid összefoglalás érinti a kernel feladatait, a fájlrendszer felépítését, a felhasználók kezelését. Foglalkozik a karakteres (parancssor) és a grafikus felülettel, azok működési elvével, bemutatja a legfontosabb parancssorban használható programokat. A modul tárgyalja a disztribúciók és a csomagkezelés témakörét is.

2. 15.2 Történeti áttekintés

1969-ben készítette Ken Thompson Unics névre hallgató egyszerű operációs rendszerét az akkoriban elterjedt PDP-7 típusú számítógépre. Szerette volna a rendszert az újabb, PDP-11-es gépeken is kipróbálni, de ahhoz az assemblyben készített programot teljesen újra kellett volna írnia.

Hogy ne kelljen minden egyes géptípusra a teljes programot újra megírni az adott számítógép assemblyjében, a fejlesztésbe bekapcsolódó Dennis M. Ritchie és Brian Kernighan kidolgozták a C programozási nyelvet. A C nyelv alkalmas megfelelő teljesítményű, de emellett hordozható (többféle architektúrára is lefordítható) programok készítésére, így ideális eszköz többféle platformon is futó operációs rendszerek írására.

Az ekkor már UNIX néven futó operációs rendszer hamar népszerű lett. Népszerűségét többek között annak köszönhette, hogy bár nem volt nyílt forráskódú, de forrásához sokan hozzáférhettek a potenciális továbbfejlesztők közül. Idővel sokféle UNIX-szerű operációs rendszer született, ezek egy része (köztük a Linux is) nem használ UNIX kódot, csak a működési elveiben és az azokhoz kapcsolódó szabványokban (a különféle POSIX szabványok vagy a fájlrendszerre vonatkozó Filesystem Hierarchy Standard) követi.

Egy finn egyetemista, Linus Benedict Torvalds, 1991-ben tette közzé egy általa írt egyszerű rendszermag (kernel) kódját, amelyet elsősorban a 80386 processzor védett módú feladat-váltó lehetőségeinek kipróbálása végett készített és működési elveiben a UNIX rendszerek mintáját követte. Bár eleinte még a szerző sem fűzött hozzá komolyabb reményeket, a fejlesztés komoly projektté nőtte ki magát elsősorban két tényezőnek köszönhetően. Az egyik, hogy a program a GPL nyílt forráskódú licenszének köszönhetően bárki számára a forráskódjával együtt szabadon hozzáférhető és akár továbbfejleszthető azzal a feltétellel, hogy a létrehozott programra is hasonló szabályoknak kell vonatkozni. A másik tényező pedig a Linus és a fejlesztésbe bekapcsolódók számára akkor már elérhető Internet, amely lehetővé tette, hogy egymástól távol élő programozók tudjanak hatékonyan együtt dolgozni egy munkán, kezdetben az e-mailben beküldött módosításaik (patch), később pedig különféle változatkezelő rendszerek (először bitkeeper, majd a saját fejlesztésű git) segítségével.

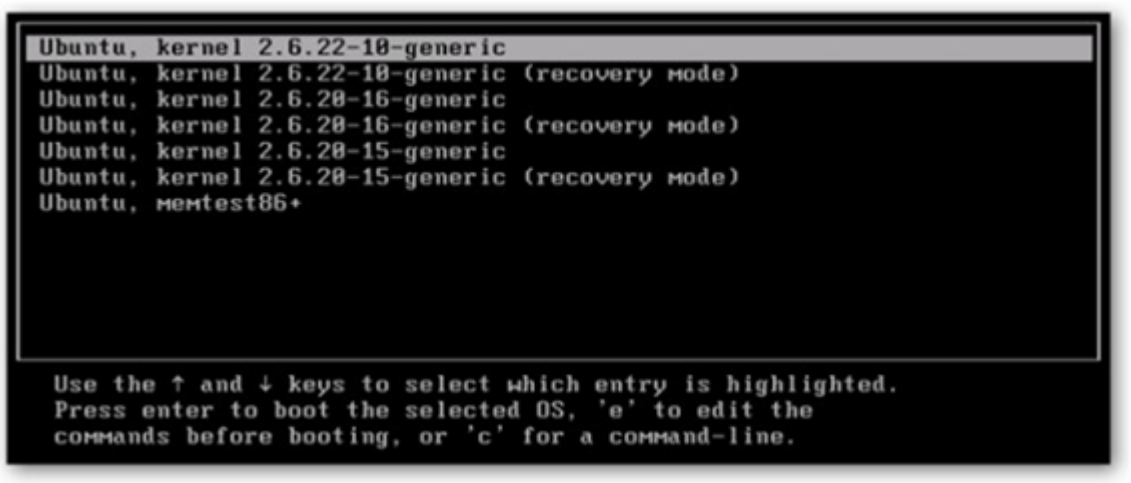
A Linus után Linuxnak nevezett operációs rendszert a fejlesztés során számos platformra portolták, rengetek eszközhöz készült támogatás, és szolgáltatásiban is rendkívüli mértékben bővült. Napjainkban egy jól használható, sokoldalú alternatívát jelent szervereken, beágyazott rendszerekben, az asztali gépekben és a különféle mobil eszközökön egyaránt.

3. 15.3 A Linux legfontosabb jellemzői

3.1. 15.3.1 A kernel

A Linus Torvalds és társai által fejlesztett program nem egy teljes operációs rendszer, hanem annak csupán a magja, a kernel. A kernel biztosítja az operációs rendszer alapvető feladatait felügyeli a programok futását, gazdálkodik a processzorral vagy processzorokkal (ütemezés) és a memóriával, elérhetővé teszi a különféle a számítógépben található eszközöket és kezeli a számítógép fájlrendszerét. Nem tartalmazza viszont a kernel a felhasználói felületet szolgáltató programokat (karakteres vagy grafikus környezet) és a számítógép használatához szükséges segédprogramokat sem.

Nem része a kernelnek a betöltését végző program sem. Korábban főleg a LILO (Linux Loader), mostanában inkább a GRUB (Grand Unified Boot) rendszerbetöltő programokat szokták használni. Ezek képesek a Linux rendszermag betöltésére és elindítására vagy át tudják adni a vezérlést egy másik rendszerbetöltőnek, egy eltérő operációs rendszer (pl. Windows) betöltése érdekében. A számítógép indításakor többféle ilyen lehetőség közül is választhatunk, ezek közül egy alapértelmezett egy megadott idő után automatikusan is elindulhat. A rendszerbetöltők így lehetővé teszik többféle operációs rendszer telepítését egy számítógépre.



Rendszerbetöltő menüje karakteres felületen

A Linux kernel monolitikus, de moduláris felépítésű. Monolitikus, mert nem egy teljesen lecsupaszított mikrokernelt használ, amihez a külön folyamatként csatlakoznak az operációs rendszer további feladatait megvalósító programok. A monolitikus kernelből adódó hátrányokat a Linux a kernelmodulok segítségével igyekszik kiküszöbölni. A kernelnek a modulokban elhelyezett részét opcionálisan, egy külön fájlból tudjuk betölteni. Jellemzően modulokban helyezkednek például el a kernelnek a különféle eszközök kezelését lehetővé tévő részei, vagyis az adott eszközök meghajtói.

3.2. 15.3.2 Programok futtatása

A Linux több program egymással párhuzamos futtatását teszi lehetővé. Az egymás mellett futó programoknak meg kell osztozniuk a számítógép erőforrásain úgy, hogy egy program hibás működése ne okozzon fennakadást a többi program számára. Ennek a biztosítása az operációs rendszer feladata.

Az egymás mellett futó alkalmazások száma jellemzően többszöröse a számítógépben található processzormagok számának, így a programoknak időben kell megosztozniuk a processzoron, amit az operációs rendszer ütemezőnek nevezett része szabályoz. Gyorsan váltogatva a futó programok között, azoknak néhány ezred- vagy századmásodperces kis szeleteket juttatva a processzor idejéből elérhető hogy akár egy csak egyetlen processzort tartalmazó gépen is (látszólag) egyszerre tud futni több program.

A számítógép másik fontos, a futó programok között elosztandó erőforrása a memória. A felhasználói programok meghatározott méretű memória részeket tudnak igényelni (majd később, ha nincs már rá szükségük felszabadítani) az operációs rendszertől, amelyre nem a valós (fizikai) hanem virtuális címek segítségével hivatkoznak majd. Az operációs rendszernek a memóriagazdálkodásért felelős részei a virtuális címeket valós címekre fordítják le, és biztosítják, hogy egy program nem tudjon hozzáférni a nem általa használt memóriaterületekhez.

A virtuális címzés lehetővé teszi, hogy a régebben használt memóriaterületek tartalmát a háttértárolóra kiírják, ahonnan szükség esetén visszaolvashatóak. A program a visszaolvasás után ugyanazon a virtuális címtartományon éri el a memóriának ezt a részét, mint korábban, így a működésébe ez (a visszaolvasáshoz szükséges időt leszámítva) nem okoz fennakadást. Ez a megoldás lehetővé teszi, hogy a programok összességében több memóriát foglaljanak le, mint amennyi RAM a számítógépben rendelkezésre áll.

A rendszer indításakor és a használata során egy összetett inicializációs folyamat gondoskodik a szükséges programok elindításáról, illetve a szükség esetén történő újraindításáról. Ezek között a programok között vannak a rendszer megfelelő működése érdekében a háttérben futó programok, különféle esetenként más

számítógépekről is elérhető szerverprogramok, valamint a felhasználók részére a bejelentkezést valamilyen módon (karakteres terminál, grafikus felület, hálózaton keresztül) biztosító programok.

3.3. 15.3.3 A fájlrendszer felépítése

A Linuxban a fájlrendszer egyetlen fából áll, aminek alapját a gyökér fájlrendszer szolgáltatja. A különféle háttértárolókon vagy hálózati helyeken elhelyezkedő fájlrendszerek már meglévő üres könyvtárakban csatlakozhatnak ehhez a rendszerhez. Ennek az elvnek az alkalmazásával minden számítógépen egyetlen egységes felépítésű fába rendeződnek az állományok, miközben a fájlrendszer egyes részei esetenként különféle háttértárolókon vagy hálózati helyeken helyezkedhetnek el.

Általában a telepítés során, a merevlemez particionálásakor lehet kiválasztani, hogy az egyes partíciókra milyen fájlrendszer kerüljön. Fájlrendszer alatt ebben a bekezdésben most azt a rendszert értjük, ami lehetővé teszi egy merevlemez-partícióhoz vagy más eszközhöz tartozó adott méretű blokkszerű adatterületen könyvtárakba rendezett fájlok adatainak és attribútumainak tárolását. A Linux sokféle fájlrendszert támogat (ext2, ext3, ext4, reiserfs, reiser4, xfs, jfs), ezek különféle tulajdonságokkal jellemezhetőek, egymáshoz képest különféle előnyökkel és hátrányokkal rendelkeznek, amelyek ismeretében a szakemberek kiválaszthatják az egy feladatra leginkább megfelelőt; egyszerű felhasználóként a legbiztosabb a disztribúció telepítője által felkínált alapértelmezésnél maradunk. Vannak olyan fájlrendszerek is, amelyeket más operációs rendszerekkel vagy különféle eszközökkel való kompatibilitás miatt támogat a Linux, mint például a vfat és az ntfs vagy CD és DVD lemezek adatait tároló iso9660.

A fájlrendszernek fontos szerep jut mivel a UNIX-szerű operációs rendszerek egyik alapelve, hogy mindent fájlokba képeznek le. A UNIX-szerű operációs rendszerek egy másik fontos alapelve is kapcsolódik a fájlrendszerhez: minden beállítást fájlokban, jellemzően olvasható szöveges állományokban tárolnak.

A Linux fájlrendszere a UNIX szokásait követve elsődlegesen nem szoftverenként, hanem feladatonként tagolt. Mivel nincsenek betűjelekkel megkülönböztetett meghajtók, egy fájl minden esetben ugyanott található meg.

A fájlrendszer gyökerét „/” karakterrel is jelöljük. Szintén a „/” karakter szolgál arra, hogy az elérési útvonalban a könyvtárakat elválasszuk egymástól. Amennyibe az elérési útvonal a „/” karakterrel kezdődik, abszolút, a fájlrendszer gyökerétől értelmezendő, egyébként relatív, az aktuális könyvtárhoz viszonyított.

Az aktuális könyvtárra a „.”, a szülő könyvtárra a „..” néven lehet hivatkozni. A felhasználók alapértelmezett, úgynevezett „home” könyvtárát a „~” néven lehet elérni.

A fájlrendszer felépítése a Filesystem Hierarchy Standard szabványt követi. A legalapvetőbb programok `/bin` és a `/sbin` könyvtárakban találhatóak, az előbbiben a minden felhasználó számára, az utóbbiban a csak a rendszer adminisztrátorának szükséges alkalmazások. Az ezeknek a programoknak szükséges függvénykönyvtárak állományai `/lib` könyvtárban találhatóak.

A legtöbb program a `/usr` könyvtárban található. Az `/usr/bin` és az `/usr/sbin` könyvtárakban a futtatható programok, az `/usr/lib` könyvtárban a bináris programok függvénykönyvtárai helyezkednek el. A `/usr/share` könyvtár a programok platformfüggetlen állományait tartalmazza. Ha valamelyik programnak a forráskódja is telepítve van, az a `/usr/src` könyvtárba kerül. Amennyiben egy függvénykönyvtárat a számítógépen lefordítandó programokban is használni akarunk, az `/usr/include` könyvtárba kerülnek azok a fejlécállományok, amelyekre az osztott könyvtárak állományain túl még ebben az esetben szükségünk van. Az `/usr/local` könyvtár alatt szintén megtalálhatóak a fenti alkönyvtárak, ezek az adott számítógépre egyedileg (jellemzően forrásból) telepített programokat tartalmazzák.

A `/usr` könyvtár tartalma a számítógép mindennapi használata során, ha nem telepítünk újabb programokat vagy távolítunk el illetve frissítünk meglévőket, nem változik; ezért a biztonság fokozása érdekében sokszor külön meghajtóra helyezik, és onnan csak olvasható opcióval csatolják fel a gyökér fájlrendszer üres `/usr` könyvtárába. Azok az állományok amelyek a programok normál működése közben is változhatnak, de nem kötődnek szorosan valamelyik felhasználóhoz (mert akkor annak a home könyvtárába kerülhetnének), a `/var` könyvtárban helyezkednek el. A `/var/log` könyvtárba kerülnek például az egyes folyamatok naplóállományai. A `/var` könyvtár alatt találhatóak egyes szerverfolyamatok futtatására használt felhasználók home könyvtárai is.

A `/home` könyvtárban találhatóak a felhasználók könyvtárai. Egy felhasználó minden adata (munkája során használt állományok, programok felhasználóra vonatkozó beállításait tartalmazó fájlok) ebben a könyvtárban

(pl. `/home/felhasznalo`) található. A root felhasználó könyvtára a `/root` könyvtár, így a rendszer adminisztrátora akkor is képes bejelentkezni és a legalapvetőbb programokat használni, ha a `/usr` és a `/home` könyvtárak tartalma a gyökértől eltérő meghajtón helyezkedik el, és azok valami miatt nem csatolhatóak fel.

Az ideiglenes állományok a `/tmp`, a `/var/tmp` vagy a felhasználó könyvtárában lévő tmp könyvtárba kerülhetnek.

Az egyes programok konfigurálását a `/etc` könyvtárban található konfigurációs állományok segítségével tudjuk elvégezni. A programok felhasználói szintű beállításai nem itt, hanem a felhasználó könyvtárán belül helyezkednek el.

A /dev könyvtárban találjuk az eszközfájlokat. Ezeken a speciális fájlokon keresztül férhetünk hozzá közvetlenül a számítógépnek az operációs rendszer által kezelt eszközeihez. Kétféle eszköz létezik: karakteres és blokk. Blokkok például az egyes merevlemezeket vagy azok partícióit leképező eszközfájlok. Ezek a leképezett eszköz méretével megegyező méretű állományok, melyek tartalmához bármilyen programmal hozzáférhet a megfelelő jogosultsággal rendelkező felhasználó. A karakteres eszközök a blokk eszközökkel ellentétben nem véletlen elérésűek, hanem egy input vagy output adatfolyamot képeznek le, mint például a számítógép soros vagy párhuzamos portja.

A `/boot` könyvtárban a rendszerbetöltő számára szükséges állományok kapnak helyet. Itt található meg általában Linux kernelt tartalmazó állomány is. Egyes esetekben a `/boot` könyvtárat külön partíción helyezik el, hogy tartalma a lemez elejére kerüljön.

A `/proc` könyvtárban a kernel és a futó programok egyes adatait érhetjük el. A gyökér fájlrendszer üres `/proc` könyvtárába ennek érdekében egy speciális fájlrendszert csatolnak fel. A `/proc/modules` állomány tartalmát lekérdezve megkapjuk a betöltött kernelmodulok listáját, a `/proc/1234/limits` fájl tartalmából pedig a 1234 sorszámú folyamat (hogy van-e ilyen, és hogy micsoda, az esetenként változik) számára beállított korlátokat.

Az alkalmilag felcsatolt fájlrendszereket (Ilyenek például cserélhető eszközök vagy más operációs rendszerek által is használt partíciók fájlrendszerei, vagy sokszor a hálózati fájlrendszerek) a `/mnt` vagy a `/media` könyvtáraiba szokás felcsatolni. Napjaink felhasználóbarát Linux disztribúciói egy USB mass storage eszköz (például egy pendrive) vagy egy DVD lemez behelyezését követően automatikusan létrehoznak egy alkönyvtárat és felcsatolják oda a kérdéses eszközön található fájlrendszert. Egy klasszikus Unix rendszerben ezt a feladatot egy a felhasználó által kiadott `mount` parancs végezné el, például a `mount -t vfat /dev/sdb1 /media/pendrive` paranccsal lehet felcsatolni az rendszer által másodlaogs SCSI eszközként emulált USB-s adathordozónk első (és egyetlen) FAT típusú partíciójának tartalmát a `/media/pendrive` könyvtár alá.

A fájlrendszerben lehetőségünk van linkek használatára is. Ennek leggyakrabban használt formája a szimbolikus link vagy más néven puha (soft) link. A szimbolikus linkek speciális állományok, amelyek egy másik állomány elérési útvonalát tartalmazzák. A link teljesen egyenértékűen használható azzal a fájllal vagy könyvtárral, amire hivatkozik. Az elérési útvonal, amire mutat, tetszőleges lehet, akár egy nem létező állományra is hivatkozhat, ami esetén természetesen nem lesz a szimbolikus link használható, de amint létrejön a megadott fájl, a linken keresztül elérhető lesz.

A linkek másik típusa a kemény (hard) link. Ez úgy működik, hogy egy fájlra több, egymással egyenrangú bejegyzés is mutat. A fájl törléséhet valamennyi rá mutató bejegyzést törölni kell. Kemény linket létrehozni a működési elvéből adódóan csak azonos fájlrendszeren lehetséges.

3.4. 15.3.4 Jogosultságok a fájlrendszerben

A Linux a többi UNIX-szerű operációs rendszerhez hasonlóan olvasási (r), futtatási (x) és írási (w) jogokat ismer amiket a fájl tulajdonosához (egy felhasználó), a fájl csoportjához (egy felhasználói csoport tagjai) és a többi felhasználóhoz rendelve tudunk meghatározni.

Az olvasás a fájl vagy a könyvtár tartalmának olvasását jelenti. Az írási jog ahhoz szükséges, hogy módosítani tudjuk a tartalmát vagy törölni tudjuk. A futtatás szükséges ahhoz, hogy egy végrehajtható kódot tartalmazó állományt a felhasználó elindítson, vagy egy könyvtárba beléphessen.

Egy adott műveletet egy felhasználó egy fájlon akkor tud végrehajtani, ha:

- a felhasználó a root felhasználó

- vagy a felhasználó a fájl tulajdonosa és a fájl tulajdonosa rendelkezik a szükséges jogosultsággal
- vagy a felhasználó tagja a fájl csoportjának és a fájl csoportja rendelkezik a szükséges jogosultsággal
- vagy a többi (other) felhasználó rendelkezik a szükséges jogosultsággal

A háromféle jogosultság (írás, olvasás, futtatás) és a háromféle jogosult (tulajdonos, csoport, többiek) kombinációiból kilenc egymástól függetlenül beállítható jog keletkezik.

Külön speciális jogosultság a setuid, amivel a fájl tulajdonosa azt tudja engedélyezni, hogy más, a fájl futtatására jogosult felhasználók a fájlba található programot az ő nevében futtathassák. Ez lehetőséget nyújt arra, hogy egyes, csak a root által elvégezhető műveleteket más felhasználók is végrehajthassanak, de egyben egy komoly biztonsági kockázatot is jelent, amennyiben az engedélyezett programban valamilyen kihasználható hiba van, aminek következtében a program eredetileg nem tervezett műveletek végrehajtására is használható.

A Linuxban nincs az állományoknak rejtett vagy rendszer attribútuma. Ha azt szeretnénk, hogy egy állomány rejtett állományokhoz hasonlóan viselkedjen, a nevét ponttal kell kezdeni. A ponttal kezdődő nevű állományokat egyes felhasználói programok megfelelő beállítások mellett nem jelenítik meg, így nem zavar például a home könyvtárunkban elhelyezkedő sok speciális állomány és könyvtár egy dokumentum kiválasztásakor.

3.5. 15.3.5 Felhasználók kezelése

A Linux többfelhasználós operációs rendszer. A rendszert használni, a programokat futtatni többféle felhasználóval, felhasználónként eltérő jogosultságokkal lehetséges.

Többfelhasználós rendszerek egyik fontos kérdése az autentikáció, vagyis a felhasználók megbízható azonosítása. Ez általában a felhasználónév és a jelszó megadásával történik egy karakteres vagy grafikus felületű beléptető program segítségével.

A másik fontos feladata egy többfelhasználós rendszernek az autorizáció, vagyis annak eldöntése, hogy a felhasználónak (a felhasználó által futtatott programoknak) milyen műveletekhez van jogosultsága. Ez a UNIX rendszerek minden fájl filozófiájával összhangban általában fájlok jogosultságain keresztül szabályozható a fájlrendszernél bemutatott módon, esetleg egyes programok esetében konfigurációs fájlokon keresztül lehet jogosultságokat beállítani.

A root (gyökér) felhasználó minden jogosultsággal rendelkezik. Ezt a felhasználót a rendszer adminisztrátora szokta használni. Egyes a háttérben futó folyamatok részére külön felhasználókat és felhasználói csoportokat hoznak létre, mert amennyiben nem feltétlenül szükséges, biztonsági okokból nem célszerű azokat a minden jogosultsággal rendelkező root felhasználó nevében futtatni.

A felhasználó és a csoport neve helyett a háttérben egy sorszámot használ a rendszer. Ezt a sorszámot tárolja a fájlok jogosultságaival kapcsolatban is.

4. 15.4 Használat parancssorból

A számítógép parancssorból történő használata egy egyszerű felületen nyújt sokféle lehetőséget a hozzáértő felhasználó számára. Az átlagos felhasználók manapság már ritkán használják, de a rendszer adminisztrálását vagy az összetettebb adatfeldolgozási és programozási feladatokat végző szakemberek számára nagyon hasznos tud lenni.

4.1. 15.4.1 A karakteres felület alapelve és fajtái

Karakteres üzemmódban a képernyőt nem pixelenként, hanem karakterenként kezeljük. A képernyőn ilyenkor meghatározott számú (például 80 karakter 25 sorban) karakter jeleníthető meg egy meghatározott karakterkészlet elemeiből, megadható színnel és háttérszínnel.

A képernyő karakteres felületén egy a grafikus felülethez hasonló környezetet is kialakíthatunk karakterekből összerakott ablakokkal és kezelőfelületi elemekkel. Az így létrejövő szöveges felhasználói felületen akár még az egeret is grafikus felületen megszokotthoz hasonlóan tudjuk használni.

A másik lehetőség a karakteres felület használatára a parancssor. A parancssorba egy felvezető karaktersorozat (a prompt) után utasításokat tudunk beírni a billentyűzet segítségével. Az utasítások végrehajtásának eredménye a képernyő soron következő részére íródik ki, a képernyő tartalma szükség esetén tovább gördül.

A Linux operációs rendszerben a képernyő karakteres üzemmódját virtuális terminálok segítségével szokás használni. Több virtuális terminál osztozhat egyetlen fizikai felületen (képernyő és billentyűzet), amelyek között az ALT billentyű és a terminál sorszámának megfelelő számú funkcióbillentyű egyidejű lenyomásával lehet váltani.

Bár grafikus felületről is át lehet váltani a karakteres üzemmódban működő virtuális terminálok valamelyikére a CTRL, az ALT és a terminál sorszámának megfelelő számú funkcióbillentyű egyidejű lenyomásával; ilyenkor inkább egy terminál ablakot célszerű használni. Ez a program a grafikus felület egy ablakában nyújt egy, vagy akár több karakteres felületet. Ha több karakteres felület is fut egy ablakban, akkor a böngészőprogramokban már megszokott módon, fülek segítségével választhatunk közöttük.

A karakteres felület hálózati adatátvitel segítségével megjelenhet egy másik számítógépen is. Ehhez a számítógépen futtatni kell a megfelelő szolgáltatást. Az adatátvitel régebben főleg a Telnet protokoll segítségével történt, manapság erre a célra szinte kizárólag az adatfolyam titkosítását is biztosító ssh protokollt alkalmazzák. A kliens lehet egy másik UNIX-szerű operációs rendszer parancssorában elindított alkalmazás vagy például egy Windows operációs rendszer alatt elindított Putty program.

4.2. 15.4.2 A shell program

A karakteres felületen futó, a parancsokat értelmező és végrehajtó programot shell-nek vagy magyarul héjprogramnak szokás nevezni. Az elnevezés arra utal, hogy a program mintegy héjat képez az operációs rendszer magján, a kernelen.

Többféle shell program létezik, a következőkben a leginkább elterjedtnek mondható Bourne shellt mutatjuk be. A program a `/bin/sh` állományban található.

A parancssorban kiadható utasítások általában programok, amelyeknek egy paraméterlistát is átadhatunk. A parancs első része (az első szóközig) a programot határozza meg, a további pedig a paraméterlistát adja.

A shell rendelkezik változókkal. Ezek értékét a nevüket követő egyenlőségjel után tudjuk megadni. Az értéküket bármilyen parancsban lekérdezhetjük, amihez „$” szimbólumot kell a nevük elé tenni. Például:

```
valtozo=szoveg
echo $valtozo
```

A fenti példában először a „szoveg” karakterláncot helyezzük el a „valtozo” nevű változóban, majd ennek a változónak az értékét (vagyis azt, hogy „szoveg”) kiírjuk a képernyőre. Érdemes megfigyelni, hogy értékadáskor nem tettünk dollárjelet a változó neve elé, csak akkor, amikor az értékét lekérdeztük. Meg kell még jegyezni, hogy a shell változók azonosítói érzékenyek a kis és nagy betűk közötti különbségekre.

Vannak olyan változók, amelyek értékének hatása van a shell vagy az abból indított programok működésére. Ilyen például a `$PATH`, ami azt határozza meg, hogy az elérési útvonal nélkül megadott utasításokhoz mely könyvtárakban keressen megfelelő programot. A `$PATH` változó értéke kettősponttal elválasztva tartalmazza a könyvtárak elérési útvonalait, a helyi könyvtár (`./`) biztonsági okokból nincs közöttük.

A futtatott programok szöveges kimenete (amit egy C nyelvű program `printf` függvénye, egy Pascal program `writeln` utasítása, egy Ruby-ban írt alkalmazás `puts` metódusa, vagy egy shell program `echo` parancsa ír ki a képernyőre) arra a karakteres eszközre kerül, ahol a shell fut. A program kimenetét lehetőségünk van átirányítani egy fájlba a „>” karakter segítségével, például a következő módon:

```
ls &gt; fajljaim.txt
```

Az `ls` parancs kiíratja az aktuális könyvtár tartalmát, de az nem a képernyőre, hanem `fajljaim.txt` állományba kerül. Ha már létezik ilyen nevű állományt azt ezzel az utasítással felülírjuk. Amennyiben ilyen esetekben a létező állomány végére akarjuk fűzni a program kimenetét a „>” helyett a „>>” használata szükséges.

Vannak programok, amelyek nem csak kimenettel, hanem bemenettel is rendelkeznek. Egyszerűen elindítva egy ilyen programot, a bemenetét a billentyűzet segítségével tudjuk bevinni, a `CTRL+D` billentyűkombinációval lezárva. Amennyiben a bemenetként megadni kívánt szöveg egy állományban már rendelkezésre áll, a fájlt a „<” karakter segítségével tudjuk a program bemenetére irányítani.

Gyakran előfordul, hogy egy program kimenetét egy másik program bemenetére szeretnénk átirányítani. Erre a két program közé elhelyezett „|” karakter (pipe, vagy csővezeték néven szokás nevezni) segítségével nyílik lehetőség.

A shell képes az úgynevezett shell programok végrehajtására. Ezek a programok szöveges állományokban helyezkednek el. A szöveges állomány első sora az értelmezőt határozza meg „`#!`” karaktereket követően, vagyis a mi esetünkben a `#!/bin/sh` szöveget tartalmazza. Egyébként hasonló módon kell eljárni más interpreteres nyelveken írt programok esetében is, hogy egy Linux alatt végrehajtható állományt kapjunk. (A Ruby nyelven írt programok esetében például az első sorba a `#!/usr/bin/ruby` szövegek kell írnunk, így a rendszer tudni fogja, hogy a programot a `/usr/bin/ruby` útvonalon elérhető értelmezővel lehet futtatni.) A program futtathatóságához fontos továbbá, hogy az állomány végrehajtható legyen.

A shell programok sorai többségében olyan utasítások, amelyeket a parancssorba begépelve egyenként is ki lehetne adni. Ezeken felül tartalmazhat még vezérlési szerkezeteket (elágazások, ciklusok) is.

4.3. 15.4.3 Fontosabb parancsok

Az egyes programokat nevüknek (a programot tartalmazó fájl nevének) a parancssorba írásával tudjuk elindítani. Amennyiben a fájl elérési útvonala nem szerepel a `$PATH` változóban, úgy azt is meg kell adni.

A program nevét a programnak szánt paraméterek követik. A paraméterek lehetnek fájlokra való hivatkozások vagy kapcsolók. A kapcsolókat egy vagy két kötőjellel kezdve szokás megadni. Egy kötőjellel az egybetűs, további opciókat nem igénylő kapcsolókat szokás kezdeni, amelyek így halmozhatóvá válnak, például az `ls -l -a` helyett használhatjuk az `ls -la` alakot.

Az alábbiakban röviden bemutatjuk a legfontosabb programokat, amelyeket a parancssorban használni szoktunk.

- `cd`: az aktuális könyvtár megváltoztatása
- `mkdir`: új könyvtár létrehozása
- `cp`: állományok másolása másik nevű állományba vagy másik könyvtárba
- `rm`: állományok törlése
- `mv`: állományok átnevezése vagy átmozgatása egy másik könyvtárba
- `ln`: szimbolikus (puha) link létrehozása egy állományhoz vagy egy könyvtárhoz
- `ls`: könyvtár tartalmának kilistázása
- `find`: állományok keresése
- `chmod`: állományok jogosultságainak beállítása
- `chown`: állományok tulajdonosának beállítása
- `chgrp`: állományok csoportjának beállítása
- `du`: egy könyvtár (és alkönyvtárai) összes állománya által foglalt összes hely kiszámítása
- `df`: a lemezek kapacitásának, használt és szabad területének kilistázása
- `echo`: egy tetszőleges szöveg kiírása
- `cat`: egy szöveges állomány tartalmának kiíratása

- `head`: egy szöveges állomány első sorainak kiíratása
- `tail`: egy szöveges állomány utolsó sorainak kiíratása
- `grep`: egy szöveges állomány azon sorainak kiíratása, amelyek megfelelnek egy megadott feltételnek
- `sort`: egy szöveges állomány sorainak kiíratása sorba rendezést követően
- `mount`: egy új fájlrendszer felcsatolása egy üres könyvtárba
- `umount`: egy fájlrendszer lecsatolása
- `ps`: futó folyamatok kilistázása
- `kill`: egy folyamat megszakítása
- `passwd`: a felhasználó jelszavának megváltoztatása
- `su`: egy másik felhasználó szerepének átvétele (amennyiben nem a root felhasználó nevében adjuk ki, a kérdéses felhasználó jelszavának megadását követően)
- `sudo`: egy utasítás futtatása egy másik felhasználó nevében (amennyiben nem a root felhasználó nevében adjuk ki, a kérdéses felhasználó jelszavának megadását követően)

Az egyes programokról a `man` program segítségével tudunk egy részletes leírást kérni. Például a `man grep` utasítást a parancssorba gépelve a `grep` program használatát leíró kézikönyvoldalt tudjuk olvasni.

5. 15.5 Grafikus felület

A Linuxban a grafikus felület nem része a rendszermagnak, és nem feltétlenül része az operációs rendszernek, mivel sok esetben (szerverek, beágyazott rendszerek) nincsen rá szükség.

Amennyiben van grafikus felület, az többféle, a Linux rendszermagjától függetlenül fejlesztett programból épül fel.

5.1. 15.5.1 Az X Window System

A Linuxban a grafikus felület kezelésére az X Window Systemet, rövidítve X11-et, vagy csak egyszerűen X-et használják. Az X szerver-kliens elven épül fel, az X szerver kezeli a grafikus felületet, a grafikus felületet igénylő programok kliensként fordulnak hozzá a felhasználói felületük megjelenítése érdekében.

A kliens-szerver felépítésnek köszönhetően a program akár egy másik számítógépen is futhat, mint amelyiken a grafikus felülete megjelenik. Főként régebben volt elterjedt szokás, hogy a karakteres terminálok mintájára X szervert futtató grafikus terminálokat használtak távoli nagy teljesítményű számítógépeken futó programok kezelésére.

Az X csak a grafikus környezet felépítéséhez szükséges elemi műveleteket biztosítja. Lehetővé teszi az ablakok kirajzolását és mozgatását és a beviteli eszközök (billentyűzet, egér, egyéb pointerek, stb.) kezelését, de önmagában még nem nyújt egy felhasználói felületet, ahhoz egy ablakkezelő rendszerre is szükségünk van.

A karakteres héjprogramokhoz hasonlóan az X-ből is többféle létezik, mint például az Xfree86 vagy az X.org.

5.2. 15.5.2 Ablakkezelő rendszerek

Mivel az X Window System csak a grafikus felület létrehozásához szükséges elemi műveleteket tartalmazza, szükségünk van még egy ablakkezelő rendszerre is. Az ablakkezelő rendszer felel az ablakok tartozékainak (keret és fejléc az ablak átméretezését vagy bezárását lehetővé tévő szokásos gombokkal) kirajzolásáért, a felhasználói felületnek programok indítását és nyomon követését lehetővé tévő részeinek (indítómenük és ikonok, tálcák) kezeléséért, valamint ezen kívül még számos a számítógép használatát megkönnyíteni vagy hatékonyabbá tenni hivatott segédprogramot biztosít.

A legtöbb Linux alatt használt ablakkezelő rendszer több munkaasztal párhuzamos kezelését teszi lehetővé. Ez úgy működik, mintha több képernyőnk lenne, amelyek mindegyikén szabadon tudjuk a futó programjaink ablakait elrendezni. Azt, hogy melyik munkaasztal jelenjen meg a képernyőnkön, az ablakkezelőnek a virtuális munkaasztalok közötti váltást biztosító részén tudjuk kiválasztani, ami jellemzően a munkaasztalok kicsinyített képeit tartalmazó gombokból áll.

Ablakkezelő programokból bőséges választék áll a rendelkezésünkre. A legismertebbek a KDE és a Gnome.

6. 15.6 Disztribúciók

Mivel egy Linux-alapú operációs rendszer sokféle külön fejlesztett (de a nyílt forráskódú licenszeknek köszönhetően szabadon hozzáférhető és felhasználható) programból épül fel, használatához ezeknek a programoknak egy gyűjteményére és a telepítésüket valamint beállításukat elvégző további alkalmazásokra van szükség. Az ilyen módon összeálló szoftvert disztribúciónak nevezzük.

Sokféle Linux disztribúció létezik, vannak közöttük informatikai nagyvállalatok által fejlesztett milliárdos piaccal rendelkezők és hobbiból készítettek egyaránt. Készülnek olyan disztribúciók is amelyeket kifejezetten egy adott alkalmazási területre vagy eszközre fejlesztenek. Sok disztribúció egy másikra épül, attól több vagy kevesebb módosítással tér el.

A disztribúciók változatait a többi szoftverhez hasonlóan verziószámokkal szokás ellátni, a disztribúció pontos meghatározásához ezért ezt is meg kell adni.

A disztribúciók sokféle programot tartalmazhatnak. A Linux-kernelen és a gép használatához szükséges alapvető programokon túl általában megtalálható bennük számos felhasználói program is, mint például az OpenOffice.org irodai programcsomag, a Firefox webböngésző, a GIMP képszerkesztő, a Scribus kiadványszerkesztő vagy a Blender 3D modellező programok. Telepítést követően általában egy teljes körűen használható rendszer kapunk.

6.1. 15.6.1 Csomagkezelők

A disztribúció egyik legfontosabb feladata a számítógépen található Linux-alapú rendszer programjainak telepítése és beállítása. Ezt a feladatot a csomagkezelő program végzi, ami a programhoz tartozó csomagból a megfelelő helyekre másolja a programot képező fájlokat, majd szükség esetén elvégzi a program beállítását.

A csomagok között függőségek lehetnek, egy csomag telepítése más csomagok telepítését igényelheti, amelyeknek szintén lehetnek hasonló függőségei. A csomagkezelő programok képesek kezelni a függőségeket, a felhasználó által igényelt csomag mellett telepítik a közvetlenül vagy közvetve szükséges további csomagokat is.

A csomagokat a csomagkezelő a disztribúció telepítő CD vagy DVD lemezéről, vagy egy meghatározott hálózati helyről érheti el. Ez utóbbi esetben új csomagok telepítésén túl lehetőség nyílik a telepített csomagok újabb változatainak figyelésére és szükség esetén az érintett programok frissítésére.

A csomagkezelés történhet parancssori eszközökkel, de a legtöbb disztribúció rendelkezik grafikus felületű, egyszerűen használható csomagkezelő programmal is. Egy ilyen programban csak ki kell választani a telepítendő alkalmazásokat, a csomagkezelő ezután elvégzi a szükséges csomagok letöltését és telepítését.

A két leginkább elterjedtebb csomagformátum a Debian rendszerek .deb és a RedHat .rpm formátuma. Ezeket az eredetiken túl számos további disztribúció használja (az .rpm-et például a Suse, a .deb-et az Ubuntu).

7. 15.7 Összefoglalás

Ellenőrző kérdések:

- Milyen technológiai előzményei voltak a Linux operációs rendszernek?
- Melyek egy operációs rendszer kerneljének legfontosabb feladatai?
- Melyek azok a fontos alapfeladatok, amelyeket egy Linux-alapú operációs rendszerben nem a kernel végez?

- Hogyan épül fel a Linux-alapú operációs rendszerek fájlrendszere?
- Hogyan érjük el Linux alatt a számítógépre csatlakoztatott ideiglenes háttértárak (például egy pendrive) tartalmát?
- Milyen fontos eltérést tapasztalunk a könyvtárak elérési útvonalának megadásakor a Windowsban megszokotthoz képest?
- Milyen jogosultságok kapcsolódhatnak a fájlokhoz és könyvtárakhoz?
- Milyen elven működik a karakteres felület?
- Mire használhatók a shell (héj) programok?
- Ismertessen néhány parancssorból használható programot!
- Milyen elven működik a grafikus felület egy Linux-alapú operációs rendszerben?
- Hogyan különülnek el az X Window System és az ablakkezelő programok feladatai?
- Milyen feladatokat oldanak meg a csomagkezelő programok? Melyek a leggyakrabban használt csomagformátumok?
- Soroljon fel néhány Linux disztribúciót!

Irodalomjegyzék

Pere László: Kiskapu Kft., Budapest, 2005.

Marcel Gagné: Kiskapu Kft., Budapest, 2002.

Büki András: Kiskapu Kft., Budapest, 2002.

http://tldp.fsf.hu/

http://kernel.org/

http://www.slackware.com/

http://www.ubuntu.com/

Fedora honlapja: http://fedoraproject.org/

http://www.redhat.com/