Linux Sistem Yöneticisinin Kılavuzu

Sürüm 0.8

Çeviren:
Yalçın Kolukısa
<yalcink01 (at) yahoo.com>

Hazırlayan:
Nilgün Belma Bugüner
<nilgun (at) belgeler·gen·tr>

Yazan:
Lars Wirzenius
<liw (at) iki.fi>

Yazan:
Joanna Oja
<viu (at) iki.fi>

Yazan:
Stephen Stafford
<stephen (at) clothcat.demon.co.uk>

Yazan:
Alex Weeks
<weeks_alex (at) yahoo.com.NOSPAM>

27 Şubat 2007

Özet

Yeni başlayanlar için Linux sisteminin sistem yönetimine bir giriş.

İçindekiler

Bu çevirinin sürüm bilgileri:

0.2	Ocak 2004	YK
0.1	Ocak 2003	YK, NBB

Çeviri, Yalçın Kolukısa tarafından .doc biçemli olarak yapılmış, Nilgün Belma Bugüner tarafından gözden geçirilip, çevrilmemiş kısımları tamamlanmış (bazı terimler, konsol çıktıları ve resimler) ve XML biçemine dönüştürülmüştür.

Özgün belgenin sürüm bilgileri:

0.8	2003/12/03 08:58:41 -500	Stephen Stafford ve Alex Weeks
0.7	2001/11/06 11:26:32	Lars Wirzenius, Joanna Oja ve Stephen Stafford

Yasal Uyarı

Legal Notice

Belgenin Özgün Sürümü

Kitabın kaynak kodu ve diğer belge biçimleri Linux Belgelendirme Projesi (LDP) sitesinde (http://www.tldp.org/) ve bu kitabın ev sayfasından (http://www.taylexson.org/sag/) HTML ve PDF biçimlerinde edinilebilir.

1. Giriş

> "Başlangıçta dosyalar boş ve şekilsizdi. Ve Yazarın Parmakları klavyenin üzerinde gezindi. Orada kelimeler olmasını istedi ve kelimeler oldu."

Linux Sistem Yöneticisinin Kılavuzu, sistem yöneticilerinin Linux kullanımını anlatmaktadır. Bu belge sistem yönetimi hakkında hemen hemen hiçbir şey bilmeyen ama temel yönetim komutlarını ve mantığını bilenler için hazırlanmıştır. Bu kılavuz Linux kurulumunu anlatmaz. Bu tür bilgiler Kurulum ve Başlangıç Kılavuzunda bulunmaktadır. Aşağıda daha ayrıntılı bilgi yer almaktadır.

Sistem yönetimi, bir bilgisayar sistemini kullanılabilir ve stabil tutmak için gereken her türlü faaliyeti kapsamaktadır. Bu : dosyalarının yedeklenmesi ve geri yüklenmesini; yeni programlara yüklenmesini; yeni kullanıcılar eklenmesi ve eskilerin silinmesini; dosya sistemlerinin sağlamlığının sağlanmasını; vb... olayları kapsamaktadır. Eğer bir bilgisayarı bir ev olarak kabul edersek; sistem yöneticisini bu evin kahyası ve hizmetçisi olarak tanımlayabiliriz. Evin temizlik, bakım–onarım ve benzeri bütün işlerinin sorumluluğu sistem yöneticisine aittir.

Bu kılavuz birbirinden bağımsız bölümlerden oluşmaktadır. Şayet sadece yedekleme ile ilgileniyorsanız, doğrudan o bölüme geçebilirsiniz. Bununla beraber bu kılavuzun bir ilk olduğu ve bir ders notları şeklinde hazırlandığı unutulmamalıdır. Bu kılavuzu ister bölümler bazında ele alın, isterseniz bir bütün olarak okuyun.

Bu kılavuz tamamen bağımsız bir kullanım için tasarlanmamıştır. Diğer Linux belgelerinde de Sistem Yöneticileri için çok faydalı bilgiler bulunmaktadır. Sistem yöneticisi bazı özel görev ve yetkileri olan bir kullanıcıdır. Başvurabileceğiniz çok sayıda faydalı kılavuz sayfaları bulunmaktadır. Buralardan komutlar hakkında ayrıntılı bilgiler almanız mümkündür. ayet hangi komuta ihtiyacınız olduğunu bilmiyorsanız **apropos** komutunu kullanabilirsiniz. Bu komut hakkında ayrıntılı bilgi Linux kılavuz sayfalarında mevcuttur.

Bu kılavuz Linux işletim sistemi için tasarlanmakla birlikte, Unix tabanlı diğer işletim sistemleri içinde kullanılabilir. Unix sistemlerinin genel özellikleri ile sistem yöneticiliğinin bazı özellikleri arasında tam bir uyum olmadığı için, bu kılavuzun bütün Unix tabanlı işletim sistemlerini kapsadığı söylenemez. Linux işletim sistemlerinin doğal gelişiminden dolayı, bu kılavuzun bütün Linux sistemlerini kapsaması da çok zordur.

Resmi bir Linux sistemi olmadığından ve dağıtımların çok çeşitli olmasından dolayı bu kılavuz her hangi bir Linux Sisteme yönelik yazılmamıştır. Mümkün olan durumlarda çeşitli dağıtımlar arasındaki farklılıklar bu kılavuzda belirtilmiştir.

Bu kılavuz olayları "5 temel adım" gibi sınıflandırmalar yapmadan, bütün ayrıntıları ile anlatmaktadır. Bu yüzden bazı bölümler herkesin işine yaramayabilir. Bu bölümleri atlamakta kendinizi serbest hissedebilirsiniz. Fakat burada yazılmış olan bütün bölümleri okumak, doğal olarak, sistem hakkındaki bilgilerinizi daha ayrıntılı ve kuvvetli hale getirecek ve yöneticilik işiniz daha kolaylaşacaktır.

Linux bağlantılı bütün geliştirme ve dökümasyon işlerinde olduğu gibi, bu kılavuzun yazım işi tamamen gönüllük esasına dayanılarak yazılmıştır. Bunun eğlenceli olacağını ve olması gerektiğini düşündüğüm için bu kılavuzu hazırladım. Bununla birlikte, her gönüllü işte olduğu gibi, burada da kısıtlı zaman, bilgi ve deyenim söz konusudur. Bu kılavuzun para karşılığı hazırlanmış olanlar kadar profesyonel olduğunu düşünmeyin. Benden uyarması.

Bir şeyi daha belirtmek isterim ki; gönüllü olarak hazırlanmış son derece kapsamlı ve iyi kılavuz sayfaları İnternet üzerinde mevcuttur ve onlar bu kılavuzun konusu dışında bırakılmıştır. Program ve komutlara yönelik ayrıntılı anlatımlara bu kılavuzda yer verilmemiştir. Sadece bu programların temel kullanımına ve özellikleri burada

anlatılmıştır. Ayrıntılı bilgi almak isterseniz bu kılavuz sayfalarına baş vurmanız gerekmektedir. Genellikle bu kılavuzlar Linux belgelendirme çalışmalarının bir parçasıdır.

2. Bu Kitap Hakkında

2.1. Teşekkürler

2.1.1. Joanna'nın Teşekkürleri

Lars bu kılavuzun mümkün olan en iyi şekilde çıkartılması için çok uğraştı. Bu işin temel sorumlusu olarak, ben de en iyiyi elde etmek ve onu korumak isterim. Bu kılavuzu daha iyi hale getirmek için bir fikriniz var ise lütfen bana bunu bildirin. Dilbilgisi hataları, gerçeklere dayanmayan bilgiler, eklenmesini veya tekrar yazılmasın istediğinizi bölümler hakkında lütfen bana yazın. Erişim bilgilerimi http://www.iki.fi/viu/ adresinde bulabilirsiniz.

Bu kitabın yazımına pek çok insan dolaylı ve dolaysız yollardan yardım ettiler. LDP gibi muhteşem bir fikre öncülük ettiği için Matt Welsh'e özellikle teşekkür etmek isterim. Andy Oram'a o çok değerli destekleri için teşekkür ederim. Olaf Kirch'e bu işin yapılabileceğine beni inandırdığı için teşekkür ederim. Yggdrasil'deki bu işi ilginç bulan bütün insanlara ve Adam Richter'e teşekkür ederim.

Stephen Tweedie, H. Peter Anvin, Remy Card, Thedore Ts'o kendi çalışmalarını bana ödünç vermiş ve bu kitabın daha kalın ve etkileyici bir görünüme kavuşmasını sağlamışlardır. Bu eserler: xia ve ext2 dosya sistemleri hakkında karşılaştırma, ext2 dosya sisteminin tanımı ve araç listeleridir. Bunun için kendilerine minnettarım. Eski sürümlerdeki bu eksiklik için özür dilerim.

Ek olarak Mark Komarinski'ye 1993'deki materyalleri ve Linux Journal'da yer alan sistem yönetimi hakkındaki bilgileri bana ulaştırdığı için çok teşekkür ederim. Bu belgeler bilgilendirici ve ilham verici olmuştur.

Çok sayıda insan tarafından çok faydalı yorumlar gönderildi. Minyatür arşivim hepsinin ismini bulmama izin vermiyor ama alfabetik olarak bazı isimler şunlardır: Paul Caprioli, Dave Dobson, Ales Cepek, Olaf Flebbe, Marie–France Declerfayt ve babası, Stephan Harris, Jyrki Havia, Jim Haynes, York Lon, Timothy Andrew Lister, Jim Lynch, Michael J.Micek, Jacob Navia, Dan Poirier, Daniel Quinlan, Jouni K. Seppanon, Philippe Steindi, G. B. Stotte. Unuttuğum herkesten özür dilerim.

2.1.2. Stephen'ın Teşekkürleri

Yeni sorumlu olarak Lars ve Joanna'ya bu kılavuz üzerine yapmış oldukları sıkı çalışmalardan dolayı teşekkür ederim. Bütün kılavuzlarda olduğu gibi bu kılavuzda da bazı hatalar bulunmaktadır. Ve zaman zaman bazı bölümler güncelliğini yitirmektedir. Bu tür hatalara rastlarsanız lütfen <bagpuss (at) debian.org> adresinden bana bildirin. HTML, salt metin ya da her neyse hemen bütün biçemlerde eposta kabul ediyorum.

Helen Topring Shaw'a bu kılavuzu çok daha iyi bir hale getirdiği için sonsuz teşekkürlerimi sunarım.

Bu kılavuzun ana web sayfasının adresi: http://people.debian.org/~bagpuss/

2.1.3. Alex'in Teşekkürleri

Lars'a, Joanna'ya ve Stephen'a bu kılavuz için senelerden beri yaptıkları muhteşem çalışmalar için teşekkür etmek isterim. Umarım başlattıkları işi gereginince devam ettirebilirim.

"Penceresiz Bir Dünyaya" doğru yapmış olduğum bu yolculukta pek çok kişinin yardımlarını gördüm. Özellikle, benim ilk gerçek Un*x satıcım Mike Velasco'naya teşekkür etmek isterim. SCO'nun "pis bir kelime" olmasından önce, Mike bana, **tar**, **cpio** ve diğer pek çok man sayfaları konusunda oldukça yardımcı olmuştu. Teşekkürler Mike! Sen gerçek "Sofa King"sin.

2.2. Yazım Düzeni

Okunurluğu kolaylaştırmak için bu kılavuzda tek tip bir yazım düzeni kullanılmıştır. Şayet daha iyi fikirleriniz olursa lütfen bildirmekten çekinmeyin.

Dizin/Dosya isimleri: `/usr/share/doc/foo`

Komutlar ve programlar: **`fsck`**

Eposta adresleri: <yalcink01 (at) yahoo.com>

URL'ler: `http://www.tldp.org`

3. Linux Sistemine Genel Bakış

> "Oku! Kalemle öğreten, insana bilmediğini bildiren Rabbin, en büyük kerem sahibidir." – Kur'an–ı Kerim

Bu bölüm Linux işletim sistemi hakkında genel bir bilgi vermektedir. Öncelikle işletim sistemi tarafından yapılan temel işlevler tanımlanmaktadır. Daha sonra fazla ayrıntıya girmeden bu servislerin işletilmesini sağlayan programlar anlatılmıştır. Bu bölümün amacı sistem hakkında bütünsel bir şekilde bilgi vermektir. Bu sebeple her bölüm hakkındaki ayrıntılı bilgi başka yerlerde bulunmaktadır.

3.1. İşletim sisteminin çeşitli parçaları

Bir Unix işletim sistemi çekirdek ve bazı sistem programlarından oluşur. Bazı uygulama programları da vardır. Çekirdek işletim sisteminin kalbidir. [1] Diskteki dosyaların izlerini tutar, programları başlatır ve yürütür, belleği ve çeşitli süreçlerin kaynaklarını düzenler, ağdan paketleri alır ve gönderir, vb... Çekirdek kendi başına çok az iş yapar, fakat diğer servislerin kullanabileceği araçları sağlar. Ayrıca donanımlara doğrudan ulaşan kişileri önleyerek, onları kendi sunduğu araçları kullanmaya zorlar. Bu yolla çekirdek, kullanıcıları diğer kullanıcılara karşı koruyacak bir yol izler. Çekirdek tarafından sağlanan bu araçlar sistem çağrıları üzerinden kullanılır. Sistem programları işletim sisteminin ihtiyacı olan çeşitli servisleri yerine getirmek için çekirdek tarafından sağlanan bu araçları kullanırlar.

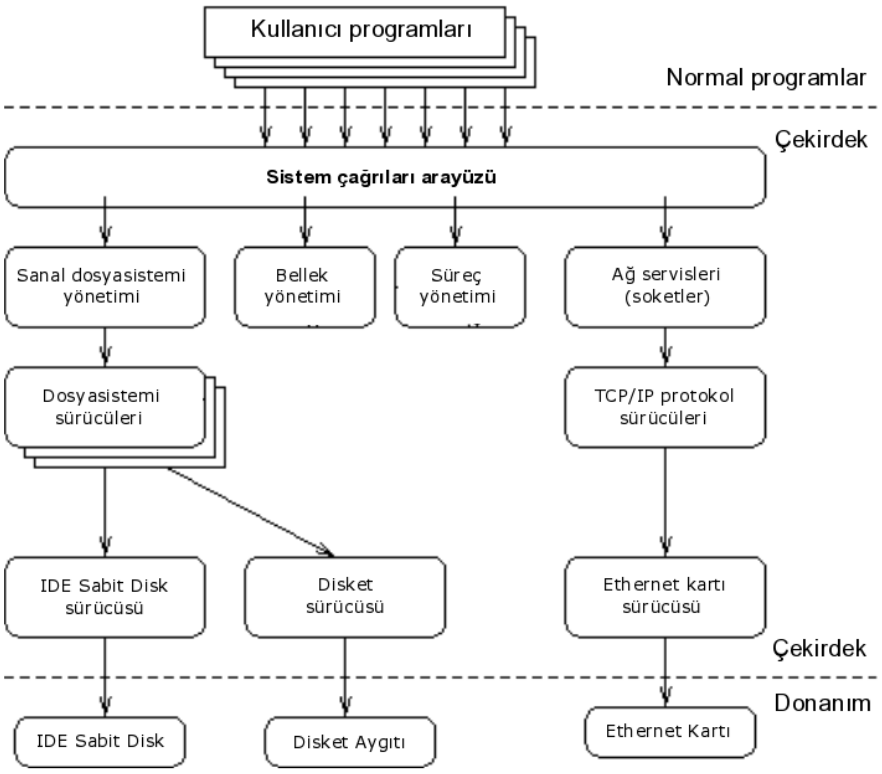
Sistem programları ve diğer bütün programlar kullanıcı kipi olarak adlandırılan 'çekirdeğin en üst bölgesi' olan yerde çalışırlar. Sistem programları ve uygulama programları arasındaki en önemli fark: sistem programlarının işletim sisteminin çalışması için gerekli olan yazılımlar olması, uygulama yazılımları ise sistemde faydalı ve eğlenceli işlerin (oyun ,vb..) yapılmasını sağlıyor olmasıdır. Kelime işlemcisi bir uygulama programıdır. **`mount`** ise bir sistem programıdır. Bunlar arasındaki farklar sadece sınıflama işlemleri için önemlidir.

İşletim sistemleri bazı derleyicileri ve onların uygun kütüphanelerini (Linux altındaki GCC ve C++ kütüphaneleri gibi) ihtiva edebilirler. Bununla beraber bütün programlama dilleri bir işletim sisteminin parçası olmak zorunluluğunda değildirler. Belgeler ve bazen oyunlar da bir işletim sisteminin parçaları olabilirler. Geleneksel olarak işletim sistemleri kurulum teybi veya disklerinin içerisindeki programlar olarak tanımlanabilir; bu tanım içerisinde Linux'ün durumu pek berrak değildir. Çünkü Linux, bütün dünya üzerinde FTP sitelerinden de yayılmaktadır.

3.2. Çekirdeğin önemli parçaları

Linux çekirdeğinin çeşitli önemli bölümleri vardır: işlemci ve bellek yönetimi, donanım ve dosya sistemleri sürücüleri, ağ yönetimi ve çeşitli parçalar. Şekil 3.1 de bunlardan bazıları görünmektedir.

Şekil 1. Linux çekirdeğinin bazı önemli parçaları



Büyük ihtimalle çekirdeğin en önemli parçaları (onlar olmadan hiçbir şey çalışmaz) işlemci ve bellek yönetim bölümleridir. Bellek yönetimi, bellek modüllerinin alanlarını ve takas alanını, süreçler ile çekirdeğin bölümleri ve tampon önbelleği için ayarlar. Süreç yönetimi ise süreçleri yaratır, işlemcide etkin olan süreçleri değiştirerek çokgörevliliği sağlar.

Çekirdeğin en alt seviyesinde, çekirdeğin desteklediği donanımlar için gerekli olan sürücüler bulunmaktadır. Dünyada çok çeşitli donanım parçaları bulunmasından dolayı, donamın sürücüleri de çok büyük bir çeşitlilik göstermektedir. Donanımların, yazılımlar tarafından kontrol edilen parçaları sıklıkla benzerlik göstermektedir. Bu sayede aynı işlevleri destekleyen sürücüler arasında bir sınıflama yapmak mümkün olmaktadır. Bu sınıflandırmadaki her bir sürücü kendi aygıtını yönetmek için gerekli olan yazılımın haricindeki bölümlerde çekirdeğin aynı parçasındaki aynı yazılımı kullanır. Örneğin: Bütün disk sürücüleri temelde aynı işlevi yerine getirirler: sürücüyü başlatmak, N sektörünü okumak ve yazmak gibi.

Çekirdeğin içinde bulunan benzer özelliklere sahip bazı yazılım servisleri aynı sınıflar içinde özetlenmiştir. Örneğin çeşitli ağ protokolleri tek bir BSD soket kütüphanesi içinde özetlenmişlerdir. Başka bir örnek de sanal dosya sistemi (VFS) katmanıdır ve kendi gerçekleştirdiği dosya sistemi işlevlerini özetler. Her bir dosya sistemi türü, her bir dosya sisteminin gerçekleştirmesi gereken işlevleri sağlar. Bağımsız işlevler dosya sistemi ile ilişkiye girdiği zaman bilgiler VFS üzerinden uygun dosya sistemi sürücüsüne gider.

3.3. Bir UNIX sisteminde temel servisler

Bu bölüm fazla ayrıntıya girmeden bazı önemli Unix servislerini anlatmaktadır. Daha sonraki bölümlerde daha ayrıntılı açıklanmaktadır.

3.3.1. init

Bir Unix işletim sistemindeki en önemli servis tek başına **`init`** tarafından sağlanmaktadır. Açılış esnasında ilk başlayan ve önyükleme sonrasında son kapanan servis **`init`** servisidir. **`init`** açılışta; artalan süreçlerinin başlatılması, dosya sisteminin bağlanması ve kontrol edilmesi gibi bazı ufak tefek işlerin yapılmasını sağlar.

init'in yaptığı işlerin tam listesi **init**'in açılış kipinde yapılan seçime bağlıdır.

Tek kullanıcılı kip: root haricinde hiç kimse bağlanamaz ve root, konsolda bir kabuk kullanır.

Çok kullanıcılı kip:aynı anda pek çok kullanıcının bağlanabildiği kiptir. Linux sistemleri genelde bu kipte açılırlar.

Bazı sürümlerde bu iki kip genel konumlar olarak alındıktan sonra X kipi de eklenerek bunlar üçlenmiştir. Bunlara çalışma seviyeleri (runlevel) denir.

Linux 0–9 arasında 10 adet çalışma seviyesine izin verir. Ama çoğu dağıtımda bunları bazıları tanımlıdır. Diğerleri kullanılmaz.

- 0: sistemi kapatır.
- 1: tek kullanıcılı kip
- 2: çok kullanıcılı ancak NFS kapalı
- 3: çok kullanıcılı kip
- 5: çok kullanıcılı kip, açılışta X'e geçiş yapılır.
- 6: sistemi yeniden başlatır.

`/etc/inittab` dosyasının içeriği bu seviyeler hakkında bilgi içermektedir ve açılışta hangi kipte çalışmaya başlanılacağı yine bu dosya içinde tanımlanmıştır. Bu dosya sayesinde dağıtımınızda hangi çalışma seviyelerinin tanımlı olduğunu bulabilirsiniz.

Normal bir açılışta **init**, **getty** programının çalışmasını sağladıktan sonra sistemdeki öksüz programları evlat edinir. (Unix işletim sisteminde bütün programlar tek bir ağaç altında olmak zorundadır. Bu yüzden bağımsız programlar **init**'e bağlanır.)

Sistem kapatıldığı zaman **init**, süreçleri öldürür, dosya sistemlerini ayırır, işlemciyi durdurur ve yapılması istenen tanımlı diğer işlevleri yerine getirir. Bir Unix işletim sisteminde ilk gelen ve en son giden her zaman **init**'tir.

3.3.2. Uçbirimlerden bağlanmak

Seri hatlar üzerinden uçbirimden ve X window oturumu kapalı iken konsoldan bağlanmak için **getty** programı kullanılır. **init** her uçbirimden ayrı ayrı bağlanabilmek için **getty** programının bağımsız örneklerinin oluşturulmasını sağlar. **getty** kullanıcı isimlerini okur ve parolaları okumak üzere **login** programını çalıştırır. Şayet kullanıcı ismi ve parolası doğru ise **login** kullanıcının çalışması için bir kabuk açar. Her hangi bir sebepten dolayı kabuk kapatılırsa (kullanıcının ayrılması, vb...) veya kullanıcı ismi ile parola uyuşmaz ve **login** programı kapatılırsa, **init** bunu fark eder ve yeni bir **getty** programını devreye sokar. Çekirdeğin kullanıcıların sisteme bağlanması ile bir alakası yoktur. Bunlar tamamen sistem programları tarafından düzenlenir.

3.3.3. Syslog

Çekirdek ve bir çok sistem programları hata, uyarı ve buna benzer pek çok mesaj üretirler. Bu mesajların daha sonra (fazla geç olmadan) incelenmesi çok önemlidir. Bu nedenle bu mesajlar bir dosya içerisine yönlendirilmelidir. Bunu **syslog** programı yapar. Şayet istenirse mesajların içerik ve önemlerine göre ayrı ayrı dosyalama yapmak da, bu programın ayarları sayesinde mümkündür. Örneğin; çekirdek mesajları içeriklerinin önemli olmasından dolayı ayrı bir dosyaya yönlendirilmekte ve oraya yazılmaktadırlar. Oluşabilecek problemlerin çözümü için bu mesajlar düzenli olarak takip edilmelidir.

3.3.4. Süreli komut uygulamaları: cron ve at

Sistem yöneticileri ve kullanıcılar sık sık belli komutları düzenli olarak kullanmak zorundadırlar. Örneğin; sistem yöneticisi, geçici dosya ve dizinlerin tutulduğu `/tmp` ve `/var/tmp` dizinlerinin içerisini, disk üzerinde yer

açmak –eski dosyalardan kurtulmak– çalışması bittikten sonra kendini temizleyemeyen dosyalardan kurtulmak amacıyla silmek isteyebilirler.

cron servisi bu tür işleri yapar. Her kullanıcı, kendi istediği zamanda istediği komutların çalışmasını sağlayacak bir `crontab` dosyasına sahip olabilir ve içeriğini düzenleyebilir. **cron** bu tanımlanmış komutların çalıştırılmasından sorumludur.

at servisi de **cron** servisine benzer, ama aralarında temel bir farklılık vardır. **at** servisinde komutlar verildiği anda sadece bir kez çalıştırılır ve daha sonra tekrar edilmezler.

Daha ayrıntılı bilgi için cron(1), crontab(1), crontab(5), at(1) ve atd(8) man sayfalarına bakabilirsiniz.

3.3.5. Grafik kullanıcı arayüzü

Unix ve Linux işletim sistemleri kullanıcı arayüzlerini çekirdeğe dahil etmezler onun yerine, bu arayüzlerin kullanıcı seviyeli programlar tarafından yönlendirilmesine izin verirler.

Bu hem metin hem de grafik arayüzler için geçerlidir. Bu düzen; sisteme geniş bir esneklik sağlamasının yanı sıra, kullanıcı arayüzlerinin eklenmesinin kolaylığından kaynaklanan, sistemi öğrenme zorluğunu da yanında getirmektedir.

Linux ile kullanılan bu grafik ortam ilk önceleri X Pencere Sistemi olarak adlandırılmıştır. Kısaca X sistemi de denir. X bir kullanıcı arayüzü eklemez, sadece kullanıcı ara yüzüne geçiş sağlayan araçlardan oluşur. Bazı popüler pencere yöneticileri şunlardır: fvwm, icewm, blackbox ve windowmaker. KDE ve GNOME ise popüler masaüstü yöneticileri olarak bilinirler.

3.3.6. Ağ İşlemleri

Ağ, iki ya da daha fazla sayıda bilgisayarı birbirine bağlayarak, onlar arasında iletişim sağlanmasıdır. Bu bağlama ve iletişimin sağlanması biraz karmaşık bir olaydır, ama sonuçlar son derece faydalı ve kullanışlı olmaktadır.

Unix işletim sistemi pek çok ağ özelliğine sahiptir. Pek çok basit servis (yedekleme, yazıcıya gönderme, dosya sistemleri işlevleri ,vb..) ağ üzerinden yapılabilir. Bu özellikler sistem yönetimi kolaylaştırmaktadır. Hem merkezi hem de yayılmış bir yönetim olanağı sunmasından dolayı; ucuz maliyet ve hatalara karşı dayanıklılık imkanlarını sağlamaktadır.

Bu kılavuz, network konusunu kısaca anlatmakta, sadece nasıl çalıştığını açıklamaktadır. Daha ayrıntılı bilgi için Linux Ağ Yöneticisinin Kılavuzu (Linux Network Administrator's Guide) isimli kılavuza `http://www.tldp.org/LDP/nag2/index.html` adresinden ulaşabilirsiniz.

3.3.7. Ağdan sisteme bağlanma

Ağ bağlantıları, normal kullanıcı bağlantılarından biraz farklıdır. Üzerinden bağlanılması mümkün olan çeşitli fiziksel seri bağlantıları mevcuttur. Ağ üzerinden bağlanan her kişi için ayrı bir sanal ağ bağlantısı mevcuttur ve bu bağlantı sayısı bant genişliğinin elverdiği ölçüde çok çeşitli olabilir. Bu yüzden muhtemel her sanal bağlantıda bağımsız bir **getty** programını çalıştırmak mümkün değildir. Bir ağa bağlanmak için çok çeşitli yollar mevcuttur; TCP/IP ağlarında, **telnet** ve **rlogin** gibi temel programlar kullanılabilmektedir.

Son zamanlarda **ssh** güvenlik gerekçesi ile sistem yöneticileri tarafından daha çok tercih edilir olmaktadır. Ağ bağlantılarında her bağlantı için bağımsız bir artalan süreci mevcuttur (**telnet** ve **rlogin** ayrı birer artalan sürecine sahiptirler). Bir bağlantı istemi geldiğinde; bağımsız bir girişim için kendinin bir örneğini oluşturur ve bu sırada orjinal örnek diğer girişimleri dinlemeye devam eder. Oluşturulan yeni örnek **getty** gibi çalışır.

3.3.8. Ağ Dosya Sistemleri

Bir ağ sistemi üzerinden yapılabilecek en faydalı işlerden birisi Ağ Dosya Sistemi üzerinden yapılacak dosya transferleri ve dosya paylaşımıdır. Genellikle Sun firması tarafından geliştirilmiş olan ağ dosya sistemi *Network File System* (NFS) kullanılmaktadır. Bir ağ dosya sistemi ile bir makine üzerinde her hangi bir program tarafından yapılmış olan herhangi bir dosya işlemi ağ üzerinden başka bir makineye gönderilebilir. Bu çalışan programın, karşı makinedeki dosyaları, sanki kendi çalıştığı makinede bulunan dosyalar olarak görmesinden kaynaklanır ve bu sayede bu işlevler yerine getirilir. Bu sayede bilgi paylaşımı her hangi bir program değişikliğine gerek kalmadan basit bir şekilde sağlanmış olur. Diğer bir popüler dosya paylaşım sistemi ise `http://www.samba.org/` adresinde bulunabilecek olan Samba dosya sistemidir. Bu protokol MS Windows makineleri üzerinden dosya ve yazıcı paylaşımına izin verir (Ağ Komşuları üzerinden).

3.3.9. Eposta

Bilgisayarlar üzerinden haberleşmede en popüler yöntem elektronik postadır. Elektronik mektup özel programlar aracılığıyla dosyalar içine depolanır ve göndermek/almak/okumak için özel programlar kullanılır.

Bütün kullanıcılar yeni epostalarının depolandığı, özel bir biçeme sahip olan gelen postalar kutusu olarak adlandırılabilen bir dosyaya sahiptirler. Birisi size eposta gönderdiği zaman, eposta programı bunu sizin posta kutunuza yerleştirir ve gelen mektubu gelen postaların tutulduğu dosyaya ekler. Şayet posta kutunuz başka bir makine üzerinde ise posta bu makineye gönderilir ve burada saklanır.

Posta sistemleri pek çok programı ihtiva ederler. Program kullanıcıları pek çok çeşitteki programları kullanırken (Eposta istemcisi – Mail User Agent – MUA; örneğin: pine, mutt, elm) postaların yerel ya da uzaktaki makinelerin posta kutularına dağıtımını bir program (Eposta Aktarım Aracısı – Mail Transfer Agent – MTA; örneğin: sendmail veya qmail) yapar. Posta kutuları genellikle `/var/spool/mail`'de depolanır.

3.3.10. Yazdırma

Aynı anda bir yazıcıyı sadece bir tek kullanıcı kullanabilir. Yazıcıların paylaşılması oldukça ekonomik bir çözümdür. Bütün yazım işleri bir yazılım olan bir yazdırma kuyruğu tarafından yönetilir. Burada bütün işler sıraya konur ve sırası gelen iş kuyruktan alınıp yazılmak üzere otomatik olarak yazıcıya gönderilir. Bu kullanıcının, bütün bu işleri organize etmek için büyük bir çaba ve zaman harcamasını önler.

Bu yazılım çıktı işlemlerini disk üzerinde ayrı bir dosya haline getirir ve bu sayede bir uygulama programı işleri parçalayarak yazıcıya gönderme şansına sahip olur. Böylece program, çıktı işlemi bitene kadar beklemek zorunda kalmaz. Bu son derece kullanışlı bir olaydır. Çünkü bu sayede birisi bir basım işini bitirmek için, diğerinin bütün işlerinin bitmesini beklemek zorunda kalmaz. Kendi işinin kuyruk sırası gelince, işleri otomatik olarak işleme konur ve yazıcıdan çıktı alınır.

3.3.11. Dosyasisteminin Yerleşim Düzeni

Bir dosya sistemi kök dosya sisteminin kolları boyunca çeşitli bölümlere ayrılır: `/bin`, `/lib`, `/etc`, `/dev` ve birkaç benzeri; program ve değişmeyen verilerden oluşan `/usr` dosya sistemi; değişen verilerden oluşan `/var` dosya sistemi; ve herkesin kişisel dosyalarının bulunduğu `/home` dosya sistemi. Donanım yapılandırmasına ve sistem yöneticisinin kararlarına göre bu dosya sistemleri farklı olabilir. Hepsi tek bir dosya sistemi halinde de olabilir.

Dizin Yapısına Genel Bakış (sayfa: 13) bölümünde dosyasisteminin yerleşim düzeni daha ayrıntılı bir biçimde anlatılmıştır. En ayrıntılı bilgi Dosyasisteminin Hiyerarşik Standardı – Filesystem Hierarchy Standard – (FHS)[B10] isimli belgede bulunabilir.

4. Dizin Yapısına Genel Bakış

> "İki gün sonra, Pooh daldan ayaklarını sarkıtmış bir biçimde orada duruyordu, yanında dört çanak bal vardı." (A.A. Milne)

Bu bölüm; dosya sistemi hiyerarşi standardını göz önüne alarak, Linux standart dizin yapısının en önemli parçalarını anlatmaktadır. Çeşitli amaçlar ile dizin yapısını farklı dosya sistemlerine bölebilirsiniz. Bu Linux'ün esnek yapısından kaynaklanmaktadır. Aynı disk üzerinde çeşitli dosya sistemleri olabilir. Bütün Linux dağıtımlarında bu mümkün olmayabilir. Ama burada genel bir bilgi verilmeye çalışılmaktadır.

4.1. Arkaplan

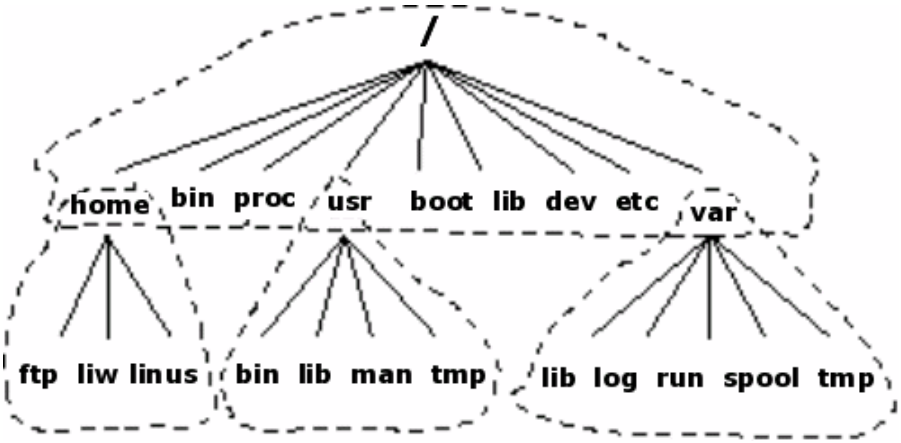
Bu bölüm Dosyasistemi Hiyerarşisi Standardının (Filesystem Hierarchy Standart – FHS[B11]) 2.1 sürümü üzerinden, Linux dizin yapısının nasıl organize edildiğini anlatmaktadır. Standardizasyon, Linux işletim sistemleri için yazılım hazırlanmasını ve uyarlanabilmesini daha kolaylaştırır, her şey belli bir standart içinde olduğundan sistem yönetiminde pek çok kolaylıklar sağlar. Bu standardizasyon için her hangi bir otorite tarafından yapılan bir baskı söz konusu değildir, ama pek çok Linux dağıtımı bu standardizasyonu desteklemektedir. Çok önemli ve geçerli sebepler olmadıkça FHS'yi bırakmak hiç de akıllıca bir fikir değildir. FHS, Unix gelenek ve göreneklerini izleyerek diğer Unix sürümleri ile Linux arasında bir uyum ve köprü vazifesi görmektedir.

Bu bölümde FHS ayrıntılı bir biçimde anlatılmamıştır. Sistem yöneticilerinin FHS'i anlayabilmeleri için FHS ile ilgili kılavuzu tamamen okuması daha yararlı olur.

Bu bölümde bütün dosyalar ayrıntılı bir biçimde anlatılmamış, sadece genel bir bakış açısı kazandırmak amacıyla dosya sistemleri konusu temelinde genel bir anlatım yapılmıştır. Bununla beraber bu kılavuz içinde veya diğer kılavuz sayfalarında burada anlatılan dosyalar hakkında daha geniş bilgi bulmak mümkündür.

Dizin ağacının tamamı çeşitli görevleri yerine getirmek, sistem yedeklerini daha kolay almak veya disk üst limitleri nedeniyle; kendi diski ya da bağımsız diskler üzerinde çeşitli küçük parçalara ayrılmış olabilir. Ana parçalar: kök (`/`), `/usr`, `/var`, ve `/home` dosya sistemleridir (Bakınız: *Bir Unix dizin ağacının yapısı* (sayfa: 14)). Her bölüm farklı bir amaca hizmet etmektedir. Dizin yapısı bir Linux makinesinin ağ üzerinde daha hızlı ve daha verimli çalışmasını sağlamak amacıyla yapılır.

Şekil 2. Bir Unix dizin ağacının yapısı



Kesikli çizgiler disk bölümlerinin sınırlarını belirtir.

Dizin ağacının farklı parçalarının rolleri aşağıda açıklanmıştır.

- Kök dosya sisteminin kendine özgü bir yapısı vardır. Genellikle yerel sabit disk üzerinde olabileceği gibi ram diskte veya ağ tarafından da yönetilebilir. Kök dosya sistemi, sistemin açılıp, diğer dosya sistemlerinin bu sisteme bağlanabilir bir hale getirilmesi için gereklidir. Kök dosya sistemi tek kullanıcılı bir makine

için yeterli olacaktır. Ayrıca bozulan bir sistemi tamir etmek için ve kaybolan yedeklenmiş dosyaların geri getirilmesini sağlayan araçlara da sahiptir.

- `/usr` dosya sistemi, bütün kütüphaneleri, komutları, normal işlemler sırasında değişmeyen dosyaları ve kılavuz sayfalarını içerir. `/usr` dosya sistemindeki hiçbir dosya her hangi bir makine için özelleştirilmez. Hepsi normal kullanım amacına yönelik hazırlanmış dosyalardır. Bu sayede buradaki dosyalar ağ üzerinden paylaşılabilir. Bu sayede sabit disklerden önemli ölçüde yer kazanmak mümkün olur. Bu da çok büyük bir tasarruf sağlar. Bir uygulamanın güncellenmesinde sadece ana `/usr` değiştirilir. Bu da `/usr`'yi ağ sistemine bağlamayı oldukça kolaylaştırır. Dosya sistemi yerel bir disk üzerinde olsa bile sadece okunabilir şekilde ayarlanması daha uygun olur. Bu sayede herhangi bir çökme durumunda dosyaların hasar görme ihtimalini azaltmış oluruz.

- `/var` dosya sistemi sürekli değişen dosyaları barındırır. Epostalar, mesajlar, spool dizinleri (news, printer, vb..), düzenlenmiş kılavuz sayfaları ve temp dosyaları bu dosya sisteminin içindedir. Geleneksel olarak `/var` dizininin altındaki her şey `/usr` dizini altında bir yerlerde mutlaka vardır. Bu da `/usr` dosya sisteminin salt okunur yapılmasına olanak verir.

- `/home` dosya sistemi kullanıcıların ev dizinlerini yani sistemde bulunan bütün gerçek verileri ihtiva eder. Ev dizinlerini çeşitli parçalara ayırmak yedekleme açısından son derece faydalı olabilir. Nede olsa bütün dizinler aynı hızla değişmiyor. Büyük `/home` dizinlerinin; `/home/staff`, `/home/student` gibi parçalara ayrılması daha akıllıca olacaktır.

Bununla beraber yukarıda dosya sistemi olarak adlandırılan bölümlerin her birinin ayrı birer dosya sistemi olması gerekmemektedir. Bütün bunlar tek bir dosya sistemi altında kolaylıkla tutulabilir. Özellikle tek kullanıcılı basit sistemlerde. Ayrıca sabit diskin yapısına ve büyüklüğüne göre çeşitli dosya sistemleri kullanılabilir.

Unix dosya sistemi yapısı bütün dosyaları amaçlarına göre gruplara ayırır. Bütün komutlar bir yerde, bütün veriler başka bir yerde, bütün belgeler üçüncü bir yerde gibi. Diğer bir alternatif ise dosyaları ait oldukları programlara göre sınıflamaktır: bütün emacs dosyaları bir yerde, teX dosyaları başka bir yerde gibi. Daha sonra paylaşılan dosyaların bulunmasında bir problem ortaya çıkmaktadır. Kılavuz sayfalarının hepsini bulan bir program yapmak ve dosyaları paylaştırmak tam bir kabusa dönmektedir.

4.2. Kök Dosya Sistemi

Kök dosya sistemi çok önemli ve nadiren değişen bilgileri içerdiği için, bozulma riskine karşın küçük tutulmalıdır. Bozulmuş bir kök dosya sistemi bütün sistemin çökmesi anlamına gelir (disket ile sistemi açmak haricinde) ki bu da oldukça büyük bir risktir.

Kök dizini genellikle `boot.image(vmlinuz)` dosyası haricinde başka bir dosya içermez ve sadece dizinleri ihtiva eder:

`/bin`
: Açılış sonrası normal kullanıcılar tarafından kullanılacak komutları içerir.

`/sbin`
: `/bin` dizinine benzer ama buradaki komutlar, normal kullanıcıların kullanabilmesi için ayrıca tanımlanmadıkça sadece root kullanıcı içindir.

`/etc`
: Makineye özel yapılandırma dosyaları bulunur.

`/root`
: root kullanıcısı için ev dizini. Diğer kullanıcılar genellikle buraya erişemezler.

`/lib`

Kök dosya sistemindeki programlar tarafından ihtiyaç duyulan paylaşımlı kütüphaneler.

`/lib/modules`
: Genellikle sistemin açılışı sırasında sorunların önlenilmesi için ihtiyaç duyulan yüklenebilir çekirdek modüllerini içerir.

`/dev`
: Aygıt dosyaları bulunur. Bazı çok kullanılan aygıt dosyaları *Aygıt Dosyaları* (sayfa: 21) bölümünde incelenmiştir.

`/tmp`
: Geçici dosyalar bulunur. Açılış sonrası çalışan programlar genelde daha fazla yer olmasından dolayı `/var/tmp` dizinini kullanırlar. Sıklıkla `/tmp` dizininden `/var/tmp` dizinine sembolik bir bağ verilmiştir.

`/boot`
: LILO ve benzeri programların dosyaları burada bulunur. Genellikle önyükleme kaydının dosyası kök dizin yerine burada saklanır. Eğer birden çok çekirdek imajı sistemde mevcut ise bu dizin çok fazla büyür. Bu nedenle, böyle durumlarda ayrı bir dosya sistemi açmak daha iyi olur. Veya `/boot` dizini IDE disk üzerinde ilk 1024 silindir içinde olduğundan emin olmak için ayrı bir dosya sistemi açılabilir. [2]

`/mnt`
: Sistem yöneticileri tarafından geçici dosya sistemi bağlama işlemlerinin yapıldığı yerdir. Dosya sistemi bağlama işlemleri bu dizine yönlendirilir. Bazı durumlarda bu dizin de alt dizinlere ayrılabilir. Msdos ve ext2 dosya sistemlerine ait aygıtları ayrı ayrı bağlamak için ayrım yapılabilir. Bazı dağıtımlarda `/mnt/cdrom` ve `/mnt/floppy` dizinleri ön tanımlı olarak gelir.

`/proc`, `/usr`, `/var`, `/home`
: Diğer dosya sistemleri için bağlama noktalarıdır. [3]

4.3. `/etc` dizini

`/etc` dizini pek çok dosyayı ihtiva eder. Bazıları aşağıda tanımlanmıştır. Diğerlerinin hangi programlara ait olduğuna siz karar vermeli ve onlara ait kılavuz sayfalarını okumalısınız. Ağ ile ilgili pek çok yapılandırma dosyası bu dizin altında bulunmaktadır. Bunlar Ağ Yöneticilerinin Kılavuzunda (Network Administrator's Guide[B15]) ayrıntılı bir biçimde anlatılmaktadır.

`/etc/rc` veya `/etc/rc.d` veya `/etc/rc?.d`
: Açılışta veya çalışma seviyesi değiştiği zaman çalışacak betikler ve betik dizinleri burada bulunur. *init* (sayfa: 56) bölümünde daha ayrıntılı bilgi bulabilirsiniz.

`/etc/passwd`
: Her kullanıcı hakkında ayrıntılı bilgilerin bulunduğu, kullanıcı veritabanı olarak da düşünebileceğimiz yerdir. Burada kullanıcıların gerçek ismi, kullanıcı isimleri, şifrelenmiş parolaları, ev dizinleri gibi bilgiler yer almaktadır. Dosya biçemi hakkında ayrıntılı bilgi almak için **`passwd`** kılavuz sayfasına bakınız. Bu günlerde şifrelenmiş parolalar genel olarak `/etc/shadow` dizininde saklanmaktadır. Bu, kullanıcı hakkında parola hariç bütün bilgilerin `passwd` dosyası içerisinde saklanmakta olduğu anlamına gelir.

`/etc/fdprm`
: Floppy disk parametreleri burada saklanır. Hangi tür disket biçemlerinin neye benzediği burada tanımlanmıştır. **`setfdprm`** tarafından kullanılır. **`setfdprm`** hakkında kılavuz sayfalarında ayrıntılı bilgi bulunmaktadır.

`/etc/fstab`

Açılış esnasında **mount -a** (/etc/rc ya da buna benzer bir başlangıç dosyası aracılığı ile) komutu ile otomatik olarak bağlanılan dosya sistemleri hakkında bilgi verir. Ayrıca Linux'de **swapon -a** komutu ile otomatik olarak kullanılan takas alanları hakkında bilgi de içerir. Daha fazla bilgi için *Dosya sistemlerinin bağlanması ve ayrılması* (sayfa: 39) bölümüne ve **mount** kılavuz sayfasına bakınız. Ayrıca fstab dosyası için kılavuz sayfalarının 5. bölümünde kendine ait bir kılavuz sayfası bulunmaktadır.

/etc/group
: /etc/passwd dosyasına benzer ama kullanıcılar yerine grupları tanımlar. Ayrıntılı bilgi kılavuz sayfalarının beşinci bölümünde bulunabilir.

/etc/inittab
: **init** için yapılandırma dosyasıdır.

/etc/issue
: Login isteminden önceki **getty** çıktıları burada tutulur. Genellikle sistem yöneticileri tarafından hazırlanan açılış ve hoş geldin mesajları yer alır. Sistem yöneticisinin isteğine göre düzenlenir.

/etc/magic
: **file** komutu için yapılandırma dosyasıdır. Hangi tür dosya biçeminin neye göre saptandığı burada yer alır. **file** ve magic kılavuz sayfalarında ayrıntılı bilgi mevcuttur.

/etc/motd
: Her başarılı girişten sonra görüntülenen günün mesajı burada bulunur. Kullanıcıları günlük işler hakkında uyarmak amacıyla kullanılır.

/etc/mtab
: Bağlı dosya sistemlerinin tam listesini verir. Açılış betikleri tarafından kurulur ve **mount** komutu ile otomatik olarak güncellenir. Bağlı dosya sistemlerinin listesine ihtiyaç olduğu zaman kullanılır. Örneğin; **df** komutu gibi.

/etc/shadow
: Shadow parola yazılımları kurulu sistemlerdeki shadow parola dosyaları burada yer alır. Shadow parolaları /etc/passwd'deki şifrelenmiş parolaları /etc/shadow'a taşır. /etc/shadow içeriğini root kullanıcıdan başkası okuyamaz. Bu parolaların kırılmasını zorlaştırır. Şayet elinizdeki Linux dağıtımı size bu seçeneği sunuyorsa, ki pek çok dağıtım bu seçeneği sunar, bu seçeneğin etkin olması şiddetle tavsiye edilir.

/etc/login.defs
: **login** komutu için yapılandırma dosyasıdır. login.defs dosyasının kılavuz sayfasını, kılavuz sayfalarının 5. bölümünde bulabilirsiniz.

/etc/printcap
: /etc/termcap'a benzer ama yazıcı ayarları içindir. Daha farklı bir sözdizimi kullanır. /etc/termcap dosyasının kılavuz sayfasını, kılavuz sayfalarının 5. bölümünde bulabilirsiniz.

/etc/profile, /etc/csh.login, /etc/csh.cshrc
: Sisteme giriş ve başlatma sırasında Bourne veya C kabukları tarafından yürütülen dosyalardır. Bu sistem yöneticisine, bütün kullanıcılar için aynı kuralları koyma şansı verir.

/etc/securetty
: Güvenli uçbirimleri ayarlar. Örneğin root kullanıcının hangi uçbirimden bağlanacağı gibi. Genel olarak sadece sanal uçbirimler listelenir. Böylece ağ veya bir modem üzerinden yapılan izinsiz girişlerde kötü niyetli kişilerin süper kullanıcı haklarını elde etmesi imkansızlaşır (en kötü ihtimal ile çok zorlaşmış olur). Bir ağ üzerinden root kullanıcı bağlantısına izin vermeyin. Normal bir kullanıcı gibi bağlanın ve **su** veya **sudo** komutlarını root haklarını kazanmak için kullanın.

`/etc/shells`
: Güvenli kabukların listesidir. **chsh** komutu kullanıcılara bu listede bulunan kabuklar arasından istediklerini seçme hakkı verir. Kullanıcı giriş sonrası seçtiği kabuğa düşer. **ftpd**, bir makine için FTP servisi veren sunucu işlemleri, bağlanan kullanıcının kabuğunun `/etc/shells` içerisinde listelenmiş olup olmadığını kontrol eder ve listeli değil ise bağlantıya izin vermez.

`/etc/termcap`
: Uçbirim yetenekleri veritabanıdır. Ne çeşit önceleme dizgelerinin hangi uçbirimlerce kullanılacağını tanımlar. Dışarıya doğru yapılan çıkışların sadece belirli cins uçbirimler üzerinden yapılmasını sağlar ve bunu kontrol eder. Pek çok program pek çok çeşit uçbirim ile çalışabilir. Daha fazla bilgi için `termcap`, curs_termcap ve `terminfo` kılavuz sayfalarına bakınız.

4.4. /dev dizini

`/dev` dizini pek çok çeşit aygıt için aygıt giriş/çıkış dosyaları içerir. Ethernet kartı hariç tüm aygıtlar için bu dizin altında – sistemde olsun ya da olmasın – bir düğüm vardır. Özel bir düzendeki aygıt dosyaları *Aygıt Dosyaları* (sayfa: 21) bölümünde açıklanmıştır. Sistem kurulumu sırasında aygıt dosyaları yaratılır. Daha sonra ise **/dev/MAKEDEV** betiği ile yenileri yapılabilir. Sadece yerel aygıt dosyalarını veya bağlarını tanımlamak için sistem yöneticileri tarafından yazılmış olan **/dev/MAKEDEV.local** betiği kullanılır. Bu işlem standart **MAKEDEV** betiği ile oluşturulamayan aygıt dosyaları ve bağları için yapılır.

4.5. /usr dosya sistemi

Bütün programlar buraya yüklendiği için bu dizin oldukça geniştir. `/usr` içindeki bütün dosyalar bir Linux dağıtımı ile gelir. Yerel kullanıcılar tarafından yüklenen diğer programlar `/usr/local` altına gider. Bu sayede aynı dağıtımdan güncelleme veya başka bir dağıtımdan kurulum yapmak mümkün olur. Bu olaylar sırasında yerel dosyalarda bir kayıp söz konusu olmaz. Ve bütün dosyaları yeniden yüklemek gerekmez. `/usr` dosya sisteminin bazı alt dizinleri aşağıda belirtilmiştir.

`/usr/X11R6`
: X Pencere Sistemi dosyalarını içerir. X'in yüklenmesi ve geliştirilmesi için X dosyaları sisteme entegre değildir. `/usr/X11R6` dizinin içeriği `/usr` dizininkine benzer.

`/usr/bin`
: Hemen hemen bütün kullanıcı komutları burada yer alır. Bazı komutlar `/bin` veya `/usr/local/bin` altındadır.

`/usr/sbin`
: Kök dosya sisteminde ihtiyaç duyulmayan sistem yöneticisi komutları burada bulunur. Örneğin pek çok sunucu programı burada bulunur.

`/usr/share/man`, `/usr/share/info`, `/usr/share/doc`
: Kılavuz sayfaları, GNU bilgi sayfaları, SSS ve çeşitli diğer belgeler.

`/usr/include`
: C programlama dili başlık dosyaları bulunur. Geleneksel olarak `/usr/lib` altında olması gerekirken, ezici bir çoğunluk bu ismi desteklemektedir.

`/usr/lib`
: Programlar ve alt sistemler için değişmeyen veri dosyaları burada bulunur. `lib` kısaltması library (kütüphane) den gelir. Genel program kütüphaneleri buradadır.

`/usr/local`

Yerel kullanıcılar tarafından yüklenmiş programlar burada bulunur. Dağıtımlar buraya hiçbir şey yüklemez. Burası tamamen yerel yöneticinin kullanımına tahsis edilmiştir. Herhangi bir güncelleme veya yükseltme durumunda buradaki programlar üzerine kesinlikle hiçbir şey yazılmayacaktır.

4.6. /var dosya sistemi

Sürekli değişen sistem bilgileri burada saklanır. Her sistem için özeldir. İstisnalar dışında diğer sistemler ile paylaşılmaz.

`/var/cache/man`
: İsteğe göre düzenlenmiş kılavuz sayfalar için önbellekleme alanı. Bazı kılavuz sayfaları ön–biçemli gelebilir. Bunlar `/usr/share/man/cat*` dizinlerinde bulunabilir, biçemlendikten sonra `/var/cache/man` altında saklanır. Genellikle kılavuz sayfaları `/usr/share/man/man?/` (?= kılavuz sayfalarının bölüm numarası – bilgi için man7 dizininde bulunan **man** kılavuz sayfasına bakınız) altında depolanır.

`/var/games`
: `/usr` altındaki oyunlara ait her türlü bilgi burada saklanır. Böylece `/usr` ile salt okunur bağlantı sağlanır.

`/var/lib`
: Sistemin normal çalışması esnasında değişen dosyalar ve bilgiler burada tutulur.

`/var/local`
: `/usr/local` altında kurulmuş çeşitli programlara ait bilgiler burada bulunur (Örneğin; sistem yöneticileri tarafından kurulmuş programlar). Şayet ayarlanmış veya yönlendirilmiş iseler yerel olarak kurulmuş bile olsalar bu programlar, `/var` dizini altındaki diğer bölgeleri de kullanırlar.

`/var/lock`
: Kilit (lock) dosyaları. Pek çok program belirli aygıt ya da dosyaları kullandıklarını göstermek için `/var/lock` dizini altında bir kilit dosyası yaratır. Diğer programlar buraya dikkat ederler ve böylece aygıt ya da dosya çakışması olması önlenir.

`/var/log`
: Sistem yöneticisini ilgilendiren günlük kayıtları burada tutulur. Syslog (tüm çekirdek ve sistem programlarının mesajları `/var/log/messages` dosyasında saklanır) ve **login** (sisteme kullanıcı giriş ve çıkışları `/var/log/wtmp` dosyasında saklanır) programlarının kayıtları burada tutulur.

`/var/mail`
: FHS tarafından onaylanan, kullanıcı eposta kutularının bulunduğu dosyalardır. Bazı dağıtımlarda `/var/spool/mail` içinde tutulurlar.

`/var/run`
: Bir sonraki açılışa kadar geçerli olan sistem hakkında bilgi içeren dosyalar burada tutulur. Örneğin `/var/run/utmp` sisteme bağlı durumdaki kişiler hakkında bilgi içerir.

`/var/spool`
: Haberler, yazıcı kuyrukları ve diğer kuyruktaki işler hakkında bilgi içeren dizindir. Her biri için ayrı bir alt dizin mevcuttur. `/var/spool/news` gibi.. Bazı dağıtımlarda eposta kutularını içeren dosyalarda `/var/spool/mail` dizini altındadır.

`/var/tmp`
: Çok büyük veya çok uzun bir zamandır var olan `/tmp` dizini içerisindeki geçici dosyaları içerir. Sistem yöneticisi `/var/tmp` içerisinde çok eski dosyalara müsaade etmeyebilir.

4.7. /proc dosya sistemi

Sanal dosya sistemidir. Disk üzerinde yer kaplamaz. Çekirdek /proc dizinini bellekte yaratır. Sistem hakkında bilgiler burada saklanır (süreçler ve isimleri ,vb..). /proc dosya sistemi ayrıntılı olarak kılavuz sayfasında açıklanmıştır. Bazı çok önemli dosya ve dizinler:

/proc/1
: 1 numaralı süreç hakkında bilgi içerir. Her sürecin kendi numarası ile anılan bir dizini vardır.

/proc/cpuinfo
: İşlemci hakkında türü, üreticisi, modeli gibi bilgileri içerir.

/proc/devices
: Çekirdekte o an için çalışan aygıt sürücülerinin listesini verir.

/proc/dma
: O anda kullanılan DMA kanallarını gösterir.

/proc/filesystems
: Çekirdekte tanımlanmış olan dosya sistemlerini gösterir.

/proc/interrupts
: Hangi kesmelerin kullanıldığını ve hangilerinin halihazırda beklediğini gösterir.

/proc/ioports
: Hangi giriş/çıkış portlarının kullanıldığını gösterir.

/proc/kcore
: Sistemdeki fiziksel belleğin bir görüntüsü. Fiziksel bellek ile aynı boyuttadır. Çalışan programlar için yaratılmıştır ama gerçek bir bellek değildir. Siz herhangi bir yere kopyalamadıkça /proc altındaki bilgiler diske yazılmaz.

/proc/kmsg
: Çekirdek mesajları. Ayrıca **syslog**'a yollanır.

/proc/ksyms
: Çekirdek sembol tablosu.

/proc/loadavg
: Sistemin ortalama yükü. Üç adet anlamsız gösterge sistemin o an ne kadar çalıştığını gösterir.

/proc/meminfo
: Hem fiziksel bellek hem de takas hakkında bilgi verir.

/proc/modules
: O an yüklü çekirdek modüllerini gösterir.

/proc/net
: Ağ protokolleri hakkında durum bilgisi verir.

/proc/self
: O an /proc'a göz atan programın süreç dizinine sembolik bağ. Eğer iki süreç söz konusu ise ikisi de ayrı bağ alır. Böylece programların süreç dizinlerine ulaşmaları daha uygun hale getirilmiş olur.

/proc/stat
: Sistem hakkında çeşitli istatistikler bulunur. Örneğin sistem açılışından beri meydana gelen hataların sayfa sayısı gibi.

`/proc/uptime`
: Sistemin açık kaldığı süreyi verir.

`/proc/version`
: Çekirdek sürüm bilgilerini içerir.

Yukarıda adı geçen dosyaların pek çoğu kolayca okunabilecek şekilde metin dosyaları halindedir. Fakat bazılarının düzenlenmesi gerekebilir. Bunları okumaktansa içeriklerini gösteren komutları kullanmak daha kolaydır. Örneğin **free** programı `/proc/meminfo`'yu okur ve bayt olan değerleri kB'a çevirerek gösterir (ve biraz daha fazla bilgi verir).

5. Aygıt Dosyaları

Bu bölümde aygıt dosyalarının ne olduğu ve nasıl yaratılacağı temel bilgiler eşliğinde anlatılmaktadır. Ayrıca daha genel aygıt dosyaları listelenmiştir. Şayet Linux çekirdeğinin kaynak kodlarına sahipseniz genel olarak aygıt listelerini `/usr/src/linux/Documentation/devices.txt` içerisinde bulabilirsiniz.

5.1. MAKEDEV Betiği

Aygıt dosyalarının pek çoğu sistem yüklemesi sırasında kurulmuş ve kullanıma hazır bir biçimde beklemektedir. Şayet sisteminizde olmayan bir aygıt dosyası yaratmak zorunda kalırsanız **MAKEDEV** betiğini kullanmalısınız. Bu betik `/dev/MAKEDEV` olarak bulunabir. Fakat `/sbin/MAKEDEV`'e de sembolik bir bağ verilmiş olabilir. Şayet PATH içerisinde tanımlanmamışsa, bu betiği kullanabilmek için yolunu PATH içeriğinde tanımlamalısınız. Betiğin genel kullanımı şu şekildedir:

```
# /dev/MAKEDEV -v ttyS0
create ttyS0   c 4 64 root:dialout 0660
```

Burada 4. anadüğüm, 64. altdüğümde sahibi root ve grubu dialout için erişim izni 660 olan `/dev/ttyS0` karakter aygıtı yaratılmış oldu.

`ttyS0` bir seri porttur. Anadüğüm ve altdüğüm numaraları, çekirdek tarafından algılanacak olan numaralardır. Çekirdek donanım aygıtlarına numaralar aracılığı ile ulaşır. Bizim için bunu yapmak çok zor olduğundan biz dosya isimlerini kullanırız. Erişim izinleri kullanıcı için (root) okuma ve yazma; grubu (dialout) için okuma ve yazma şeklindedir. Başka hiçbir kimse bu aygıta erişemez.

5.2. mknod komutu

MAKEDEV halihazırda var olmayan aygıt dosyalarını yaratmak için kullanılır. Bununla beraber bazı zamanlar **MAKEDEV** yaratmak istediğiniz aygıta ilişkin dosyayı tanımayabilir. Burada **mknod** komutu devreye girer. Bu komutu kullanabilmek için anadüğüm ve altdüğüm numaralarını bilmek zorundasınız. Çekirdek içerisindeki `devices.txt` dosyası, bu bilgilerin yer aldığı temel kaynaktır.

Diyelim ki bizim elimizdeki **MAKEDEV** istediğimiz `/dev/ttyS0` aygıt dosyasını nasıl yapacağını bilmiyor. Bu durumda **mknod** komutunu kullanmak zorundayız. `devices.txt` dosyasından bakarak 4. anadüğüm ve 64. altdüğümde bir karakter aygıt yapmamız gerektiğini öğreniriz.

```
# mknod /dev/ttyS0 c 4 64
# chown root.dialout /dev/ttyS0
# chmod 0644 /dev/ttyS0
# ls -l /dev/ttyS0
crw-rw----   1 root dialout    4,   64 Oct 23 18:23 /dev/ttyS0
```

Gördüğünüz gibi bir dosyayı çok fazla adımda yaratmak durumundayız. Bu örnekte gereken süreçleri görebilmekteyiz. Burada `/dev/ttyS0` aygıt dosyasının **MAKEDEV** komutu tarafından yaratılamamış olması uç bir örnektir.

5.3. Aygıt Listesi

Aşağıdaki aygıt listesinde çok fazla ayrıntıya girilmemiştir. Bu aygıt dosyalarının pek çoğu çekirdekte derlenmiş halde bulunmakta ve donanımları doğrudan desteklemektedir. Daha fazla ayrıntı için çekirdek belgelerini okuyabilirsiniz.

Şayet burada bulunması gerektiğini düşündüğünüz ama burada olmayan aygıt dosyaları var ise lütfen bana bildirin. Onları da diğer sürümde bu listeye ekleyeceğim.

`/dev/dsp`
: Digital Signal Processor – Sayısal Sinyal İşleyici. Ses kartı ile ses üreten yazılım arasındaki arayüz olarak tanımlanabilir. 14. anadüğüm, 3. altdüğümde bulunan bir karakter aygıtıdır.

`/dev/fd0`
: Birinci disket sürücüsü. Şayet birden fazla sürücüye sahip olacak kadar şanslı iseniz diğer sürücüler ardışık bir biçimde numaralanacaktır (fd1, fd2, ...). `/dev/fd0` 2. anadüğüm, 0. altdüğümdeki bir karakter aygıtıdır.

`/dev/fb0`
: Birinci karetamponu (framebuffer) aygıtı. karetamponu grafik donanımı ile yazılımı arasındaki soyut bir katmandır. Uygulamalar, ne tür grafik donanımı kullandığınızı bilmek zorunda kalmadan, sadece açıkça tanımlanmış ve standart haline getirilmiş olan karetamponu sürücüsünün uygulama programlama arayüzü (API – Application Programming Interface) ile nasıl haberleşeceklerini bilirler. Birinci karetamponu 29. anadüğüm, 0. altdüğümde bir karakter aygıtıdır.

`/dev/hda`
: `/dev/hda` birincil IDE denetleyicisine bağlı ana IDE sürücüdür. Birinci IDE kablosuna bağlı `master` olarak ayarlanmış sabit disk olarak da tanımlanabilir. `/dev/hdb` birincil IDE denetleyicisine bağlı yardımcı sürücüdür. `/dev/hdc` `/dev/hdd` ise ikincil IDE denetleyicisine bağlı ana ve yardımcı IDE sürücüleridir. Her disk bölümlere ayrılmıştır. 1–4 arası bölümler birincil bölümler, 5 ve üzeri bölümler ise ek bölüm içindeki mantıksal bölümlerdir. Bu yüzden her bölüm için yapılmış aygıt dosyaları çeşitli bölümlerden meydana gelmişlerdir. Örneğin; `/dev/hdc9` ikincil IDE denetleyicisine takılmış bir ana IDE sabit diskinin ek bölümü içerisindeki bir mantıksal bölümü işaret eder. Ana ve alt düğüm numaraları biraz karmaşıktır. Birincil IDE denetleyicisi için bütün bölümler 3. anadüğümdeki blok aygıtlarıdır. Ana sürücü `hda` için altdüğüm 0, yardımcı sürücü `hdb` için altdüğüm 64'tür. Sürücü içindeki her bölümün, bölüm numarası bu altdüğüm numarasına eklenir. Örneğin; `/dev/hdb5` için anadüğüm 3, altdüğüm ise 69 (64+5) dur. İkincil IDE sürücüye bağlı sürücünün hesaplamaları da aynı yöntemle yapılır. Tek fark anadüğüm numarası 22 dir.

`/dev/ht0`
: Birinci IDE teyp sürücüdür. Sonraki sürücüler `ht1`, `ht2` şeklinde sıralanır. 37. anadüğümdeki karakter aygıtlarıdır ve `ht0` için altdüğüm 0, `ht1` için 1, `ht2` için 2 olarak sıralanır.

`/dev/js0`
: Birinci analog oyunçubuğu. Sonraki oyunçubukları `js1`, `js2`, `js1`, ... şeklinde sıralanır. 15. anadüğümdeki karakter aygıtlarıdır. Analog oyunçubukları 0. altdüğümden 127. altdüğüme kadar sıralanır. Sayısal oyunçubukları ise 128. altdüğümden başlar.

`/dev/lp0`

Birinci paralel yazıcı aygıtı. Diğer yazıcılar `lp1`, `lp2`, `lp3`, ... şeklinde sıralanır. 6. anadüğümdeki karakter aygıtlardır. Altdüğümler 0 dan başlar ve her aygıt için ardışık şekilde devam eder.

`/dev/loop0`
Birinci geridönüş (loopback) aygıttır. Geridönüş aygıtları diskler üzerinde bulunmayan blok aygıtlarındaki dosya sistemlerini bağlamak için kullanılırlar. Şayet bir iso9660 CD ROM'u bir CD'yi okumadan bağlamak istiyorsak geridönüş aygıtlarını kullanmalıyız. Bu genellikle kullanıcıya açıktır ve **`mount`** komutu tarafından kullanılır. **`mount`** ve **`losetup`** kılavuz sayfalarına göz atmanız bu kavramları anlamak için oldukça faydalı olacaktır. Geridönüş aygıtları 7. anadüğümdeki blok aygıtlarıdır. Altdüğümler 0 dan başlar ve ardışık devam eder.

`/dev/md0`
Birinci meta disk grubu. Meta diskler RAID (Redundant Array of Independent Disk – Birleştirilmiş Bağımsız Disk Dizisi) aygıtları ile bağlantılıdır. Daha ayrıntılı bilgi için LDP'deki (Linux Document Project – Linux Belgelendirme Projesi) RAID–HOWTO belgelerine başvurabilirsiniz. Meta diskler 6. anadüğümdeki blok aygıtlarıdır. 0. altdüğümden başlar ve ardışık devam eder.

`/dev/mixer`
Bu OSS'nin (Open Sound Driver – Açık Ses Sürücüsü) bir parçasıdır. `http://www.opensound.com/` adresindeki OSS belgelerinden daha ayrıntılı bilgi edinebilirsiniz. 14. anadüğüm, 0. altdüğümdeki karakter aygıtıdır.

`/dev/null`
Gönderilen her verinin bir daha tekrar geri dönmemek üzere yok edildiği bir kara deliktir. `/dev/null`'a gönderilen her şey görünmez olur. Bu, bir komut çalıştırmak ama uçbirimde her hangi bir geri bildirim almamak istediğiniz durumlarda faydalı olabilir. 1. anadüğüm, 3. altdüğümdeki bir karakter aygıtıdır.

`/dev/psaux`
PS/2 fare portudur. 10. anadüğüm, 1. altdüğümdeki bir karakter aygıtıdır.

`/dev/pda`
Paralel port IDE diskleri. Dahili IDE denetleyicileri gibi isimlendirilirler. Blok aygıtlardır ve anadüğüm numarası 45 dir. Alltdüğüm numaraları için biraz daha fazla açıklamaya ihtiyaç vardır. İlk aygıt `/dev/pda`'dır ve aldüğüm numarası 0 dır. Bölüm numaraları altdüğüm numarasına eklenerek bu aygıttaki bölümler bulunur. Her aygıt 15 bölüm ile sınırlandırılmıştır. Dahili IDE aygıtlarında bu 63 dür. `/dev/pdb` için aldüğüm numarası 16'da başlar, `/dev/pdc` için 32'de, `/dev/pdd` için 48'de başlar. Örneğin; `/dev/pdc6` için altdüğüm numarası 38 (32+6) dir. Bu düzen her biri 15 bölümlü 4 paralel disk ile bizi sınırlar.

`/dev/pcd0`
paralel port CD ROM sürücüsü. 0'dan başlayarak numaralandırılırlar. 46. anadüğümdeki blok aygıtlarıdır. Altdüğümler `/dev/pcd0` için 0 dan başlar ve ardışık devam eder (`/dev/pcd1` için 1, `/dev/pcd2` için 2, `/dev/pcd3` için 3, ...).

`/dev/pt0`
Paralel port teyp sürücüsü. Teyplerin bölümleri olmadığı için sadece ardışık biçimde numaralandırılırlar. Karakter aygıtlardır ve anadüğüm numarası 96 dır. Altdüğümi `/dev/pt0` için 0, `/dev/pt1` için 1, `/dev/pt2` için 2, ... şeklinde sıralanır.

`/dev/parport0`
İşlenmemiş paralel port. Paralel porta bağlanan pek çok aygıt kendi sürücülerini kullanırlar. Bu ise porta doğrudan ulaşmak için kullanılan bir aygıttır. Anasüğüm numarası 99 olan karakter aygıtıdır. Alltdüğüm numarası 0'dan başlar ve ardışık olarak her bir aygıt için birer artarak devam eder.

`/dev/random` veya `/dev/urandom`

Bunlar çekirdeğin rasgele sayı üreteçleridir. `/dev/random` rasgele sayı üreten ve o an için üretilmiş sayıya bakıp gelecek sayısı tahmin edilebilen üreteçlere benzemez. Sayı üretmek için sistem donanımlarının arabirimlerinin ürettiği ganimetler (entropy)[4] kullanır. Kullanılacak ganimet kalmadığında daha fazlasının toplanması için beklemek zorunda kalır. Bu durumda iken daha fazla sayı okunmasına izin vermez. `/dev/urandom` da buna benzer şekilde çalışır. İlk olarak sistem donanımlarının ganimetlerini kullanır ama ganimetler bitince karışık rasgele sayı üreten bir formül üzerinden çalışmaya ve sayı vermeye devam eder. Şifrelenmiş çift anahtar üreteçleri gibi çok önemli amaçlar için bu sistem biraz güvensiz görünebilir. Şayet güvenlik birinci planda önemli ise `/dev/random`'u kullanın, eğer hız önemli ise `/dev/urandom`'u kullanın. Rasgele sayı üreteçleri anadüğüm numarası 1 olan karakter aygıtlarındandır. `/dev/random` için altdüğüm numarası 8, `/dev/urandom`/dev/urandom için ise 9'dur.

`/dev/zero`
: Pek çok sıfır elde etmenin en kolay yoludur. Bu aygıttan her okuduğunuzda veri sıfıra dönüşür. İçeriği önemli olmayan sabitlenmiş uzunluktaki bir dosya ile ilgilendiğiniz zamanlarda bu aygıt yararlı olabilir. 1. anadüğümün 5. altdüğümündeki karakter aygıtıdır.

6. Diskler ve Diğer Depolama Ortamları

"temiz bir diskte sonsuza kadar araştırma yapabilirsiniz."

Sisteminizi güncellediğinizde veya kurduğunuzda yapmanız gereken pek çok iş vardır. Diskiniz üzerinde bilgi depolayabilmek için dosya sistemleri yapmalısınız ve sisteminizin çeşitli bölümleri için yer ayırmalısınız.

Bu bölümde bütün bu öncelikli adımlar açıklanmaktadır. Genellikle, bir kere sistemi ayarladıktan sonra tekrar bu tür işleri yapmak zorunda kalmazsınız. Sadece disketler hariç. Yeni bir disk eklediğinizde veya disk kullanımını daha verimli bir hale getirmek istediğinizde bu bölüme tekrar dönmelisiniz.

Diskleri yönetmek için temel basamaklar şunlardır (bu basamak sıraları değişebilir):

- Diski biçemleyin. Bu kullanıma hazırlamak için gereken pek çok işlevi yerine getirir. Günümüzde diskler genellikle biçemlenmeye ihtiyaç duymazlar.
- Diski bölümleyin. Şayet bir bölümün diğerine karışmasını istemiyorsanız disk üzerinde bölümler (partition) oluşturun. Aynı disk üzerinde farklı işletim sistemleri kurmak istiyorsanız diski mutlaka bölümlemek zorundasınız. Diğer bir sebep de sistem dosyalarının kullanıcı dosyaları ile karışmasını önlemek olarak sayılabilir. Ayrıca yedekleme amacı ile de diski bölümlere ayırabilirsiniz.
- Her diskte ve her bölümde amacına uygun bir dosya sistemi yaratın. Siz bir dosya sistemi yaratmadıkça disk Linux'den hiçbir şey anlamayacaktır. Zaten bu durumda diski kullanmanız mümkün değildir. Dosya sistemi yaratıldıktan sonra (ext2, fat32, vfat, vb..) disk üzerine dosyalar kaydedilebilir.
- İster otomatik, ister el ile değişik dosya sistemlerini tek bir çatı altında toplayın. Unutmayın ki el ile bağlanmış bir dosya sistemi mutlaka el ile ayrılmalıdır.

Bellek Yönetimi (sayfa: 47) bölümü sanal bellek ve disk üzerindeki önbellek hakkında bilgiler içermektedir. Diski kullanırken bu bilgiler çok işinize yarayacaktır.

6.1. Aygıtların iki çeşidi

Linux ve Unix işletim sistemleri aygıtları iki şekilde tanırlar. Birinci rasgele erişimli blok aygıtları (diskler gibi), diğeri ise karakter aygıtlardır (seri hatlar ve teypler gibi). Sistemdeki bazı aygıtlar seri, bazıları rasgele erişimli olabilir. Her desteklenen aygıt dosya sisteminde bir aygıt dosyası ile temsil edilir. Bir aygıt dosyasına yazdığınız

veya okuduğunuz zaman veriler temsil edilen aygıttan gelir ve aynı aygıta gider. Aygıtlara ulaşmak için her hangi bir özel uygulama ya da programa ihtiyaç yoktur. Yazıcıya dosya göndermek istiyorsak sadece komut vermemiz yeterli olacaktır.

```
$ cat dosya > /dev/lp1
$
```

Tabii ki bu, böyle bir durumda sistemde dosya" adında bir dosya olmalı ve bu dosyanın içeriği yazıcının anlayabileceği bir yapıda olmalıdır. Bu komut sayesinde yazıcıdan dosya isimli dosyanın çıktısını alabiliriz. Bununla birlikte, sistem üzerinde pek çok kişi **cat** komutunu kullanarak yazıcıdan çıktı almaya çalışabileceği göz önünde tutularak, birisi de özel bir program kullanabilir. Bu genellikle **lpr**'dir. Bu program aynı anda sadece tek bir dosyanın basılacağını garanti eder ve bu dosya bittikten sonra diğerlerini otomatik olarak yazıcıya yollar. Buna benzer şeyler bütün aygıtlar için gereklidir. Aslında aygıt dosyaları için çok nadiren endişe etmemiz gerekir.

Aygıtlar sistemde dosyalar olarak gösterildiği için **ls** veya başka bir komut yardımı ile sistemde bulunan aygıtları (/dev dizininde) görebiliriz. **ls -l** komutunun çıktısında satırın en başında aygıt türünü ve erişim izinlerini görmek mümkündür. Örneğin seri bir aleti inceleyecek olursak;

```
$ ls -l /dev/ttyS0
crw-rw-r--    1 root     dialout    4,  64 Aug 19 18:56 /dev/ttyS0
$
```

Birinci satırın ilk karakteri, yani crw-rw-rw- harflerinin en başındaki 'c' harfi bu aygıtın türünü bize gösterir. Bu aygıt bir karakter aygıtıdır. Normal dosyalar için ilk karakter '-', dizinler için 'd', blok aygıtları için 'b', bağ dosyaları için ise 'l'dir. Daha ayrıntılı bilgiyi **ls** komutunın kılavuz sayfasında bulabilirsiniz.

Unutmayın ki sistemde olsun olmasın bütün aygıt dosyaları sistem içerisinde mevcuttur. Sisteminizde /dev/sda olması sizin SCSI sabit diske sahip olduğunuz anlamına gelmez. Bütün aygıt dosyalarına sahip olmak ilerde ekleyeceğiniz donanımlar için doğru parametreleri bulmak ve aygıt dosyası yapmak zahmetinden sizi kurtarır. Aynı amanda yükleme programlarını daha basit hale getirir. Bu sayede sistemi yüklerken her donanım aygıtınız için tek tek uğraşmak zorunda kalmazsınız.

6.2. Sabit diskler

Bu bölümde sabit disklere ait terimler ve terminolojiye yer verilmiştir. Şayet bu konuda bilgi sahibi iseniz bu bölümü atlayabilirsiniz.

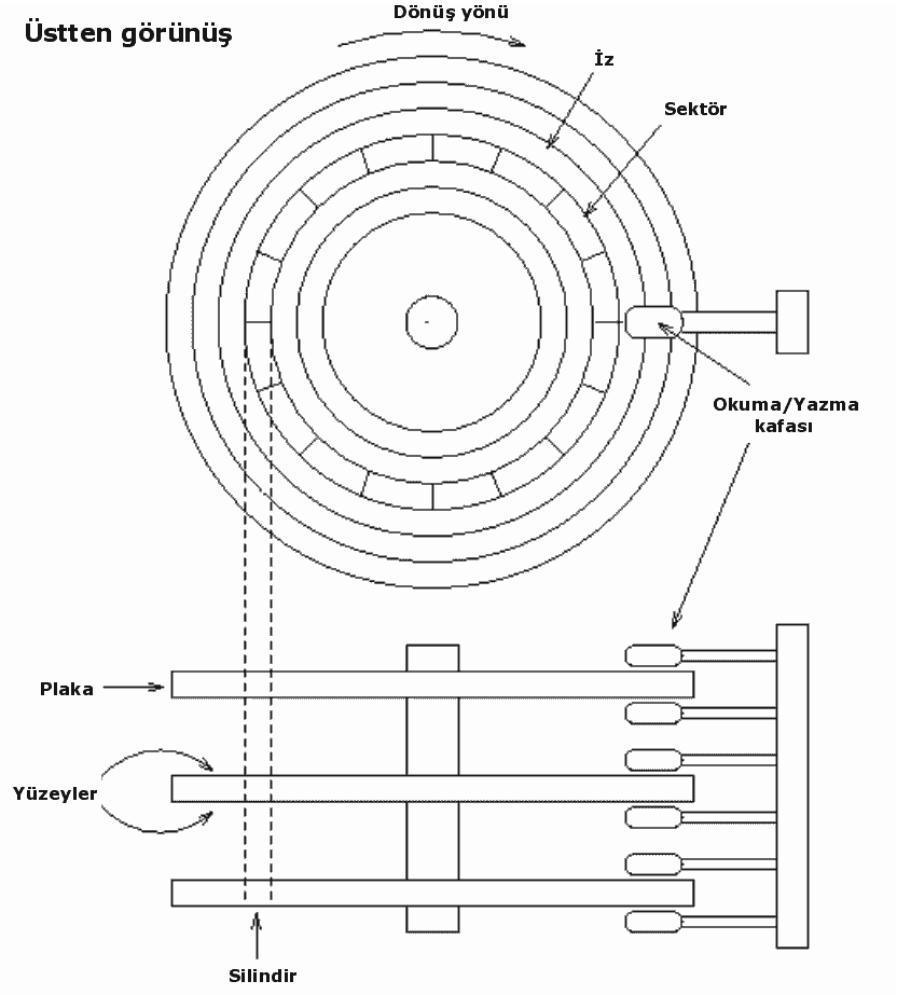
Bir sabit diskin yapısı (sayfa: 26) de bir sabit diskin önemli parçaları gösterilmiştir. Bir sabit disk bir veya daha fazla sayıda metal plaka ihtiva eder. Bu disklerin her iki *yüzeyi* de bilgi depolayabilmek için manyetik bir tabaka ile kaplanmıştır. Her yüzeyde bilgileri okuyan ya da değiştiren bir *okuma/yazma kafası* bulunur. Diskler ortak bir eksen/mil etrafında dönerler. Genellikle 5400 devir/dak ile 7200 devir/dak arası bir hıza sahiptirler. Yeni sabit disklerin bazıları daha hızlı olabilmektedir. Bazı eski sabit disklerin hızı ise daha düşüktür. Kafalar plakaların/disklerin yarı çapları boyunca hareket ederler. Bu hareket plakaların dönüşü ile birleştirilerek kafaların bütün plaka/disk yüzeyini okuması/işlemesi sağlanır.

İşlemci (CPU) ve disk, bir disk denetleyicisi vasıtası ile haberleşirler. Çeşitli türdeki diskler bilgisayarın işlemci dışında kalan kısmı ile haberleşmek için aynı arayüzü kullanırlar, bu sayede bilgisayarın geri kalan parçalarının disk ile haberleşmek için ne kullanmaları gerektiğine dair bir fikir elde etmeleri gerekmez ve bu da işlemleri kolaylaştırır. Bu sayede bilgisayar diske "hey disk ne istiyorsam çabuk ver" diyebilir. Bu işler böyle olmasa idi bilgisayar karmaşık ve uzun elektrik sinyallerini diskin kafasına yollayacak, kafa disk üzerinde belli bir konumda doğru verileri bulana kadar bekleyecek ve tekrar bunları bilgisayara iletecekti. Yapması gereken pek çok karışık ve hoş olmayan işlerle uğraşacaktı. Buda sistemin performansını olumsuz etkileyecekti. Aslında diskin kullandığı bu arayüz de karmaşık olarak nitelenebilir. Ama diğer yapılması gereken işlemlere göre çok daha az karmaşıklık ihtiva eder. Denetleyiciler otomatik hatalı sektör taraması ve düzeltmesi gibi değişik işler yapabilirler.

Yukarıdaki açıklamalar birisinin sabit diskleri anlaması için gereken açıklamaların hepsini içermektedir. Ayrıca sabit disklerde kafanın ve plakaların çalışmasını ve dönmesini sağlayan bir motor, mekanik bölümleri kontrol eden elektronik bölümler de vardır. Fakat bunların, sabit diskin çalışma mantığının anlaşılması ile doğrudan bir ilgisi yoktur.

Yüzeyler eş merkezli çemberlere ayrılmıştır. İz olarak adlandırılan bu bölümlerde *sektör*lere ayrılmıştır. Bu sayede disk yüzeyindeki yerleşimleri tanımlamak, bilgilerin bulunduğu yeri tespit etmek ve dosyaları disk yüzeyine yerleştirmek mümkün olabilmektedir. Disk yüzeyindeki bir yeri bulmak için "yüzey 3, iz 5, sektör 7" gibisinden bir tanımlama yapılabilir. Genellikle sektör sayısı bütün izler için aynıdır. Ama bazı sabit disklerde en dıştaki plakada daha fazla sektör olabilmektedir. Bütün sektörler aynı fiziksel alana sahiptirler, bu nedenle sabit disklerin dış plakasında daha fazla sektör vardır. Genel olarak bir sektör 512 byte bilgi içerir.

Şekil 3. Bir sabit diskin yapısı



Aynı yolla bütün yüzeyler iz ve sektörlere ayrılmıştır. Bir yüzeydeki bir kafa bir iz üzerindeyken diğer yüzeydeki kafa da ona mukabil gelen iz üzerindedir. Bütün karşılıklı izlerin hepsine birden silindir denir. Bir izden diğerine geçmek zaman alır. Sık sık kullanılan dosyalar aynı silindir üzerine yazılarak kafaların bütün disk yüzeyinde hareket etmeleri mecburiyeti ortadan kaldırılmış olur. Bu sayede zamandan tasarruf sağlanır ve performans artışı elde edilir. Tabii bu her zaman mümkün olmaz, bu durumdaki dosyalara *parçalanmış* (fragmented) dosyalar denir.

Kafaların veya yüzeylerin sayısı, ki ikisi de aynı şeyi ifade eder (her yüzeye bir kafa), silindirler ve sektörler bir sabit diskte çok çeşitlilik gösterebilirler. Her birinin numaralarının tarifine diskin geometrisi denir. Bu geometri

CMOS RAM diye anılan batarya destekli özel bir bellek bölümünde saklanır. İşletim sistemi açılış sırasında ya da sürücü başlangıcında gidip bu bölümden gerekli bilgileri alır.

Büyük bir şanssızlık eseri CMOS RAM'de iz sayısı 1024 ile sınırlandırılmıştır. Bu büyük sabit diskler göz önüne alındığında oldukça küçük bir rakam olarak kalmaktadır. Bu problemin üstesinden gelmek için denetleyici bilgisayara yalan söyler (günümüzde bu tür problemler ortadan kalkmıştır. Pek çok yerde standart olarak 60–80Gb sabit diskler satılmakta ve BIOS ayarlarında LBA kipinin etkinleştirilmesi sayesinde 1024 silindir sınırı ortadan kalkmaktadır).

Bu sınır SCSI diskler için söz konusu değildir. Çünkü SCSI diskler işlemci ile farklı bir yol ile haberleşirler. SCSI disk denetleyicileri ardışık sektör numaraları sayesinde haberleşirler. Bilgisayar bu durumda diskin gerçek geometrisini asla bilmemektedir.

Linux, böyle durumlarda diskin gerçek yapısını bilememekte olup, asla dosyaları bir silindir içine depolamaz, onun yerine ardışık sektör numaralarının olduğu bölüme bilgileri yazar, ki bu sistemde diğeri kadar iyi performans verir. Buradaki en önemli nokta bunların ve daha fazlasının denetleyicinin önbelleği sayesinde yapılıyor olmasıdır. Bütün bu işlemler otomatik olarak yönetilir.

Her sabit disk bağımsız birer aygıt dosyası ile temsil edilir. Genellikle 2 veya 4 adet IDE disk sisteme bağlı haldedir (bir veya üç adet disk olmasının IDE sabit diskler için hiçbir mahsuru yoktur.). Bunlar `/dev/hda`, `/dev/hdb`, `/dev/hdc` ve `/dev/hdd` olarak sıralanır ve adlandırılırlar. SCSI diskler ise `/dev/sda`, `/dev/sdb`, `/dev/sdc`, ... olarak sıralanırlar. Sabit disk çeşitleri ve isimleri hakkında daha ayrıntılı bilgi *Aygıt Dosyaları* (sayfa: 21) bölümünde mevcuttur. Unutmayın ki bu isimler disk bölümlerine ulaşmak için değil, bütün sabit diske ulaşmak için geçerlidir. Böyle durumlarda çok dikkatli olunmazsa bilgiler ve bölümler karıştırılabilir. Disk aygıt dosyaları sadece MBR'ye (Master Boot Record – Ana Önyükleme Kaydı) ulaşmak için kullanılırdı. Bu konu da ilerde açıklanacaktadır.

6.3. Disket Sürücüleri

Diketler esnek bir ince zar ile yüzeyi kaplanmış, bir ya da her iki yüzü de manyetik alan haline getirilmiş, sabit disklere benzeyen ama saklama kapasiteleri daha küçük olan aygıtlardır. Disket üzerinde bir okuyucu–yazıcı kafa bulunmaz. Disketin içinde bir adet plaka vardır. Bu taşınabilir donanım aygıtları her hangi bir disket sürücü içine konulup okuma ve yazma işlemi yapabilirler. Aynı disket sürücü ile pek çok disketi okumak mümkündür. Bu özellikleri ile taşınmaz hard disklerden ayrılırlar.

Tıpkı sabit diskler gibi disketlerde izler, sektörler ve silindirlere ayrılırlar fakat bunların sayısı sabit diskin yanında çok küçük kalmaktadır.

Bir disket sürücü çeşitli disketleri kullanabilir. Örneğin; 3,5" lik bir disket sürücü 3,5" çapına sahip 720 Kb ve 1,44MB lık disketleri kullanabilir. Sürücü biraz farklı işlemler gerçekleştirebildiği ve işletim sisteminin disketin kapasitesi hakkında bilgi sahibi olması gerektiği için pek çok çeşit disket sürücü aygt dosyası vardır. Bunlar sürücünün çeşidi ve birleşimlerine göre çeşitlilik gösterebilirler. `/dev/fd0H1440` birinci disket sürücü (fd0), 3,5" çapında ve yüksek yoğunlukta (H) 1440 KB bir disketi temsil eder. Bu da bugünlerde kullanılan normal bir diskete tekabül eder.

Disket sürücülerin isimleri biraz karmaşıktır, bu nedenle Linux sürücüde hangi tür disket olduğunu otomatik olarak tespit eden özel bir disket aygıt tipine sahiptir. Yeni yerleştirilen bir disketin ilk sektörünü okur ve elindeki örneklerle karşılaştırır. Bu işlem doğru disket aygıtını bulana kadar devam eder. Disketin önceden biçemlenmiş olması gerekir. Bu otomatik sürücü aygıtları `/dev/fd0`, `/dev/fd1`, ... şeklinde sıralanırlar.

Linux standart disketlerin yanında pek çok standart dışı disketleri de desteklemektedir. Şimdi bu konuya geçeceğiz ama ondan önce `/etc/fdprm` dosyasına göz atmanızda fayda olabilir. Bu dosya **`setfdprm`** komutunun ayarlarını içerir.

Sürücüdeki disketler değiştiği zaman işletim sistemi bundan haberdar edilmelidir. Yoksa sistem önbelleğe almış olduğu disket bilgileri üzerinden işlem yapmaya devam eder. Bunun için kullanılan sinyal hattı bazen kırılmış veya bozulmuş olabilir veya MSDOS biçeminde bir disket için bunu dikkate almayabilir. Şayet disket kullanırken garip hatalar ile karşılaşıyorsanız, belki de sebebi budur. Bunun için sürücüyü tamir etmekten başka bir yol yoktur.

6.4. CD–ROM'lar

Bir CD sürücüsü plastik kaplı bir diski optik okuyucusu yardımı ile okur. Bu diskte kaydedilmiş olan bilgiler, yüzey üzerinde merkezden kenarlara kadar spiral biçimde uzanan "delikler" içinde kayıtlı olarak tutulurlar. Sürücü buraları okumak için lazer ışını kullanır. Lazer bir deliğe çarptığında farklı, düz bir yüzeye çarptığında farklı bir tepki verir. Okuyucu kafa bunları "0" ve "1" ler olarak alır ve disk yüzeyindeki bilgiyi çözümler.

CDROM'lar sabit disklere nazaran daha yavaştırlar. Ortalama bir sabit diske tarama için 15 milisaniye yeterli olurken, en hızlısından bir CDROM için saniyeler gereklidir. Aslında bir CDROM'un veri aktarım hızı saniyede yüzlerce KB ile ölçülür. Burada ifade edilmek istenen CDROM'un sabit diskler kadar hızlı olamadığıdır. Bazı Linux dağıtımları; "canlı" CDROM dosya sistemleri ile dağıtılmakta ve bu sayede bilgiler sabit diske kopyalanmak zorunda kalmamakta, bunun sonucunda hatırı sayılır bir disk alanı tasarruf edilmektedir. Yeni bir yazılımı yüklemek için CDROM'lar iyi bir alternatif sayılırlar. Ne de olsa yükleme sırasında çok yüksek hız gerekli değildir.

Bir CDROM'daki bilgileri düzenlemenin pek çok yolu vardır. En popüler yöntem uluslararası standart olan iso9660 dır. Bu standart MSDOS'un kullandığından daha "ham" bir dosya sistemini tanımlar. Oldukça küçüktür. Diğer taraftan bu kadar küçük olması her türlü işletim sisteminin onu tanımasına olanak verir.

Normal bir Unix kullanımı esnasında iso9660 yeterli gelmemektedir. Bu nedenle "Rock Ridge extension" denen bir ekleme yapılmaktadır. Rock Ridge, uzun dosya isimlerine, sembolik bağlara ve daha pek çok şeye izin verir. CDROM'un az çok Unix dosya sistemine benzemesini sağlar. İşin iyi tarafı Rock Ridge dosya sistemi geçerli bir iso9660 dosya sistemi gibi kullanılabilmekte ve diğer dosya sistemine sahip bilgisayarlar ile kullanılabilmektedir. Linux iso9660'ı ve Rock Ridge dosya sistemini desteklemektedir. Uzantılar otomatik tanımlı ve kullanımlıdır.

Dosya sistemi problemin yarısıdır. Pek çok CDROM'un içindeki bilgilerin okunabilmesi için özel programlar gerekmekte ve bu programların pek çoğu Linux altında çalışmamaktadır (dosemu Linux Msdos öykünümü altında ve wine windows öykünümü hariç. Ayrıca ticari bir yazılım olan, X86 makinesindeki yazılımları komple çeviren VMWARE de bunların içinde sayılabilir).

Bir CDROM sürücüye, karşılıklı aygıt dosyası üzerinden ulaşılır. Bir CDROM sürücüyü, bilgisayara bağlamanın çok çeşitli yolları vardır: ses kartı üzerinden bağlayabilirsiniz, SCSI veya EIDE üzerinden bağlayabilirsiniz. Donanım "hacking" bu kitabın konusu dışındadır fakat bağlantı şekli hangi aygıt dosyasının kullanılacağına karar verir.

6.5. Teypler

Bir teyp sürücü müzik kasetlerine benzeyen bantlar kullanırlar. Normalde teypler seri aygıtlardır. Bu nedenle bir yeri okumak için o yere kadar gitmeniz gerekir. Bir diske ise rasgele erişim mümkündür. İsterseniz diskin her hangi bir yerine doğrudan ulaşabilirsiniz. Seri erişimli teypler bu yüzden nispeten daha yavaştırlar.

Diğer taraftan hızlı olmaları gerekmediği için teypler oldukça ucuzdur. Oldukça uzun olabilirler ve bu sayede daha fazla veri saklayabilirler. Arşivleme ve yedekleme işlemleri için büyük kapasiteli ama hızlı olması gerekmeyen ucuz bir yöntem arıyorsanız teypler sizin için çok uygundur.

6.6. Biçemleme

Biçemleme izlerin ve sektörlerin işaretlenmesi için kullanılan, manyetik ortama işaretlerin yazılması sürecidir. Biçemlenmiş bir disk yüzeyi karışık manyetik işaretler ile doludur. Biçemlendiği zaman çizgilerin gittiği izler ve

bölümlere ayrılmış sektörler haline dönerek karmaşıklığa bir son verilmiş olur ve disk yüzeyine düzen gelir. Her biçemleme işlemi tam olarak bu şekilde meydana gelmeyebilir. Burada önemli olan biçemlenmemiş bir diskin kullanılamayacak olmasıdır. Kısaca biçemleme, bir diskin manyetik olarak adımlanması ve dosya sistemi haline (bu MSDOS için geçerlidir) getirilmesi demektir.

Terminoloji burada biraz çelişkilidir: MS–DOS ve MS Windows'da, biçemleme dosya sistemini oluşturulmasını da kapsar. Bu iki işlem özellikle disketler için birleşiktir. Gerçek biçemleme düşük seviyeli biçemleme olarak bilinirken, dosya sisteminin oluşturulması *yüksek seviyeli biçemleme* olarak bilinir. Bu kitabın kapsamında bu ikisi biçemleme ve dosya sistemi oluşturma olarak ele alınacaktır.

Pek çok kişi biçemleme işlemlerinden endişe duyduğu için IDE ve bazı SCSI diskler fabrika çıkışlı orijinal biçemlidirler. Ve bu işlemin tekrarlanması gerekmez. Bu da pek çok endişeyi ortadan kaldırır. Gerçekten de bir sabit diskin biçemlenmesi onun performans kaybına uğramasına sebep olabilir. Disk yüzeyindeki bozuk sektörlerin bazen özel bir program tarafından onarılması gerekiyor olabilmektedir.

Biçemlenmesi gereken veya gerekebilecek olan disklerin biçemlenmesi için özel programlara sıklıkla ihtiyaç olmaktadır. Çünkü disk sürücü içinde bulunan biçemleme arayüzü aygıttan aygıta değişiklik gösterebilmektedir. Biçemleme programını BIOS içinde veya MS DOS programı şeklinde bulabilirsiniz. Şayet ikisi de yoksa Linux içinde size uygun olabilecek bir tanesini mutlaka bulursunuz.

Biçemleme esnasında hatalı sector / *hatalı iz* olarak adlandırılan bozuk sektör ve bozuk izler ile karşılaşmak olasıdır. Bunlar genellikle sürücü tarafından kolaylıkla halledilecek problemlerdir, fakat bu bozuk alanlar ilerlemiş ise bunların kullanımını ve dolayısı ile ilerde meydana gelebilecek veri kayıplarını önlemek için bazı programlar kullanılabilir. Bunu yapmak için aşağıda dosya sistemine nasıl bilgi eklenebileceğini anlatan bölümle aynı mantığı kullanabilirsiniz. Alternatif olarak bu bozuk kısımları kapsayan küçük bir disk bölümü (partition) yaratılabilir. Bu yaklaşım, genişlemiş bozuk yüzeylerin olduğu durumda daha akıllıcadır. Çünkü dosya sistemleri, geniş bozuk alanlar ile pek iyi geçinemezler.

fdformat komutu ile disketleri biçemlendirebilirsiniz. Disket aygıt dosyası gerekli parametreleri verecektir. Örneğin aşağıdaki komut birinci disket sürücüde bulunan 3,5" HD bir disketi biçemlemeye yarar.

```
$ fdformat /dev/fd0H1440
Double-sided, 80 tracks, 18 sec/track. Total capacity
1440 kB.
Formatting ... done
Verifying ... done
$
```

Şayet otomatik olarak tespit edilen bir aygıt kullanmak istiyorsanız (`/dev/fd0` gibi). Öncelikle **setfdprm** ile parametrelerin ayarını yapmalısınız. Aşağıdaki örnek bunun yolunu göstermektedir.

```
$ setfdprm /dev/fd0 1440/1440
$ fdformat /dev/fd0
Double-sided, 80 tracks, 18 sec/track. Total capacity
1440 kB.
Formatting ... done
Verifying ... done
$
```

Genellikle, disketin tipi ile uyuşan aygıt dosyasını seçmek daha uygun olur. Disketler hangi bilgiler ve parametreler ile tasarlanmışlarsa, o seçeneklere uygun olarak biçemlendirilmelidir.

fdformat ayrıca disket yüzeyindeki bozuk sektörleri de tarar ve geçerli hale getirir. Bozuk sektörler üzerine yazmak için defalarca uğraşır, bunu değişen disket sürücü sesinden de anlayabiliriz. Şayet okuma/yazma kafasındaki tozdan veya buna benzer şeylerden meydana gelen bir hata var ise **fdformat** bunu düzeltene kadar uğraşır, fakat daha önemli hatalar var ise program kesintiye uğrar ve işlem iptal edilir. Çekirdek bulduğu her

G/Ç hatası için bir mesaj yayınlar. Bu konsola veya ayarlı ise **syslog** için /usr/log/messages dosyasına yönlendirilir. **fdformat** hata bulduğu zaman bunu kendiliğinden bildirmez. Kullanıcının buna dikkat etmesi gerekir. Zaten disketler, bozulduklarında kolayca çöpe atılabilecek kadar ucuzdurlar.

```
$ fdformat /dev/fd0H1440
Double-sided, 80 tracks, 18 sec/track. Total capacity 1440 kB.
Formatting ... done
Verifying ... read: Unknown error
$
```

badblocks komutu disketler dahil her hangi bir diski veya disk bölümünü kontrol etmek için kullanılabilir. Biçemleme işlemi yapmadığı için hali hazırda kurulu olan dosya sistemleri üzerinde de denenebilir. Aşağıdaki örnek 3,5 inçlik bir diskette iki adet bozuk blok bulduğunu göstermektedir.

```
$ badblocks /dev/fd0H1440 1440
718
719
$
```

badblocks komutu bulduğu bozuk blokların numaralarını ekrana çıktı olarak verir. Pek çok dosya sistemi bozuk bloklardan uzak durmaya çalışır. Dosya sisteminin kurulumu esnasında tespit edilen bozuk blokların listesi tutularak, daha sonra bunlar üzerinde değişiklikler yapılır. İlk tarama, dosya sistemini başlangıç durumuna getiren **mkfs** komutu ile yapılabilir. Fakat daha sonraki taramalar mutlaka **badblocks** komutu ile yapılmalıdır. Ve yeni tespit edilen bozuk bloklar **fsck** ile eklenmelidir. **mkfs** ve **fsck** daha sonra anlatılacaktır.

Yeni disklerin pek çoğu bozuk blokları otomatik olarak bulmakta ve özel bir işlem olarak onları saklanmış (reserved) sağlam bloklar ile değiştirmeye çalışmaktadır. Bu işletim sistemi tarafından görülemez. Şayet sizin diskinizin de bu özelliğe sahip olup olmadığını merak ediyorsanız diskin kullanım kılavuzuna bakın, eğer böyle bir özellik var ise burada mutlaka belirtilmiş olması gerekir. Eğer disk içindeki bozuk blok sayısı iyice artarsa, disk kullanılamaz hale gelebilir. Ve diskiniz ile vedalaşmak zorunda kalabilirsiniz.

6.7. Disk Bölümleri

Bir sabit disk çeşitli bölümlere ayrılabilir. Her bölüm ayrı bir sabit diskmiş gibi faaliyet gösterebilir. Diyelim ki; bir sabit diskiniz var ve siz iki adet işletim sistemine sahip olmak istiyorsunuz. Bu durumda sabit diski ikiye bölmeniz gerekecektir. Her işletim sistemi kendi bölümünü kullanır ve diğerininkine dokunmaz. Bu yöntemle iki ayrı işletim sistemi aynı disk üzerinde kardeşçe yaşayabilir. Bölümleme olmasa idi her kurmak istediğimiz işletim sistemi için ayrı bir sabit disk almamız gerekecekti.

Disketlerin bölümlenmesine teknik bir engel olmamasına rağmen, zaten çok küçük olmalarından dolayı böyle bir şeye pek ihtiyaç olmamaktadır. Keza CDROM'larda da bölümleme gerekmemekte ve bir CDROM'u büyük bir bölümmüş gibi kullanabilmekteyiz. Zaten bir CD ROM üzerinde birden fazla işletim sistemi bulunmasını gerektirecek pek fazla durumla karşılaşılmamaktadır.

6.7.1. MBR, önyükleme sektörleri ve bölümleme tablosu

Bir sabit diskin nasıl bölümlendiğine ait bilgi, birinci plakanın birinci izinin içindeki birinci sektörde yer alır. Makine ilk açıldığı anda BIOS'un okuyup çalıştırdığı bu ilk sektöre Ana Önyükleme Kaydı (MBR – Master Boot Record) adı verilir. MBR içerisinde bölümleme tablolarının ve hangi bölümüm etkin olduğunu belirten küçük bir program vardır. Ayrıca bu etkin bölümün açılış sektörü de burada belirtilir.(MBR aynı zamanda bir önyükleme sektörü olmasına rağmen özel durumundan dolayı ayrı bir adla anılır.) Önyükleme sektörü içinde de işletim sisteminin başlamasını sağlayan küçük bir program vardır. Bu program işletim sisteminin ilk bölümünü okur ve sistemi başlatır.

Bölümleme işlemi donanımla ya da BIOS ile doğrudan alakalı bir şey değildir. Bu işletim sistemlerinin geleneksel olarak kabul ettikleri ve uyguladıkları bir işlemdir. Pek çok işletim sistemi disk bölümlenmesine olanak verir. Bazıları ise istisnadır ve disk bölümlemesine izin vermezler. Bazı işletim sistemleri bölümlemeyi desteklerler fakat, sabit disk üzerindeki bir bölümü işgal edip orada bulunan bölümleme yöntemi ile diski bölerler. Diğerleri ise, Linux dahil, başka bir işletim sistemi ile aynı disk üzerinde hiçbir problem çıkarmadan yaşayabilirler. Özel bir şey yapmamıza gerek kalmaz. Ama disk bölümlemeyi desteklemeyen bir işletim sistemi ile aynı disk üzerinde başka bir işletim sistemi kuramazsınız.

Bir güvenlik önlemi olarak bölümleme tablosunu bir kağıda yazmak akıllıca olacaktır. Şayet bir arıza olursa bu yolla düzeltme imkanınız olur. Bozulmuş bir bölümleme tablosu **fdisk** ile düzeltilebilir. **sfdisk -l** komutu ile sisteminizdeki tüm sabit disklerin bölümleme tablolarını görebilirsiniz.

```
# sfdisk -l
Disk /dev/hda: 4870 silindir, 255 kafa, 63 sektör/iz
birimler = 8225280 baytlık silindir, 1024 baytlık blok, 0'dan başlayarak

   Aygıt Önykl Başlangıç Bitiş silindir blok   Kiml Sistem
                              sayısı sayısı
/dev/hda1   *      0+     65      66-   530113+   b  Win95 FAT32
/dev/hda2         66    4869    4804  38588130    f  Win95 Ext'd (LBA)
/dev/hda3          0       -       0         0    0  Boş
/dev/hda4          0       -       0         0    0  Boş
/dev/hda5         66+   1266    1201-  9647001   83  Linux
/dev/hda6       1267+   2467    1201-  9647001    b  Win95 FAT32
/dev/hda7       2468+   3646    1179-  9470286   83  Linux
/dev/hda8       3647+   4869    1223-  9823716   83  Linux
#
```

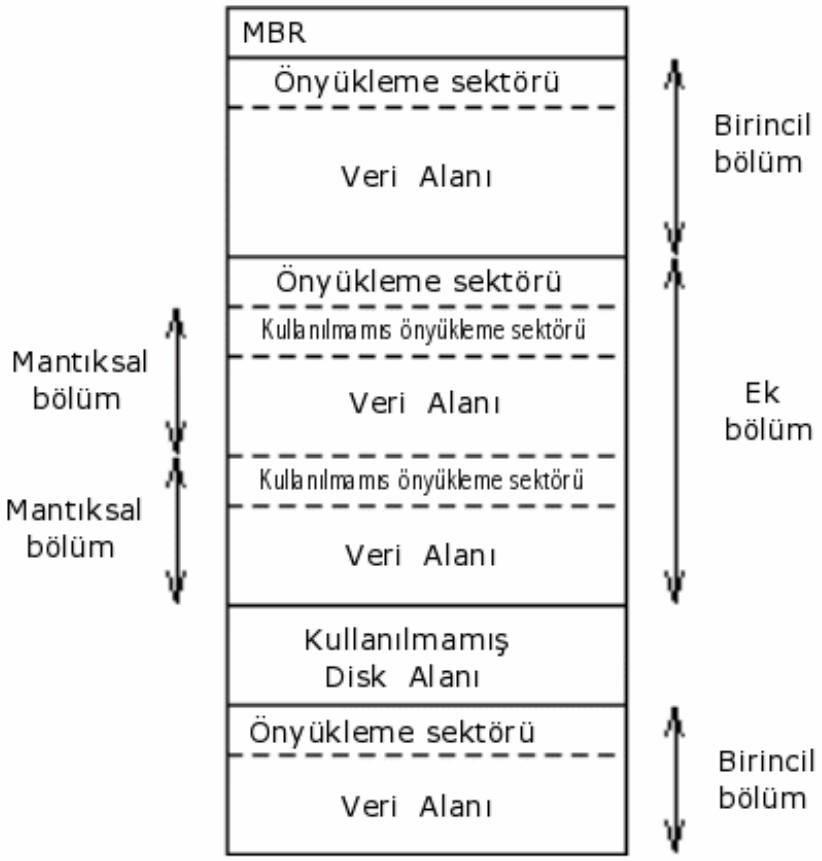
6.7.2. Ek ve Mantıksal Bölümler

PC sabit disklerinin orijinal bölümleme planı sadece dört bölüme izin verir. Gerçek hayatta bu sınırlama bazı insanlar için hiç hoş bir olay olmamaktadır. Çünkü onlar bilgisayarlarında dört taneden fazla işletim sistemi (Linux, MS DOS, OS/2, FreeBSD, NetBSD, Windows NT, vb.) kurmak istemektedirler. Aslında bir işletim sistemi bulunan diskleri de bölümlemek oldukça iyi bir fikirdir. Örneğin hızdan feragat etmemek için takas alanını, Linux'un kendi bölümüne değil de, takasın kendine ayrılmış olan bölüme kurmak daha akıllıcadır (disk üzerinde Linux ve Linux takas olarak iki bölüm yapabilirsiniz – yukarıdaki /dev/hdd gibi).

Bu sorunu aşmak için ek bölümler kullanılmaktadır. Bu hile *birincil bölüm*ü alt bölümlere ayırmaya izin verir. Birincil bölüm *ek bölümler* adıyla alt bölümlere ayrılır. Bu alt bölümler *mantıksal bölümler*dir. Bu bölümler birincil bölüm gibi davranırlar fakat yaratılma biçimleri farklıdır. Bu bölümler ile birincil bölümler arasında bir hız farkı yoktur.

Bir sabit diskin bölümlemesi *Bir sabit diskin bölümleme örneği* (sayfa: 31)'ye benzetilebilir. Disk üç adet birincil bölüme, bunlardan ikincisi ise iki adet mantıksal bölüme ayrılmıştır. Disk tamamen bölümlenmemiştir. Her birincil bölümün kendine ait bir önyükleme sektörü vardır.

Şekil 4. Bir sabit diskin bölümleme örneği



6.7.3. Bölüm türleri

Bölümleme tabloları (biri MBR'da diğeri ek bölümde) her bölüm için bölüm türünü tanımlayan birer baytlık bilgi içerir. Bu sayede bölümü kullanan işletim sistemine neyi kullandığı bildirilir. Buradaki amaç büyük ihtimal ile iki ayrı işletim sisteminin tek bir bölümü kullanmasını engellemektir. Bununla beraber ,özellikle Linux; gerçekte bölüm türü byteları ile ilgilenmez. Daha da kötüsü bazı işletim sistemleri bu bilgileri yanlış kullanır. DR DOS'un bazı sürümleri iletilen byte sinyallerini önemsemezler ve yok sayarlar.

Byte'ların anlamları standartlaşmamıştır fakat genel kabul gören bazıları aşağıda sıralanmıştır. Tam bir listesi Linux **cfdisk** programında bulunabilir.

Bölüm türleri

0	Boş	40	Venix 80286	94	Amoeba BBT
1	DOS 12–bit FAT	51	Novell?	a5	BSD/386
2	XENIX kök	52	Microport	b7	BSDI fs
3	XENIX usr	63	GNU HURD	b8	BSDI takas
4	DOS 16–bit FAT <32M	64	Novell	c7	Syrinx
5	Linux Ek	75	PC/IX	db	CP/M
6	DOS 16–bit >=32M	80	Old MINIX	e1	DOS erişimi
7	OS/2 HPFS	81	Linux/MINIX	e3	DOS R/O
8	AIX	82	Linux takas	f2	DOS ikincil
9	AIX önyük-lenebilir	83	Linux	ff	BBT
f	Win95 Ek (LBA)	93	Amoeba		

6.7.4. Bir sabit diskin bölümlenmesi

Bölüm oluşturmak ve kaldırmak için pek çok program kullanılabilir. Pek çok işletim sistemi kendine ait bir program kullanır. Disk bölümlemek için bu programı kullanmak daha akıllıca olur, çünkü işletim sistemi ile alakalı bazı işleri diğer programlar yapamazken, bu program alışılmadık şeyleri de yapabilir. Bu programların pek çoğu **fdisk** olarak adlandırılırlar. Linux **fdisk** hakkında ayrıntılı bilgiyi kılavuz sayfasında bulabilirsiniz. **cfdisk** komutu **fdisk** ile aynı işi yapar ama daha hoş bir tam ekran arayüze sahiptir.

IDE diskler kullanılırken, /boot dizinin bulunduğu açılış bölümü (açılış yapabilen çekirdek biteşlem dosyalarının bulunduğu bölüm) ilk 1024 silindir içinde olmak zorundadır. Bunun sebebi açılış esnasında sistem korumalı kipe geçmeden önce açılışın BIOS tarafından yapılıyor olmasıdır. BIOS, 1024 silindirden fazlasını okuyamaz. Bazen sadece bir bölümü 1024 silindirlik sınırın içinde olan açılış bölümleri kullanmak mümkün olabilir. Bu sadece, okunması gereken dosyaların ilk 1024 silindir içinde olması durumunda işe yarar. Bunu ayarlamak çok zor olduğu için, bunu yapmaya kalmak hiç de iyi bir fikir değildir. Çekirdek güncellemesi veya defrag (disk birleştirilmesi) sonrası sistemimizde açılış özelliği kalmamış olan bir işletim sistemi ile karşı karşıya kalabiliriz. Bu nedenle açılış dosyalarının (/boot dizini) ilk 1024 silindir içinde olduğundan emin olun. [5]

Bazı yeni BIOS ve sabit disklerde bu 1024 silindir sorunu yoktur. Şayet böyle bir sistem kullanıyorsanız bu problemle asla karşılaşmazsınız. Emin olamıyorsanız /boot dizininizin ilk 1024 silindir içinde kalmasını sağlamaya özen gösterin.

Her bölüm çift sektör sayısına sahip olmalıdır. Linux işletim sistemi bilgileri ikişer sektör halinde yani birer kb.lık paketler halinde okur. Sonda kalmış tek sektör sistem tarafından okunmayacaktır. Bu her hangi bir problem yaratmaz ama bazı **fdisk** sürümleri sizi bu konuda uyarabilir.

Bir bölümün boyutunu değiştirirken ilk önce kurtarmak istediğiniz bilgileri yedekleyin, sonra bölümü silip yeni bölümü oluşturun ve yedeklenmiş bilgileri geri yükleyin. Şayet bölüm boyutunu değiştiriyorsanız ve bunu diğer bölümlerden alan alarak yapıyorsanız bu işlemleri (yedekle–sil–oluştur–yükle) onlara da yapmak zorundasınız.

Bölümlerin boyutlarını değiştirmek çok sancılı ve sıkıntılı bir işlem olduğundan, bunu en başta bir kere yapmak, sağlam ve kullanışlı bir yedekleme programı kullanmak daha mantıklıdır. Eğer dışarıdan pek fazla müdahaleye ihtiyaç duymadan bir araçtan yükleme yapıyorsanız (CD ROM gibi) en başta kolaylıkla çeşitli yapılandırmalar deneyebilirsiniz. Elimizde yedeklenmiş ve tekrar yüklenilmesi gereken bir bilgi yığını olmadığı, daha doğrusu boş bir diske sahip olduğumuz için, çeşitli defalar yükleme yapmak, yedeklenmiş bilgileri geri yüklemek kadar zor değildir.

MS DOS sistemleri için **fips** adında bilgileri yedeklemek ve yeniden yüklemek zorunluluğu olmadan, MSDOS bölümlerinin boyutunu ayarlayabilen bir program vardır. Fakat diğer işletim sistemleri için bu hala gereklidir. **fips** programı pek çok Linux dağıtımında bulunur. Ayrıca ticari bir dağıtım olan "partition magic" programı da buna benzer özelliklere sahiptir. **parted** isimli GNU programı diğer dosya sistemlerini de bölümleyebilir. Unutmayın ki bölümleme ve yeni boyut verme işlemi her zaman risk taşır. Mümkün ise bütün bilgileri ayrı bir ortamda yedekleyin ve bu yedeklerin sağlıklı olduğundan emin olun.

6.7.5. Aygıt dosyaları ve disk bölümleri

Her bölüm ve ek bölüm kendi aygıt dosyasına sahiptir. Geleneksel isimlendirme yöntemine göre aygıt isminden sonra bir numara gelmektedir. Yine geleneksel olarak 1–4 arası numaralar kaç adet olduğuna bakılmaksınız birincil bölümlere ayrılır. 5 ve daha sonrakiler ise mantıksal bölümlere aittir. Buradaki önemli nokta sistemde kaç adet birincil veya mantıksal bölüm olduğunun ÖNEMLİ OLMADIĞIDIR. Örneğin /dev/hda1 birinci IDE sabit diskteki ilk birincil bölümdür. /dev/sdb7 ikinci SCSI sabit diskteki üçüncü ek bölümdür.

7. Dosya Sistemleri

7.1. Dosya sistemleri nedir?

Dosya sistemi disk üzerindeki dosyaların organize edilmesidir. Bir işletim sisteminin bir disk veya bölümleri üzerindeki dosyalarının izlerini bulmak için kullandığı yapı ve yönteme dosya sistemi (filesystem) denir. Ayrıca dosya sistemi terimi, dosyaların veya dosya sistemlerinin depolandığı bir disk veya disk üzerindeki bir bölümü tanımlamak için de kullanılabilir. Bu nedenle birisi "ben iki adet dosya sistemine sahibim" derken; aslında disk üzerinde her birinde dosyaların tutulduğu, iki adet bölüme veya ek bir disk bölümüne sahip olduğu anlaşılmalıdır.

Bir disk veya disk bölümü ile dosya sistemi arasındaki farklılık ne ihtiva ediyor olduğuna göre önem arz eder. Çok az program işlenmemiş disk veya bölüm yüzeyinde işlem yapabilir. Buna dosya sistemi yapabilen programlar dahildir. Şayet orada bir dosya sistemi var ise bu programların kullanılması sonucu silinir veya büyük ölçüde hasar görürler. Pek çok program ise dosya sistemleri üzerinde çalışır; olmayan veya yanlış parametreler içeren bölümler üzerinde çalışamazlar. Genelde dosya sistemi olmayan bir disk yüzeyinde bir program çalışmaz. Programların çalışması için bir dosya sistemine ihtiyaç vardır; dosya sistemlerinin olabilmesi için de bir disk veya disk bölümüne ihtiyaç duyulur.

Bir disk veya disk bölümü dosya sistemi olarak kullanılmadan önce, disk yüzeyi ilk haline döndürülmeli ve gerekli bilgiler diske yazılmalıdır. Bu işlem dosya sistemi oluşturma olarak adlandırılır.

Pek çok Unix dosya sistemi küçük farklılıklar dışında benzer bir genel yapıya sahiptirler. Genel kavramlar olarak superblok, *dosya düğümü* (inode), *veri bloğu*, *dizin bloğu* ve *dolaylı blok* sayılabilir. Superblok, dosya sisteminin bütünü hakkında bilgi içerir. Aslında bu dosya sistemlerine göre değişiklik gösterebilir. Dosya sisteminin boyutu gibi bilgiler burada yer alır. Dosya düğümü ise bir dosya hakkında, ismi hariç, bütün bilgileri ihtiva eder. Dosya ismi dizin içinde dosya düğümünün numarası ile birlikte yer alır. Bir dizin girişi; dosya ismine ve bu dosyanın yer aldığı dosya düğümünün numarasına bağlıdır. Dosya düğümü; dosyalardaki bilgileri depolamak için kullanılan veri bloklarının numaralarını içerir. Dosya düğümü içinde birkaç tane veri bloğu numarası için yer vardır. Bununla beraber dha fazla yere ihtiyaç olursa dinamik bir yapıyla bu yeni yer ayrılır. Bu dinamik yerleştirilmiş bloklar dolaylı bloklardır. Bu veri bloklarını bulmak için önce dolaylı veri bloklarının numaralarını bulmamız gerekir.

Unix dosya sistemleri genellikle `lseek()` sistem çağrısı vasıtası ile bir dosya içerisinde delik oluşturulmasına izin verirler. Burada dosya sistemi, dosya içerisindeki özel bir alanda sanki sıfır byte varmış gibi davranır. Aslında dosya içerisinde bunun için ayrılmış bir disk sektörü yoktur (disk üzerinde daha az yer kaplanmış olur). Bu olay genellikle küçük çalıştırılabilir dosyalar, paylaşılmış Linux kütüphaneleri, bazı veri tabanları ve bazı özel durumlarda sık sık ortaya çıkar. Delikler dolaylı bloklardaki veya dosya düğümü içindeki veri bloklarının adresleri gibi özel değerler verilerek kullanılırlar. Bu özel adresler, dosyanın bahsi geçen bölümü için her hangi bir veri bloğunun ayrılamamış olduğunu, orada bir delik bulunduğunu gösterir.

7.2. Dosya sistemi bolluğu

Linux çok çeşitli dosya sistemlerini desteklemektedir. En önemlileri aşağıda tanıtılmıştır:

minix
: En eski, en güvenli olarak kabul edilen ama kısıtlı yeteneklere ve özelliklere sahip olan dosya sistemidir. (En fazla 64 MB lık dosya sistemi, en çok 30 karakterlik dosya isimleri, ara sıra kaybolan tarih zaman damgaları gibi...

xia
: Dosya isimleri ve dosya sistemi boyutlarının sınırlarını kaldıran ama bundan başka pek bir yenilik getirmeyen, sadece minix dosya sisteminin yenilenmiş halidir. Çok popüler değildir ama oldukça iyi çalıştığı rapor edilmektedir.

ext3

ext3 dosya sistemi, ext2'nin bütün özelliklerine sahip bir dosya sistemidir. Aradaki temel fark, günlükleme özelliğinin eklenmiş olmasıdır. Böylece, herhangi bir sistem çökmesi esnasında, geri kurtarma zamanı kısaltılır ve performans artışı sağlanır. ext3, ext2'den daha popüler olmuştur.

ext2
En yetenekli Linux dosya sistemidir. İleriye dönük kolay geliştirilebilen bir dosya sistemi olarak tasarlanmıştır. Dolayısıyla yeni sürümü, dosya sistemi kodlarını, kurulu bir sisteme uygulamak için yeni ayarlar yapmayı gerektirmez.

ext
Ext2'nin geliştirilmeye uygun olmayan eski sürümüdür. Pek çok insan ext2 dosya sistemine yönelmiştir.

reiserfs
Çok sağlam bir dosya sistemidir. Veri kayıplarını en aza indirmek için günlükleme (journalling) yöntemi kullanılır. Günlükleme; yapılmış veya yapılan işlemlerin kayıtlarının tutulması mekanizmasıdır. Bu sayede dosya sistemi meydana gelmiş olan hasarları son derece kolay bir biçimde onarabilir.

Bunlara ek olarak, çok sayıda yabancı dosya sistemine destek bulunmaktadır. Böylece işletim sistemleri arasında dosyaları değişmek kolaylaştırılmıştır. Bu yabancı dosya sistemleri, makine üzerinde doğal Linux dosya sistemleri gibi çalışabilirler. Ama Unix'in bazı özelliklerinden faydalanamazlar, bazı kısıtlamalara tabidirler veya bazı acayiplikler sergilerler.

msdos
MS–DOS FAT dosya sistemleri (OS/2 ve Windows NT) ile uyumlu bir dosya sistemidir.

umsdos
Msdos dosya sistemi sürücülerine, Linux altında daha uzun dosya isimleri, sahipler, izinler, bağlar ve aygıt dosyaları erişimi sağlar. Bu sistem; normal bir MSDOS dosya sisteminin sanki Linux dosya sistemiymiş gibi kullanılmasını sağlar ve böylece Linux için bağımsız bir bölüm oluşturulması zorunluluğunu ortadan kaldırır.

vfat
FAT32 olarak bilinen dosya sisteminin bir uzantısıdır. Pek çok MS Windows diski vfat'tır. FAT'tan daha büyük disk alanlarını destekler.

iso9660
CD ROM'lar için standart dosya sistemleridir. Daha uzun dosya isimlerine izin veren Rock Ridge uzantısı otomatik olarak desteklenir.

nfs
Bir ağ dosya sistemidir. Dosya sisteminin pek çok bilgisayar tarafından paylaşılmasını sağlar.

smbfs
MS Windows bilgisayarlarla paylaşım sağlayan bir ağ dosya sistemidir. Windows dosya paylaşım protokolleri ile uyumludur.

hpfs
OS/2 dosya sistemi.

sysv
SystemV/386, Coherent ve Xenix dosya sistemleri.

Dosya sistemi seçimi duruma göre değişir. Uyumluluk ve diğer sebepler doğal olmayan dosya sistemlerinin kullanılması mecburiyetini getirebilirler. Şayet özgürce seçebilseydik, en mantıklısı ext3 dosya sistemi olurdu çünkü hem ext2'nin bütün özelliklerine sahiptir, hem de günlükleme yapabilmektedir.

Ayrıca bir de "proc" dosya sistemi vardır. `/proc` dizini altından ulaşılabilen bu dosya sistemi aslında bir dosya sistemi değildir. Proc dosya sistemi bazı çekirdek yapı bilgilerine (süreç listesi gibi) ulaşımı kolaylaştırırlar. Böylece bu veri yapılarının bir dosya sistemi gibi görünmesini sağlar ve dosya sisteminin sağlamış olduğu bütün olanakları kullanıma açar. Örneğin bütün süreçlerin listesini alabilmek için şu komutu kullanabiliriz.

```
$ ls -l /proc
toplam 525256
dr-xr-xr-x    3 root     root            0 Oca 13 04:41 1
dr-xr-xr-x    3 root     root            0 Oca 13 04:41 11
dr-xr-xr-x    3 root     root            0 Oca 13 04:41 111
dr-xr-xr-x    3 root     root            0 Oca 13 04:41 112
dr-xr-xr-x    3 root     root            0 Oca 13 04:41 14
dr-xr-xr-x    3 root     root            0 Oca 13 04:41 1563
dr-xr-xr-x    3 root     root            0 Oca 13 04:41 21
dr-xr-xr-x    3 root     root            0 Oca 13 04:41 3
dr-xr-xr-x    3 root     root            0 Oca 13 04:41 4
dr-xr-xr-x    3 root     root            0 Oca 13 04:41 5
dr-xr-xr-x    3 root     root            0 Oca 13 04:41 6
dr-xr-xr-x    3 root     root            0 Oca 13 04:41 7
dr-xr-xr-x    3 xfs      xfs             0 Oca 13 04:41 747
dr-xr-xr-x    3 apache   apache          0 Oca 13 04:41 766
dr-xr-xr-x    3 root     root            0 Oca 13 04:41 8
dr-xr-xr-x    3 root     root            0 Oca 13 04:41 9
-r--r--r--    1 root     root            0 Oca 13 04:41 apm
dr-xr-xr-x    6 root     root            0 Oca 13 04:41 bus
-r--r--r--    1 root     root            0 Oca 13 04:41 cmdline
-r--r--r--    1 root     root            0 Oca 13 04:41 cpuinfo
-r--r--r--    1 root     root            0 Oca 13 04:41 devices
-r--r--r--    1 root     root            0 Oca 13 04:41 dma
dr-xr-xr-x    3 root     root            0 Oca 13 04:41 driver
-r--r--r--    1 root     root            0 Oca 13 04:41 execdomains
-r--r--r--    1 root     root            0 Oca 13 04:41 fb
-r--r--r--    1 root     root            0 Oca 13 04:41 filesystems
dr-xr-xr-x    3 root     root            0 Oca 13 04:41 fs
dr-xr-xr-x    4 root     root            0 Oca 13 04:41 ide
-r--r--r--    1 root     root            0 Oca 13 04:41 interrupts
-r--r--r--    1 root     root            0 Oca 13 04:41 iomem
-r--r--r--    1 root     root            0 Oca 13 04:41 ioports
dr-xr-xr-x   18 root     root            0 Oca 13 04:41 irq
-rw-r--r--    1 root     root            0 Oca 13 04:41 isapnp
-r--------    1 root     root    536809472 Oca 13 04:41 kcore
-r--------    1 root     root            0 Oca 12 14:31 kmsg
-r--r--r--    1 root     root            0 Oca 13 04:41 ksyms
-r--r--r--    1 root     root            0 Oca 13 04:41 loadavg
-r--r--r--    1 root     root            0 Oca 13 04:41 locks
-r--r--r--    1 root     root            0 Oca 13 04:41 mdstat
-r--r--r--    1 root     root            0 Oca 13 04:41 meminfo
-r--r--r--    1 root     root            0 Oca 13 04:41 misc
-r--r--r--    1 root     root            0 Oca 13 04:41 modules
-r--r--r--    1 root     root            0 Oca 13 04:41 mounts
-rw-r--r--    1 root     root          208 Oca 13 04:41 mtrr
dr-xr-xr-x    3 root     root            0 Oca 13 04:41 net
dr-xr-xr-x    2 root     root            0 Oca 13 04:41 nv
```

```
-r--r--r--    1 root     root            0 Oca 13 04:41 partitions
-r--r--r--    1 root     root            0 Oca 13 04:41 pci
lrwxrwxrwx    1 root     root           64 Oca 12 14:31 self -> 1563
-rw-r--r--    1 root     root            0 Oca 13 04:41 slabinfo
-r--r--r--    1 root     root            0 Oca 13 04:41 stat
-r--r--r--    1 root     root            0 Oca 13 04:41 swaps
dr-xr-xr-x   11 root     root            0 Oca 13 04:41 sys
dr-xr-xr-x    2 root     root            0 Oca 13 04:41 sysvipc
dr-xr-xr-x    4 root     root            0 Oca 13 04:41 tty
-r--r--r--    1 root     root            0 Oca 13 04:41 uptime
-r--r--r--    1 root     root            0 Oca 13 04:41 version
dr-xr-xr-x    3 root     root            0 Oca 13 04:41 video
$
```

(Yukarıdaki örnek kısaltılmıştır. Birkaç tane daha süreç ismi listelenebilir.)

Her ne kadar bir dosya sistemi diye adlandırılsa bile proc dosya sistemi sadece çekirdeğin bir hayalidir. Diskte yer kaplamaz. Proc dosya sisteminin her hangi bir bölümüne bakmak istediğimiz zaman, çekirdek sanki bu bölüm varmış gibi davranır ki aslında böyle bir bölüm yoktur. Öyle ki, disk üzerinde yer kaplamayan çok–megabytelık bir `/proc/kcore` dosyası vardır.

7.3. Hangi dosya sistemi kullanılmalı?

Pek çok değişik dosya sistemi arasında genellikle küçük farklılıklar vardır. Kesinlikle ext3 en popüler dosya sistemidir. Çünkü günlükleme yapabilmektedir. Reiserfs ise diğer popüler bir dosya sistemidir. Bu sistemde de günlükleme yapılabilmektedir. Günlüklemenin aşırı yük getiren yapısından, performanstan, güvenilirlikten, uyumdan ve pek çok diğer sebeplerden dolayı; başka bir dosya sistemi kullanmak daha uygun olabilir. Dosya sistemi seçimi kişilere ve ihtiyaçlarına göre değişiklik gösterir.

Günlükleme yapma yeteneğine sahip dosya sistemleri, aynı zamanda, günlüklü dosya sistemleri diye de adlandırılırlar. Günlüklemeli bir dosya sistemi, sistemde olan bitenin kaydını veya günlüğünü tutar. Bir sistem çökmesi durumunda ya da benim iki yaşındaki oğlumun yapmayı çok sevdiği gibi, bilgisayarın fişinin aniden çıkarılması durumunda; günlükleme sistemi kaydedilmemiş veya zarar görmüş verilerin kurtarılmasını sağlarlar. Böylece, veri kayıpları oldukça aşağıya çekilmiş olur. Bu nedenle, muhtemelen gelecek Linux dağıtımlarında, bu özellik standart hale gelecektir. Bununla birlikte, günlüklemenin, sizde boş bir güven duygusu yaratmasına izin vermeyin. Acil durumlarda kullanabilmek için, verilerinizin yedeğini almayı asla ihmal etmeyin.

7.4. Bir dosya sisteminin oluşturulması

Dosya sistemleri **mkfs** komutu ile oluşturulur. Her dosya sistemi türü için ayrı bir program kullanılır. **mkfs** istenen dosya sistemi türüne göre ayrı bir program çalıştırır. Türü `-t fstype` seçeneği ile seçilir. Genel ve en önemli seçenekler aşağıda belirtilmiştir. Ayrıntılı bilgi için komutun kılavuz sayfasına bakılabilir.

`-t dstürü`
: Dosya sisteminin türü seçilir.

`-c`
: Bozuk bloklar için tarama yapar ve bozuk blok listesini hazırlar.

–l DOSYA
: Daha önceki bozuk blok listesini DOSYA isimli dosyadan okur.

Disket üzerinde ext2 dosya sistemi oluşturmak için aşağıdaki komutları uygulamak gerekir:

```
# fdformat -n /dev/fd0u1440
```

```
Çift-yüzlü, 80 iz, 18 sektör/iz. Toplam 1440 kB.
Biçemlendiriliyor...tamam

# badblocks /dev/fd0u1440 1440 $>$ bad-blocks
# mkfs -t ext2 -l bad-blocks /dev/fd0u1440
mke2fs 1.25 (20-Sep-2001)
Dosya sistemi ismi =
İşl. Sist. türü: Linux
Blok boyu = 1024 (günlük kaydı = 0)
Adımlama boyu = 1024 (günlük kaydı = 0)
184 düğüm, 1440 blok
72 blok (%5.00) süper kullanıcı için ayrıldı
İlk veri bloğu = 1
1 blok grubu
Grup başına 8192 blok ve 8192 sekme
grup başına 184 düğüm

Düğüm tabloları yazılıyor: bitti
Süperblokların ve dosya sisteminin hesap bilgileri yazılıyor: bitti

Bu dosya sistemi her 30 bağlamada bir ya da 180 günde bir, hangisi önce
gerçekleşirse, otomatik olarak denetlenecektir. Bu değerleri değiştirmek
için tune2fs'yi -c veya -i seçeneği ile çalıştırınız.
#
```

İlk önce disket biçemleniyor (`-n` seçeneği bozuk sektör taraması gibi ek faaliyetlerin yapılmasını önler). Daha sonra bozuk sektör taraması, sonuçları `bad-blocks` isimli bir dosyaya yönlendirilecek şekilde **badblocks** ile yapılıyor. En sonunda, **badblocks** komutu ile bulunan bozuk sektör listesi okutularak bir dosya sistemi oluşturuluyor.

mkfs ile `-c` seçeneği ayrı bir dosya ve **badblocks** komutu yerine kullanılabilirdi. Aşağıda buna bir örnek görmekteyiz.

```
# mkfs -t ext2 -c /dev/fd0u1440
mke2fs 1.25 (20-Sep-2001)
Dosya sistemi ismi =
İşl. Sist. türü: Linux
Blok boyu = 1024 (günlük kaydı = 0)
Adımlama boyu = 1024 (günlük kaydı = 0)
184 düğüm, 1440 blok
72 blok (%5.00) süper kullanıcı için ayrıldı
İlk veri bloğu = 1
1 blok grubu
Grup başına 8192 blok ve 8192 sekme
grup başına 184 düğüm

Hatalı bloklar için denetleniyor (salt-oku testi): bitti
Düğüm tabloları yazılıyor: bitti
Süperblokların ve dosya sisteminin hesap bilgileri yazılıyor: bitti

Bu dosya sistemi her 30 bağlamada bir ya da 180 günde bir, hangisi önce
gerçekleşirse, otomatik olarak denetlenecektir. Bu değerleri değiştirmek
için tune2fs'yi -c veya -i seçeneği ile çalıştırınız.
#
```

`-c` seçeneği başka bir dosya ile birlikte **badblocks** komutunu kullanmaktan daha uygun bir kullanım şeklidir. Ama daha sonra bozuk blokları bulmak için **badblocks** komutu yeniden kullanılmak zorundadır. Sabit disk

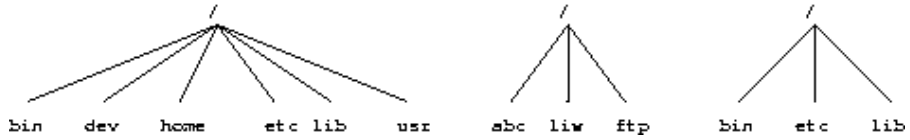
veya bir bölüm üzerinde dosya sistemi hazırlamak disketler için yapılan işlemler ile benzerlik gösterir. Sadece biçemlemek mecburi değildir.

7.5. Dosya sistemlerinin bağlanması ve ayrılması

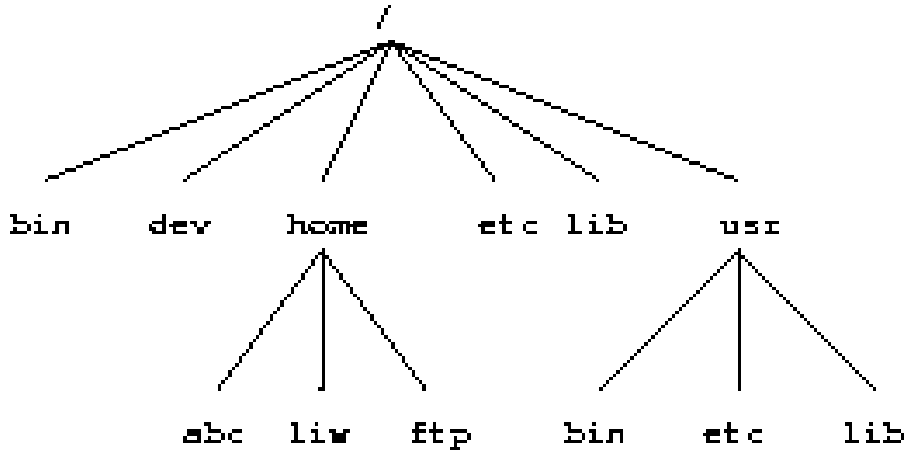
Bir dosya sistemi kullanılmadan önce mutlaka bağlanmalıdır. Daha sonra işletim sistemi her şeyin doğru bir şekilde çalıştığından emin olmak için çeşitli muhasebe işlemleri yapar. Unix altında bütün dosya sistemleri tek bir dizin ağacı altında gözüktüğü için, işletim sistemi yeni bağlanan dosya sistemlerini eskiden bağlanmış olan dosya sistemlerinin alt dizinleriymiş gibi gösterir ve bu şekilde işlem yapar.

Aşağıda *Üç ayrı dosya sistemi* (sayfa: 39)'de kendi kök dizinleri altında bulunan birbirinden bağımsız üç dosya sistemi gösterilmektedir. Son iki dosya sistemi sırayla, birincinin `/home` ve `/usr` dizinleri altına bağlanırsa */home ve /usr bağlı.* (sayfa: 39)'deki gibi tek bir dizin ağacı elde ederiz.

Şekil 5. Üç ayrı dosya sistemi



Şekil 6. `/home` ve `/usr` bağlı.



Aşağıdaki örnekte gösterildiği şekilde bu bağlama işlemleri yapılabilir.

```
# mount /dev/hda2 /home
# mount /dev/hda3 /usr
#
```

mount komutu iki argüman alır. Birincisi diske uygun bir aygıt dosyası veya dosya sistemini içeren bir disk bölümüdür. İkincisi ise bağlandığı dizindir. Bu işlemler yapıldıktan sonra bu iki aygıtın içerikleri sanki disk üzerindeki `/home` ve `/usr` dizinlerinin içerikleriymiş gibi kullanılabilirler. Şayet bağlama işlemi başka bir dizin altına yapılmış olsaydı, aygıtların içeriklerini görebilmek için o dizinlerin altına bakmamız gerekecekti. Ayrıca aygıt dosyası ve bağlanılan dizin arasındaki farkı mutlaka göz önünde bulundurmalıyız. Unutmayın ki aygıt dosyası, (`/dev/hda2`) diskin ham içeriğini verir. Bağlandığı dizin ise (`/home`) disk üzerindeki dosyalara ulaşım imkanı verir. Disk bölümünün bağlandığı dizine bağlama noktası adı verilir.

Linux pek çok dosya sistemini destekler. **mount** dosya sisteminin türünü tahmin etmeye çalışır. İsterseniz `-t fstype` seçeneğini dosya sisteminin türünü belirtmek için kullanabiliriz. Bazı zamanlar bu gerekli olmaktadır,

mount komutu dosya sistemini tanımlayamayabilir. Örneğin bir MS DOS disketini bağlamak için şu komut kullanılır:

```
# mount -t msdos /dev/fd0 /floppy
#
```

Bağlantı dizini mutlaka sistemde var olmalı ve içi boş olmalıdır. İçerisinde her hangi bir dosya bulunursa bağlama işlemi sonucunda ulaşılamaz hale gelir (hali hazırda açılmış dosyalar ve başka bir dizinden sabit bağ verilmiş olan dosyalar hala ulaşılabilir durumdadırlar). Örneğin; bazı kişiler `/tmp` ve `/var/tmp` dizinlerini eş anlamlı olarak kullanırlar ve `/tmp` dizinini `/var/tmp` dizinine sembolik bağ olarak atarlar. Sistem açılırken, `/tmp` dosya sistemi bağlanmadan önce kök dosya sisteminde bulunan `/var/tmp` dizini onun yerine kullanılır. `/var` dizini bağlandığı zaman, bu kök dosya sistemi altındaki `/var/tmp` dizinini ulaşılmaz kılar. Şayet `/var/tmp` dizini kök dosya sistemde yok ise, `/var` dizinini bağlamadan geçici dosyaları kullanmak imkansız olacaktır.

Dosya sistemine her hangi bir şey yazmaya niyetiniz yok ise bağlama işlemi sırasında bu bağlantının salt okunur olmasını sağlayacak `-r` seçeneğini kullanabilirsiniz. Bu çekirdeğin her hangi bir yazım girişiminde bulunmasını ve dosya düğümleri içindeki erişim zamanlarının güncellenmesini engeller. Üzerine yazılamayan ortamları (örn; CDROM) bağlarken bu seçenek gereklidir.

Uyanık okuyucular küçük bir mantıksal problemi fark etmişlerdir. Birinci dosya sistemi (burada kök dosya sistemi oluyor) nasıl bağlanmıştır? Kök dosya sistemi açılış esnasında sihirli bir şekilde bağlanmıştır ve kullanıcı bunun her açılışta tekrar olacağına güvenmelidir. Şayet kök dosya sistemi bağlanmazsa açılış yapılamaz. Sihirli bir biçimde kök dosya sistemini bağlayan dosya sisteminin adı çekirdek içinde derlenmiştir veya LILO ya da **rdev** kullanılarak ayarlanmıştır.

Kök dosya sistemi ilk olarak salt okunabilir şekilde bağlanır. Daha sonra başlayan betikler onun geçerliliğini kanıtlamak için **fsck** programını çalıştırırlar. Şayet bir problem çıkmaz ise oku–yaz olarak tekrar bağlanır. Böylece yazma işlemlerine olanak sağlanır. **fsck**, oku–yaz bağlı bir dosya sistemi üzerinde yapılmamalıdır. Çünkü dosya sisteminde meydana gelen her değişiklik **fsck** programını etkileyecek ve çalışması esnasında sorunlar ortaya çıkmasına sebep olacaktır. Kök dosya sistemi, denetlenirken salt okunur şekilde bağlanmış olduğu için **fsck** herhangi bir problemi sıkıntıya girmeden giderebilir. Dosya sisteminin bellekte tuttuğu bütün bilgiler yeniden bağlama işlemi esnasında silinecektir.

Pek çok sistemde açılış esnasında otomatik olarak bağlanması gereken dosya sistemleri mevcuttur. Bunlar `/etc/fstab` dosyası içinde belirtilmişlerdir. Ayrıntılı bilgi **fstab** kılavuz sayfasında mevcuttur. Açılış esnasında bağlanılacak dosya sistemlerinin özellikleri çok çeşitli faktörlere dayanır ve bu sistem yöneticisinin şahsi ihtiyaç ve görüşlerine göre değişiklik gösterebilir. Daha ayrıntılı bilgi *Açılışlar ve Kapanışlar* (sayfa: 52) bölümünde mevcuttur.

Bir dosya sisteminin daha fazla bağlı durumda kalması gerekmiyorsa **umount** komutu ile dosya sistemi sistemden ayrılmalıdır.[6] **umount** argüman olarak aygıt dosyası ismini veya bağlama noktası dizininin adını kullanır. Bu işlem aşağıdaki şekillerden birisi ile gerçekleştirilir:

```
# umount /dev/hda2
# umount /usr
#
```

Komutun değişik kullanımlar için kılavuz sayfalarına bakabilirsiniz. Bağlı bir disketin ayrılması zorunludur. Sakın disketi doğrudan dışarı çıkarmayın. Disk önbellekleme olayı yüzünden, gerekli bilgiler siz disketi **umount** komutu ile sistemden ayırana kadar diskete yazılmaz. Erken çıkarma sonucu verilerde hasar veya karışıklık meydana gelebilir. Disketten sadece okuyorsanız bu bir problem yaratmaz, fakat yazım işlemleri de yaptıysanız sonuç bir yıkım olabilir.

Sadece root kullanıcı dosya istemlerini bağlayabilir ve ayırabilir. Şayet her kullanıcı istediği bir dizine bir disket bağlarsa, sonuçta `/bin/sh` dizini gibi görünen bir truva atı yaratmak çok kolay hale gelirdi. Bununla beraber kullanıcılara disket bağlama izni vermek kaçınılmazdır. Bunun çeşitli yolları vardır:

- Kullanıcılara root parolasını vermek. En kolay fakat en az güvenli seçenektir. Ağa bağlı olmayan kişisel sistemler için en uygun çözümdür.
- Kullanıcıların dosya sistemi bağlama işlemlerini yapabilmesine izin vermek için **sudo** gibi bir program kullanabilirsiniz. Bu hala az güvenli bir seçenektir, sadece süper kullanıcı yetkileri herkese doğrudan verilmemiş olur.[7]
- Kullanıcıları **mtools** kullanmaya sevk edin. Böylece MS–DOS dosya sistemlerini bağlamaya gerek kalmadan kullanabilirler.
- `/etc/fstab` dosyası içinde bağlanması olası dizin ve aygıt dosyası adlarını uygun seçeneklerle tanımlayın.

`/etc/fstab` dosyası içine şu şekilde bir satır eklemek son bir alternatif olabilir:

```
/dev/fd0            /floppy      msdos    user,noauto      0 0
```

Birinci sütundan itibaren, bağlanacak aygıt dosyası, bağlantı dizini, dosya sistemi, seçenekler, yedekleme yapma sıklığını (**dump** tarafından kullanılır) ve açılış esnasında hangi dosya sistemlerinin **fsck** tarafından denetleneceğini belirten numaralar (sıfır= birşey yapma) belirtilmiştir.

`noauto` seçeneği açılış esnasında dosya sisteminin otomatik olarak bağlanmayacağını belirtir. `user` seçeneği ise bütün kullanıcılara bu dosya sistemini bağlama izni verirken güvenlik sebeplerinden dolayı, bağlanmış dizinden aygıt dosyalarının yorumlanmasına ve normal programların dahi çalıştırılmasına izin vermez. Artık her hangi bir kullanıcı msdos dosya sistemine sahip bir disketi bağlamak için aşağıdaki komutu kullanabilir:

```
$ mount /floppy
$
```

Bu disket **umount** komutu ile daha sonra sistemden ayrılmalıdır.

Şayet çeşitli disketlere ulaşım verilmesini istiyorsanız, farklı bağlama noktaları tanımlamalısınız. Msdos ve ext2 dosya sistemlerine sahip disketlere ulaşabilmek için `/etc/fstab` dosyasına şu satırlar eklenmelidir:

```
/dev/fd0    /dosfloppy    msdos    user,noauto  0  0
/dev/fd0    /ext2floppy   ext2     user,noauto  0  0
```

MS–DOS dosya sistemleri için, genellikle `uid`, `gid` ve `umask` dosya sistemi seçeneklerini kullanarak erişime sınırlamalar getirmek isteyebilirsiniz. Şayet dikkatli davranmazsanız, bağlı bir MS_DOS dosya sistemi için herkese en azından okuma izni vermiş olursunuz. Bu da pek iyi bir fikir olmayabilir.

7.6. fsck ile dosya sistemi bütünlüğünün sınanması

Dosya sistemleri zaman zaman hata vermeye eğilimli olan karmaşık yapıdaki yaratıklardır. Bir dosya sisteminin doğruluğu ve geçerliliği **fsck** komutu ile sınanır. Bulduğu küçük hataları onarmak ve onaramadığı daha önemli hatalar için kullanıcıyı uyarmak üzere programlanabilirler. Şans eseri dosya sistemlerinin kodlarını onarma işleminde oldukça etkilidirler. Bununla birlikte çok nadir olarak kullanıcı, donanım veya elektrik kesintilerinden kaynaklanan hatalar meydana gelir.

Pek çok sistemde **fsck** komutu açılışta çalışmak üzere ayarlanır. Ve bu sayede dosya sisteminde meydana gelmiş olan hataların sistem kullanıma başlanmadan önce düzeltilmesi umulur. Bozulmuş dosya sistemleri işleri yanlış yönlendirirler: veri sistemleri karıştıysa, dosya sistemi büyük olasılıkla onları daha fazla karıştıracaktır. Bununla birlikte büyük dosya sistemlerinde **fsck** komutunun çalışması biraz vakit alabilir ama sistem düzgün kapatılmış ise dosya sisteminde hata meydana gelme olasılığı hemen hemen hiç yoktur. Böyle durumlarda sınamanın yapılmasını önlemek için bazı hileler vardır. Birincisi: şayet `/etc/fastboot` dosyası varsa sınama işlemi yapılmaz. İkincisi ise: superbloklar içinde sistemin bir önceki kapanışta düzgün kapatılıp kapatılmadığını

belirten işaretlerdir. Şayet sistem düzgün bir şekilde ayrılmışsa **e2fsck** (ext2 dosya sistemi için **fsck** programı) komutu işleme girmez. /etc/fastboot hilesi sizin açılış esnasında kullandığınız betiklere göre çalışır ama ext2 hilesi **e2fsck** kullandığınız sürece işe yarar. **e2fsck**'den kurtulmak için açık bir şekilde seçenekler belirtilmiş olmalıdır. Ayrıntılar için **e2fsck** kılavuz sayfasına bakınız.

Otomatik sınama sadece açılış esnasında otomatik bağlanan dosya sistemleri için geçerlidir. Disket ve benzeri aygıtlar için **fsck** komutunu kendiniz kullanmalısınız.

Şayet **fsck** tamir edemeyeceği problemler ile karşılaşırsa; iyi yedekleme, dosya sistemlerinin kullanımı ve bozulmuş dosya sistemlerinin türleri hakkında ayrıntılı ve derin bilgiye sahip olmanız gerekecektir. Daha sonrası kolaydır, genellikle de sıkıcı. Kendi kendinize yetemeyeceğiniz durumda bir arkadaşınızdan, posta listelerinden, haber gruplarından veya bunlara benzer bir yerlerden yardım alınabilir. Size daha fazlasını anlatmak isterdim fakat bu konuda ki eğitimimim ve deneyimlerim eksikliği buna engel olmaktadır. Theodore Ts'o'nun **debugfs**[8] programı bu konuda yardımcı olabilir.

fsck mutlaka bağlı olmayan dosya sistemleri üzerinde yapılmalıdır. Sadece açılış esnasında salt okunur konumdaki kök dosya sistemi bu durum için istisnadır. Bunun sebebi **fsck** komutunun disk yüzeyine doğrudan erişerek dosya sistemi üzerinde yaptığı bazı değişikliklerin, işletim sistemi tarafından anlaşılamama olasılığı bulunması ve bunun da işletim sistemi üzerinde problemler yaratabilecek olmasıdır.

7.7. Disk hatalarının badblocks ile denetlenmesi

Bozuk bloklar için periyodik denetimler yapmak iyi bir fikir olabilir. Bu işlem **badblocks** komutu ile yapılabilir. Bu komut bulabildiği bütün bozuk bloklar için bir liste verir. Bu liste **fsck** için dosya sistemi veri yapısı içinde bulunan kayıtlara yönlendirilebilir, böylece işletim sisteminin bozuk blok hatalarının kaydı tutulmak zorunda kalınmaz. Aşağıdaki örnek bunun nasıl yapılacağını açıklamaktadır.

```
# badblocks -o bad-blocks /dev/fd0u1440
# fsck -t ext2 -l bad-blocks /dev/fd0u1440
fsck 1.25 (20-Sep-2001)
e2fsck 1.25 (20-Sep-2001)
1. geçiş: dosya indeksleri, bloklar ve uzunluklar denetleniyor
2. geçiş: Dizin yapısı denetleniyor
3. geçiş: Dizin bağlanabilirliği denetleniyor
4. geçiş: Başvuru sayısı denetleniyor
5. geçiş: grup özet bilgileri denetleniyor

/dev/fd0u1440: ***** DOSYA SİSTEMİ DEĞİŞTİRİLDİ *****
/dev/fd0u1440: 11/184 dosya (0.%0 yanyana olmayan düğüm), 41/1440 blok
#
```

Şayet kullanımdaki bir bloğun hatalı olduğu rapor edilirse **e2fsck** bu bloğu başka bir yere taşıyacaktır. Şayet durum gerçekten kötü ise bu blok içindeki dosyaların içeriği bile bozulabilir.

7.8. Dosya sistemi üzerindeki parçalanmalarla savaşmak

Bir dosya diske yazılırken, yazma işlemi her zaman ardışık bloklar halinde gerçekleşmez. Ardışık bloklar halinde depolanmamış dosyalara parçalanmış dosyalar (fragmented files) denir. Bu tür dosyaların okunması daha uzun sürer, çünkü okuyucu kafa disk üzerinde daha fazla hareket etmek zorunda kalır. Bu durumdan kaçınabilmek arzu edilen bir durumdur, ama bununla birlikte disk üzerinde iyi bir tampon belleğe sahipseniz bu durum pek problem yaratmaz.

Ext2 dosya sistemi dosya parçalanması olayını minimumda tutabilmek için, dosyalar ardışık bloklara yazılamasa bile, mümkün olan en yakın bloklar içine yazma yapar. Ext2 etkin bir şekilde, boş blokları bir dosya içindeki

bloklara en yakın yere yerleştirir. Ext2 için parçalanma problemi pek nadir olarak ortaya çıkan bir şeydir. Ext2 dosya sistemi için **defrag**[B27] isimli bir birleştirme programı vardır ve bu program oldukça etkilidir.

MS–DOS için de, blokları dosya sistemi içinde taşıyan, pek çok dosya birleştirme programı vardır. Diğer dosya sistemleri için ise izlenecek yol şöyledir: önce dosya sisteminin bir yedeği alınır, sonra dosya sistemi yeniden yaratılır ve yedekler geriye yüklenir. Dosya birleştirme öncesi sistemin yedeğini almak bütün dosya sistemleri için en iyi yöntemdir; Dosya birleştirme işlemi sırasında pek çok hata meydana gelebilir ve yahut yanlışlıklar olabilir.

7.9. Diğer dosya sistemi araçları

Dosya sistemlerini yönetmek için başka araçlar da vardır. **df** komutu seçeneksiz kullanıldığında bağlı tüm dosya sistemlerinin dolu ve boş alanları hakkında bilgi verir. **du** komutu bir dizinin ve içerdiği dosyaların ne kadar disk alanı kullandığını gösterir. Bu araçlar disk alanının kullanımı hakkında bilgi vermesi bakımından olası sorunların yakalanmasında yardımcı olabilir. Ayrıntılı bilgi için her ikisinin de kılavuz sayfalarına bakabilirsiniz.

sync komutu diske henüz yazılmamış ve tampon bellekte tutumakta olan bilgilerin diske yazılmasını sağlar. (*Tampon bellek* (sayfa: 51) bölümüne bakınız.) Bunu yapmak nadiren gerekli olur. Çünkü bir artalan süreci olan **update** bu işlemi otomatik olarak yapar. Genellikle yıkım anlarında faydalı olur; örneğin **update** ya da onun yardımcı süreci **bdflush** ölmüşse veya *now* seçeneği ile sistemi kapatmak üzereyken **update**'in devreye girmesini bekleyemeyecekseniz... Bu konuda daha ayrıntılı bilgileri kılavuz sayfalarından bulabileceğinizi tekrarlıyorum. **man** komutu Linux'da en iyi arkadaşınızdır. Kuzeni olan **apropos** komutu da kullanmak istediğiniz komutun ya da konunun adını hatırlayamadığınızda ve bir anahtar sözcük verdiğinizde ilgili kılavuz sayfalarının bir listesini göstererek size yardımcı olacaktır.

7.10. Diğer ext2 dosya sistemi araçları

Dosya sistemi yaratıcısı **mke2fs** ve denetleyicisi **e2fsck**'ye ek olarak ext2 dosya sistemi, dosya sisteminden bağımsız olarak ulaşılabilen ve yahut doğrudan kullanılabilen, bazı yararlı ek araçlara da sahiptir.

tune2fs dosya sistemi parametrelerini ayarlar. Bazı ilginç parametreler şunlardır:

- Maksimum bağlama sayısı. Dosya sistemleri belirli bir bağlama sayısına ulaşılınca, sistem temiz işaretine rağmen, **e2fsck** dosya sistemini kontrol etmeye kalkışır. Geliştirme veya sınama amacıyla kullanılan bir sistemde bu sayıyı düşürmek iyi bir fikir olabilir.
- Denetimler arasındaki maksimum zaman. Sistem temiz işaretine ve denetim için gerekli bağlama sayısına ulaşılmamış olsa bile; en son denetimden beri geçen süre belli bir değere ulaşınca **e2fsck** sistemi denetlemek isteyebilir. Bu seçenek istenirse kapatılabilir.
- root kullanıcı için ayrılmış blok sayısı. Ext2, dosya sistemi dolmuş olsa bile, root kullanıcının her hangi bir şeyi silmeden işlerine devam edebilmesi için bazı bloklar ayrılmıştır. Bunun için ayrılan blokların oranı öntanımlı %5'dir. Bu da pek çok disk için fazla sayılmayacak bir orandır.

Daha fazla bilgi edinmek için **tune2fs** kılavuz sayfasına bakınız.

dumpe2fs ext2 dosya sistemi hakkında, genellikle süper bloktan alınan, bilgileri gösterir. Bu bilgilerden bazıları teknik ayrıntılardır ve dosya sisteminin nasıl çalıştığının bilinmesini gerektirir, fakat pek çoğu kolayca anlaşılabilir açıklamalardır. Aşağıda böyle bir çıktı vardır:

```
# dumpe2fs -h /dev/hdb2
dumpe2fs 1.25 (20-Sep-2001)
Filesystem volume name:   /b2
Last mounted on:          <not available>
Filesystem UUID:          ad9437d2-0e25-490b-8c9d-9a0da50c95df
```

```
Filesystem magic number:  0xEF53
Filesystem revision #:    1 (dynamic)
Filesystem features:      has_journal filetype needs_recovery sparse_super
Filesystem state:         clean
Errors behavior:          Continue
Filesystem OS type:       Linux
Inode count:              366592
Block count:              2931862
Reserved block count:     146593
Free blocks:              780635
Free inodes:              306325
First block:              1
Block size:               1024
Fragment size:            1024
Blocks per group:         8192
Fragments per group:      8192
Inodes per group:         1024
Inode blocks per group:   128
Last mount time:          Mon Jan 13 16:47:05 2003
Last write time:          Mon Jan 13 16:47:05 2003
Mount count:              821
Maximum mount count:      -1
Last checked:             Thu Oct 25 16:02:44 2001
Check interval:           0 (<none>)
Reserved blocks uid:      0 (user root)
Reserved blocks gid:      0 (group root)
First inode:              11
Inode size:               128
Journal UUID:             <none>
Journal inode:            8
Journal device:           0x0000
First orphan inode:       0


Grup 0: (1 ile 8192 arasındaki bloklar)
    İlk süperbloğun yeri:  1
  Grup tanımının yeri:     2-13
  Blok biteşleminin yeri:  14 (+13)
  Düğüm biteşleminin yeri: 15 (+14)
  Düğüm tablosunun yeri:   16-143 (+15)
  Serbest blok sayısı:     0
  Serbest düğüm sayısı:    792
  Dizin sayısı:            19
  Serbest bloklar:
  Serbest düğümler:        233-1024
Grup 1: (8193 ile 16384 arasındaki bloklar)
  Yedek süperbloğun yeri:  8193
  Grup tanımının yeri:     8194-8205
  Blok biteşleminin yeri:  8206 (+13)
  Düğüm biteşleminin yeri: 8207 (+14)
  Düğüm tablosunun yeri:   8208-8335 (+15)
  Serbest blok sayısı:     0
  Serbest düğüm sayısı:    855
  Dizin sayısı:            31
  Serbest bloklar:
  Serbest düğümler:        1194-2048
```

```
[çıktının Grup 1 ile Grup 357 arasında grup bilgileri silindi]

Grup 357: (2924545 ile 2931861 arasındaki bloklar)
  Blok biteşleminin yeri:  2924545 (+0)
  Düğüm biteşleminin yeri: 2924546 (+1)
  Düğüm tablosunun yeri:   2924560-2924687 (+15)
  Serbest blok sayısı:     5452
  Serbest düğüm sayısı:    812
  Dizin sayısı:            25
  Serbest bloklar:         2924699, 2925104-2925488, 2925495-2925498, 2925525, \
                           2926801-2931861
  Serbest düğümler:        365580, 365607, 365783-366592
#
```

debugfs dosya sistemleri için kullanılan bir hata ayıklama programıdır. Disk üzerinde bulunan dosya sistemi veri yapısına doğrudan erişim sağlar ve **fsck** komutunun onaramadığı hataları düzeltmeye yarar. Ayrıca silinmiş dosyaları geri almak için kullanıldığı da bilinmektedir. Bununla birlikte **debugfs** ne yaptığınızı bilmeniz gereken bir programdır, aksi halde bütün bilgilerinizi yok edebilirsiniz.

dump ve **restore** ext2 dosya sisteminin yedekleme işlemlerinde kullanılırlar. Geleneksel Unix işletim sistemlerinin, ext2 dosya sistemi için geliştirilmiş özel sürümleridir. *Yedek Alma* (sayfa: 67) bölümünde yedekleme hakkında daha ayrıntılı bilgi bulabilirsiniz.

7.11. Dosya sistemleri olmayan diskler

Bütün bölümler ve diskler dosya sistemlerini kullanmazlar. Örneğin takas bölümü bir dosya sistemine sahip değildir. Pek çok disket teyp sürücü tarzında kullanılırlar. Bu nedenle **tar** veya disket yüzeyine doğrudan yazan programlar dosya sistemlerini kullanmazlar. Linux açılış disketleri bir dosya sistemi içermezler, sadece temel çekirdeğe sahiptirler.

Dosya sistemleri disklerin daha kullanışlı olmalarını sağlamakla birlikte, daima aşırı kayıtlama olayından dolayı, bu özelliklerini kısıtlamaktadırlar. Diskleri diğer sistemler ile daha uyumlu hale getirirler. Örneğin dosya sistemleri pek çok sistemde farklı olduğu durumlarda bile, tar dosya şekli bütün sistemlerde aynıdır. Linux açılış disketleri dosya sistemlerine ihtiyaç duymuyor olmalarına rağmen bir dosya sistemine sahip olabilirler.

İşlenmemiş diskleri kullanmamızın bir sebebi de onların biteşlem kopyalarını yaratmaktır. Eğer bir disk kısmen zarar görmüş bir dosya sistemi içeriyorsa, onu tamir etmeye çalışmadan önce tam bir kopyasını çıkarmak, meydana gelebilecek sorunlara karşı iyi bir önlem olabilir. Böylece tamir işlememiz başarısız olsa bile, bu işlemi tekrar yapabilme şansımız olur. Bunu yapmanın bir yolu **dd** komutunu kullanmaktır:

```
$ dd if=/dev/fd0u1440 of=floppy-image
2880+0 records in
2880+0 records out
$ dd if=floppy-image of=/dev/fd0u1440
2880+0 records in
2880+0 records out
$
```

İlk **dd** `floppy-image` dosyasına disketin tam bir kopyasını yaratır. İkincisi ise `floppy-image`'i diskete yazar (burada kullanıcının ikinci komuttan önce disketi değiştirmiş olduğu varsayılmaktadır. Zaten aksi bir durumda bu komut çiftinin hiçbir anlamı kalmayacaktır).

8. Disk Alanının Ayarlanması

8.1. Disk bölümleme şemaları

Bir diski bölümlere ayırmanın hem en iyi hem de en kolay bir yolu yoktur. Dah kötüsü bunu yapmanın genelleşmiş bir yolu da yoktur. Çünkü bölümleme kararınızı etkileyecek pek çok faktör vardır.

Geleneksel yöntem; küçük bir kök dosya sistemi yaratmak ve bunun içinde `/bin`, `/etc`, `/dev`, `/lib`, `/tmp`, gibi sistemin çalıştırılması ve yönetilmesi için ihtiyaç duyulan dosya sistemlerinin yerleştirilmiş olmasıdır. Bu yolla, kök dosya sistemi sistemin açılması ve çalışması için gerekli her şeyi ihtiva etmiş olur. Kök dosya sisteminin küçük tutulmasındaki ve daha az iş yüküne sahip olmasındaki ana sebep, sistem çökmeleri esnasında kök dosya sisteminde daha az hasar meydana gelmesini ve kolayca tamir edilebilmesini sağlamaktır. Daha sonra başka bölümler ya da başka diskler kullanarak dizin ağacına `/usr` dosya sistemini, kullanıcılar için ayrı bir `/home` dizini ve bağımsız bir takas bölümü oluşturabilirsiniz. Ev dizinlerini bağımsız bölümler içinde yerleştirmek yedekleme işleminde büyük kolaylıklar sağlar. Bir ağ ortamında `/usr` dizinini çeşitli makineler arasında paylaştırmak da mümkündür. Böylece her makinenin ihtiyaç duyduğu yüzlerce megabytelık alanlara olan ihtiyaç azaltılmış olur.

Pek çok disk bölümüne sahip olmak, boş disk yüzeyinin parçalar halinde ayrılmış olması problemini yanında getirir. Günümüzde, işletim sistemlerinin ve disklerin daha güvenli olması ile, kullanıcılar bütün dizinleri kapsayan tek bir dizine doğru yönelmektedir. Bununla birlikte küçük bir ikinci bölüm yedekleme işleminin daha kolay yapılmasını sağlayabilir.

Küçük bir sabit diske sahipseniz (çekirdek geliştirmesi yapmayacağınız varsayılarak) tek bir bölüme sahip olmanız daha uygun olacaktır. Büyük bir diske sahipseniz, herhangi bir şeyin ters gidebileceği ihtimalini göz önüne alarak, birkaç adet büyük disk bölümüne sahip olmanız daha mantıklı olacaktır. (Buradaki büyük ve küçük kelimeleri tamamen öznel kavramlardır. Disk bölümlerinin büyüklüğünü tamamen sizin kullanım amaç ve gereksinimlerinize göre ayarlamalısınız.)

Şayet birkaç tane sabit diske sahipseniz; birine `/usr` dizinini de içeren bir kök dizini, diğerinde ise `/home` dizini yarleştirebilirsiniz.

Çeşitli bölümleme tabloları için değişik deneyler yapmanız iyi olabilir. Bu biraz iş yükü getirmektedir. Çünkü her seferinde çeşitli kereler işletim sistemini yeniden kurmanız gerekecektir. Ama bu da yaptığınız işin doğruluğundan emin olmanın tek yoludur.

8.2. Alan gereksinimleri

Kuracağınız Linux dağıtımı, size değişik yapılandırmalara göre ne kadarlık bir disk alanına ihtiyacınız olduğunu bildirir. Bağımsız olarak kurulan programlar da bunu yaparlar. Bu sayede disk alanının kullanımı için planlarınızı yapabilirsiniz, ama ilerisi için hazırlıklı olmalı ve ilerde duyabileceğiniz ihtiyaçları göz önünde bulundurarak fazladan disk alanı ayırmalısınız.

Kullanıcı dosyaları için ayıracağınız bölümün hacmi, kullanıcıların ne yapmak istediklerine göre değişir. Pek çok insanın, dosyaları için daha fazla alana ihtiyaçları varmış gibi görünmektedir, fakat ihtiyaç duyulan alan kişiden kişiye değişiklik gösterir. Bazı kişiler hafif yazım işleri ile uğraşırlar ve onlar için birkaç gigabyte alan yeterli olabilir. Bazıları ise onlarca gigabyte gerektirecek grafik işleri ile uğraşırlar.

Sırası gelmişken, kB veya MB cinsinden belirtilen dosya boyutları ile, MB cinsinden belirtilen disk boyutunu karşılaştırırken, her iki birimin farklılıklar içerebileceğini unutmayın. Bazı üreticiler bir kilobyte'ı 1000 byte, bir megabyte'ı 1000 kilobyte olarak ayarlamışken, bazıları 1024 ve katlarını birimler olarak kullanırlar. Bu nedenle benim 345 MB'lık hard diskim aslında 330Mb'dır.

Takas alanının ayrılması *Takas alanının ayrılması* (sayfa: 50) bölümünde anlatılmıştır.

8.3. Sabit disk bölümleme örnekleri

Daha önce 109 Mb lık bir sabit diskim vardı. Şu an ise 330 MB lik bir disk kullanıyorum. Bu diskleri nasıl ve niçin bölümlediğimi anlatacağım.

Kullandığım işletim sistemleri ve ihtiyaçlarım değiştiği için pek çok defalar 109 Mb lık sabit diski bölümledim. İlk önceleri Linux ile birlikte MS–DOS kullandım. Bunun için 20 MB lık bir diske ihtiyacım vardı veya klostrofobiye kapılmadan MS–DOS, bir C derleyici, çeşitli yararlı araçlar, bir metin düzenleyici ve üzerinde çalıştığım program için ancak yeten bir yere sahip olduğumu söylemeliyim. Linux takas alanı için 10 Mb ayırmıştım, geri kalan 79 MB ise Linux için ayrılmış tek bir dizin idi. Kök, `/usr`, ve `/home` dizinlerini bağımsız bir şekilde yerleştirmek için çeşitli teşebbüslerde bulundum fakat bu olayı ilginç ve gerekli kılacak kadar disk alanım maalesef yoktu.

MS DOS'a daha fazla ihtiyacım kalmadığı zaman diski 12 MB takas alanı ve geri kalanını tek bir dizin halinde Linux olarak tekrar biçimlendirdim.

330 MB'lik disk ise şu şekilde çeşitli bölümlere ayrıldı.

5 MB	kök dosya sistemi
10 MB	takas alanı
180 MB	`/usr` dosya sistemi
120 MB	`/home` dosya sistemi
15 MB	karalama bölümü

Karalama bölümü üzerinde oynanmak için kendilerine ayrı bir bölüm isteyen şeyler içindir. Örneğin çeşitli dosya sistemlerinin karşılaştırılması veya çeşitli dosya sistemlerinin hızlarının karşılaştırılması gibi... başka bir şeye ihtiyaç kalmayınca bu bölümü takas alanı olarak kullanırdım. (günümüzde karalama bölümü tarih olmuş sayılabilir. Artık pek çok kullanıcın gigabytelar ile ölçülen disk alanları var.)

8.4. Linux için daha fazla disk alanı eklemek

Donanımınız doğru bir şekilde yüklenmiş ise Linux'e fazladan yer vermek çok kolay bir iştir. Yukarıda tanımlandığı şekilde gerekiyorsa diski biçemleyin, disk bölümlerini ayırın ve dosya sistemlerini kurun. `/etc/fstab` dosyasına otomatik bağlama işlemi için gerekli satırları ekleyin.

8.5. Disk alanını kazanmak için ipuçları

En iyi tavsiye gereksiz programları kurmamanızı söylemek olacaktır. Pek çok Linux dağıtımı içerdikleri paketlerin belirli bölümlerini kurmanız için gereken kurulum seçeneklerine sahiptirler. Dikkatle seçim yaparsanız pek çok programa ihtiyacınız olmadığını görebilirsiniz. Bazı programlar çok büyük olduğu için bu yolla geniş bir disk alanını kazanabilirsiniz. Bir paket ya da programın bir bölümüne ihtiyacınız olsa bile onun hepsine ihtiyacınız olmayabilir. Örneğin: bazı belgeler gereksiz olabilir, GNU Emacs için bazı Elisp dosyaları gereksiz olabilir, X11 için bazı yazıtipleri gereksiz olabilir veya programlama kütüphaneleri gereksiz olabilir.

Paketleri kaldıramıyorsanız, en azından sıkıştırın. **`gzip`** veya **`zip`** gibi sıkıştırma programları kişisel veya grup dosyalarınızı sıkıştıracaklardır. **`gzexe`** programı sıkıştırma ve açma işlemlerini kullanıcıya hissettirmeden yapar (kullanılmayan dosyaları sıkıştırır ve ihtiyaç olunca geri açar). Daha gelişmiş DouBle sistemi bir dosya sistemindeki bütün dosyaları onları kullanan programlara hissettirmeden sıkıştırır (windows'taki drivespace veya msdos'taki stacker gibi programlar hakkında bilgi sahibiyseniz, bu programın mantığını daha iyi anlayabilirsiniz. Temel çalışma mantıkları benzerlik gösterir).

9. Bellek Yönetimi

"Minnet, jag har tappat mitt minne, är jag svensk eller finne, kommer inte ihåg..." (Bosse Österberg)

> A Swedish drinking song, (rough) translation: “Memory, I have lost my memory. Am I Swedish or Finnish? I can’t remember”

Bu bölümde Linux bellek yönetiminin özellikleri anlatılmıştır. Sanal bellek ve disk tampon belleği gibi şeylerin çalışma prensipleri, amaçları ve sistem yöneticisinin dikkat etmesi gereken durumlar anlatılmaktadır.

9.1. Sanal bellek nedir?

Linux, disk yüzeyini RAM belleğin bir uzantısıymış gibi kullanan ve böylelikle fiziksel belleğin görünürdeki miktarını arttıran sanal bellek desteğine sahiptir. Çekirdek bellekteki kullanılmayan bloklarda bulunan bilgileri, disk yüzeyine yazar ve bellek başka işler için serbest kalmış olur. Bu bölümler gerektiği zaman bunlar belleğe tekrar alınırlar. Bu olaylar kullanıcıdan bağımsız bir şekilde gerçekleşir. Linux altında çalışan programlar geniş bir bellek alanı görürken, aslında bazı bilgilerin hemen yanı başlarında olduğundan habersizlerdir. Elbette ki sanal bellek kullanımı RAM kullanımı kadar hızlı değildir, bu nedenle program hızlarında düşüş yaşanır. Sabit disk üzerinde sanal bellek için kullanılan bölüme *takas alanı* denir.

Linux, dosya sistemi içindeki normal bir dosyayı veya bağımsız bir disk bölümünü takas alanı olarak kullanabilir. Bağımsız bir takas bölümü daha hızlı iken, bir takas dosyasının boyutunu değiştirmek daha kolaydır (bütün sabit diski yeniden bölümlemeniz gerekmez). Ne kadarlık bir takas alanına ihtiyacınız olduğunu biliyorsanız bağımsız bir takas bölümü oluşturun; ama şayet bundan emin değilseniz bir süre bir takas dosyası kullanın ve sistemin ne kadarlık bir takas alanına ihtiyaç duyduğundan emin olunca bir takas bölümü oluşturun.

Ayrıca bilmelisiniz ki Linux çeşitli takas bölümleri ve/veya dosyalarını aynı anda kullanmanıza izin verir. Şayet arada sırada daha büyük bir takas alanına ihtiyaç duyuyorsanız, bütün sistemi baştan ayarlamak yerine, kendinize bir takas dosyası yaratabilirsiniz.

İşletim sistemi terminolojisine ait bir not: bilgisayar bilimi takas (bütün işlemleri takas alanına yazmak) ve sayfalama (belirli bir zamanda belirli bir parçayı yazmak) olaylarını ayrı şeyler olarak almaktadır. Linux’un yapmış olduğu şey sayfalamadır ama Linux terminolojisinde bu takas olarak yerleşmiştir.

9.2. Bir takas alanının oluşturulması

Bir takas dosyası çekirdek ile özel bir bağı olmayan sıradan bir dosyadır. Çekirdeği ilgilendiren tek şey, o dosya içinde oyukların (hole) olmadığı ve **mkswap**’ın kullanımı için hazırlanmış olduğudur. Ayrıca yerel bir disk üzerinde bulunması zorunludur. Bir NFS sistemi üzerinde bulunması uygulama şartlarından dolayı mümkün değildir.

Oyuklar hakkındaki ayrıntı önemlidir. Bir takas dosyası belli bir disk yüzeyini ayırır ve böylece çekirdek hızlı bir şekilde sayfa değiş tokuşunu sağlar. Bunu yaparken normal zamanlarda, bir disk yüzeyinin bir dosya için ayrılması işlemi için gerekli olan bütün işlemleri yerine getirmek zorunda kalmaz. Çekirdek sadece dosyaya ayrılmış sektörleri kullanır. Çünkü bir dosya içindeki oyuklar, ayrılmış bir sektör olmadığı anlamına gelir ve bunları kullanmaya çalışmak çekirdek için hiç de iyi bir fikir değildir.

Aşağıdaki komutla içinde oyuklar olmayan bir takas dosyası oluşturabilirsiniz:

```
# dd if=/dev/zero of=/ek-takas bs=1024 count=1024
1024+0 records in
1024+0 records out
#
```

`/ek-takas` takas dosyasının adıdır ve boyutu `count=`’dan sonra belirtilmektedir. 4kB’ın katlarından oluşan bir alan ayırmak daha uygundur çünkü çekirdek bu alana 4 Kb lık boyutlar ile yazma yapar. Şayet 4kB’ın katları şeklinde bir alan ayırmazsanız, son bölüm kullanılamayan bir alan haline gelebilir.

Aslında bir takas bölümü özel bir bölüm değildir. Onu da diğer bölümleri yaratır gibi yaratırız, tek fark onun içinde işlenememiş bir alan bulunması ve bir dosya sistemi ihtiva etmemesidir. Aslında çekirdek için gerekli olmasa da

takas alanını tür 82 (linux takas) olarak işaretlemek iyi olur. Bu, bölüm listesinin daha temiz ve kolay anlaşılır olmasını sağlayacaktır.

Bir swap dosyası veya bölümü oluşturulduktan sonra onun başlangıç bölümüne, çekirdek tarafından kullanılan ve bazı yönetim bilgilerini içeren, bir imza/iz koymalısınız. Bunun için **mkswap** komutu kullanılır:

```
# mkswap /ek-takas 1024
Takas alanı sürüm 1, uzunluk = 1044480 bayt olarak ayarlanıyor
#
```

Takas dosyası var olmasına rağmen henüz kullanımda değildir. Çekirdek bu bölümü sanal bellek sağlaması için kullanmamaktadır.

Uyarı

mkswap komutunu kullanırken çok dikkatli olun, çünkü bu komut işlem yapacağı bölüm veya dosyanın herhangi bir şey içermesine önem vermez. Kolaylıkla son derece önemli bilgiler bulunan bir dosya veya bölüm üzerine yazabilirsiniz. Şans eseri, **mkswap** komutuna sadece işletim sistemini kurarken ihtiyaç duyabilirsiniz.

Linux bellek yönetimi çeşitli teknik sebeplerden dolayı takas alanı boyutunu 127 MB [(4096–10) * 8*4096= 139890048 byte] ile sınırlar. Bununla beraber toplam 8 adet takas alanı kullanabilirsiniz.

Aslında bu, şu an için doğru değildir. Yeni çıkan çekirdeklerin ve **mkswap** komutunun yeni sürümlerinin mimarilerine göre değişiklik göstermektedir. i386 ve üzeri için bu sınır 2GB dır. Takas alanları ile ilgili ayrıntılı bilgi **mkswap**(8) kılavuz sayfasında bulunabilir.

9.3. Bir takas alanının kullanılması

Hazırlanmış bir takas alanı **swapon** ile kullanıma sunulur. Bu komutla çekirdeğe takas alanını kullanabileceğini iletiriz. Bir fikir olarak, takas alanını yolu da belirtilebilir. Bu komutla geçici bir takas dosyası üzerine yazılmaya başlanır.

```
# swapon /ek-takas
#
```

Takas alanları `/etc/fstab` dosyası içinde listelenerek otomatik olarak kullanılabilirler.

```
/dev/hda8       none        swap        sw     0     0
/swapfile       none        swap        sw     0     0
```

Başlangıç betikleri **swapon -a** komutunu çalıştıracak ve böylece `/etc/fstab` dosyasında listelenmiş bütün takas alanları kullanıma açılacaktır. Bununla beraber, **swapon** komutu genellikle ek takas alanına ihtiyaç duyulduğu zamanlarda kullanılır.

Takas alanı kullanımını **free** komutu ile görüntüleyebilirsiniz.

```
$ free
             total       used       free     shared    buffers     cached
Mem:        512368     453692      58676      40144      35568     201236
-/+ buffers/cache:     216888     295480
Swap:      1951856          0    1951856
$
```

Çıktının birinci satırı (`Mem:`) fiziksel belleği gösterir. 1 Mb civarı, çekirdek tarafından kullanılan bellek, burada gösterilmez. `used` sütunu kullanılan bellek miktarını gösterir (ikinci sıradaki `-/+ buffers/cache:` sayılmaz).

`free` sütunu tamamen boş olan alanı gösterir. `shared` sütunu çeşitli uygulamalar tarafından paylaşılan alanı gösterir. `buffers` sütunu ise diskin tampon bellek alanını gösterir.

Son satır (`Swap:`) ise takas alanları için benzer bilgileri gösterir. Bu alan tamamen sıfır ise bu takas alanınızın etkinleştirilmediğini gösterir.

Aynı bilgiler **`top`** komutu ile veya proc dosya sistemindeki `/proc/meminfo` sayesinde de elde edilebilir. Belirli bir takas alanının kullanımı hakkında bilgi almak oldukça zordur.

Bir takas alanı **`swapoff`** ile kullanımdan kaldırılabilir. Bu normal olarak gerekli değildir, sadece geçici takas alanlarını kaldırmak için kullanılır. Takas alanı içinde öncelikle kullanımdaki sayfalar takas edilir. Bunları tutacak yeterli fiziksel bellek yok ise bir diğer takas alanına gönderilirler. Şayet bu sayfaları tutacak yeterli sanal bellek yok ise Linux kıvranmaya başlar. Uzun bir süre sonra kendine gelir fakat bu arada sistemi kullanamazsınız. Kullanımdaki bir takas alanını kaldırmadan önce **`free`** komutu ile yeterli miktarda boş bellek bulunup bulunmadığını kontrol etmelisiniz.

`swapon -a` ile otomatik bağlanan bütün takas alanları **`swapoff -a`** ile kaldırılabilir. Sistem neyi kaldıracağını bulmak için `/etc/fstab` dosyasının içine bakar. El ile açılmış takas alanları ise kullanılmaya devam eder.

Bazı zamanlar büyük miktarda boş fiziksel bellek olmasına rağmen, pek çok takas alanı kullanımda olabilir. Bu olabilir, örneğin: bir yerde takas işlemine ihtiyaç vardır ama daha sonra bellekte çok yer kaplayan bir program öldürülür ve bellek boşaltılır. Dışarıya takas edilmiş bilgi ihtiyaç duyulana kadar geri alınmaz. Bunun için boş yere endişe duymaya gerek yoktur fakat ne olup bittiğini bilmek rahatlatıcı olabilmektedir.

9.4. Takas alanının başka işletim sistemleriyle paylaşılması

Pek çok işletim sisteminde sanal bellek vardır. Bu alana sadece çalıştıkları vakit ihtiyaç duydukları için, çalışmayan işletim sistemleri için ayrılmış olan alan boşa israftır. Hepsinin tek bir takas alanını paylaşması daha etkili bir çözüm olacaktır. Bu mümkün fakat biraz beceri ister. Bunun nasıl yapılacağına dair bir HOWTO (NASIL) belgesi vardır.

9.5. Takas alanının ayrılması

Bazı insanlar size, sahip olduğunuz RAM miktarının iki katı kadar bir takas alanı ayırmanızı söyleyebilirler. Bu tamamen geçersiz/sahte bir kuraldır. Bunun doğru bir şekilde nasıl yapılacağı aşağıda anlatılmaktadır.

- Yaklaşık bellek ihtiyacınızı belirleyin. Bu genellikle bir seferde çalıştıracağınız programların ihtiyaç duyacağı toplam RAM miktarıdır. Yani bir seferde en fazla ne kadar belleğe ihtiyacınız olduğunu bulun. Bunu aynı anda çalıştırmak istediğiniz programları çalıştırarak öğrenebilirsiniz.

  Şayet X oturumunu çalıştırmak istiyorsanız 8MB, gcc için, bazı zamanlar 10 MB civarı belleğe ihtiyaç duysa da, genellikle 4 MB. Çekidek 1MB alanı kendine ayırır. Kabuklar ve diğer birkaç araç yüzlerce KB yer ister – bunu da 1 Mb olarak kabul edebiliriz. Bunu tam olarak tahmin etmek gereksizdir, kabaca ne kadarlık bir belleğe ihtiyaç duyduğumuzu bulmak yeterli olacaktır. Fakat siz kötümserlerin yanında yer almak isteyebilirsiniz.

  Aklınızdan çıkarmayın ki; sistemde birden fazla kullanıcı aynı anda yer alacaksa, hepsi ayrı ayrı bellek tüketeceklerdir. Bununla beraber, iki kişi aynı anda aynı programı kullanıyorsa; aynı veri ve kütüphaneleri kullanıyor olacaklarından bellek tüketimi iki kat olmayacaktır.

  Bellek ihtiyaçlarını değerlendirmek için **`free`** ve **`ps`** komutları oldukça uygundur.

- Birinci adımda anlatılan değerlendirmeler için bir parça yanılma payı ekleyin. Büyük ihtimalle kullanmak istediğiniz bazı programları bu hesaplama sırasında unutabilirsiniz, bazılarının gereksinimlerini yanlış

hesaplayabilirsiniz veya ileride daha fazla alan ihtiyacı duyabilirsiniz. Birkaç MB disk alanını fazladan ayırmak her zaman iyi olur. Çok küçük bir takas alanı yapmaktansa, biraz büyük bir alan yaratmak her zaman daha iyidir. Ama bunu da fazla abartıp bütün diski takas alanı olarak ayırmayın. Unutmayın ki kullanılmayan alan boşa gitmiş sayılır. Daha sonradan da takas alanına ekleme yapılabileceğini göz önünde bulundurun. Tam sayılarla uğraşmak daha kolay olacağından ihtiyaç duyulan alan hesaplamasını yuvarlayabilir, bir MB büyük alabilirsiniz.

- Yukarıdaki karşılaştırmalar ve hesaplamalar sonucu şu an ne kadarlık bir belleğe ihtiyaç duyabileceğinizi biliyor olmalısınız. Yerleştireceğiniz takas alanını hesaplamak için, toplam ihtiyaç duyulan alandan fiziksel belleği çıkarın. Kalan sonuç takas alanının boyutunu belirtir. Bazı Unix sürümlerinde bu alanı RAM'in bir yansıması gibi yerleştirmeniz gerekeceğinden, ikinci basamakta hesaplanan alan sizin ihtiyacınızı gösterecek ve çıkartma işlemine gerek kalmayacaktır.
- Şayet hesaplarınız sonucu ortaya çıkan takas alanı, fiziksel belleğinizin bir kaç katıysa, yeni bir RAM eklemenin zamanı gelmiş demektir. Aksi taktirde performansta çok büyük bir düşüş yaşayabilirsiniz.

Hiç ihtiyacınız olmasa bile bir parça takas alanına sahip olmak iyi olur. Linux, takas alanını oldukça tasarruflu bir biçimde kullanır ve bu size mümkün mertebe boş bir fiziksel bellek sağlar. Takas ihtiyacı olmasa bile, Linux kullanılmayan bellek sayfalarını takas alanına gönderir. Bu sayede, disk pasif halde iken takas yapılarak takasa ihtiyaç duyulduğu anlarda bekleme yapılması önlenir.

Takas alanı çeşitli diskler arasında bölünebilir. Bazı zamanlarda, disklerin hızları ve erişim durumlarına göre, bu performans artışı sağlayabilir. Birkaç değişik alternatif alan tablosu ile uğraşmak isteyebilirsiniz ama bunun oldukça zor bir iş olduğunu ve aslında hiç birinin diğerine karşı büyük bir üstünlüğünün olmadığını unutmayın. Asla tam doğruyu bulamazsınız.[9]

9.6. Tampon bellek

Bir diskten okumak gerçek bellekten okumaktan çok daha yavaştır. Ek olarak, nispeten kısa zaman aralıkları ile aynı disk yüzeyi tekrar tekrar okunacaktır. Örneğin birisi; önce gelmiş bir mesajı okuyabilir, daha sonra cevaplarken bir metin düzenleyiciden o mesajı okur ve postalarken, sistemin eposta dizinine bir kopyasını alması için onu okutabilir. Veya **ls** komutunun bir sistem üzerinde kullanıcılar tarafından ne kadar sıklıkla kullanıldığını bir düşünün. Diskten bir kere okutup, bu bilgiyi işi bitince bellekte tutmak yöntemiyle sistem hızını artırabiliriz fakat önce okuyun. Bu işleme disk tamponlanması ve bu iş için kullanılan belleğe *tampon bellek* denir.

Bellek sınırlı ve kıt kaynaklara sahip olduğu için tampon bellek yeterli büyüklükte olamaz (ihtiyaç duyulan tüm bilgileri bir seferde okutmak için). Tampon dolduğu zaman, uzun süre kullanılmayan bilgiler yeni bilgilerin alınabilmesi için silinir.

Disk tamponlama tabii ki yazarak çalışır. Bir yanda, sık sık yazılan ve hemen ardından okunan bir bilgiyi (örneğin; bir dosyaya kopyalanan bir kaynak kodu dosyası ve ardından onu okuyan derleyici program) tampon belleğe yerleştirmek iyi bir fikir olabilir. Diğer yandan, bilgiyi tampona yerleştirerek yazma işleminin bir seferde yapılmasını sağlayamazsınız çünkü yazma programı oldukça yavaştır. Yazma işlemi, diğer programları yavaşlatmaksızın, artalanda yapılır.

Pek çok işletim sistemi bu isimle anılmasa da bir önbelleğe sahiptirler. Fakat çalışma prensipleri farklı olabilir. Bazıları doğrudan yazma (write–through) prensibiyle, bilgiler önbellek içinde saklanırken diske bir seferde yazma işlemini yaparlar; yazma işlemi daha sonra yapılan önbellekler *arkadan yazma* (write–back) olarak adlandırılırlar. Arkadan yazma, doğrudan yazma işlemine göre daha etkili ama hatalara daha meyillidir. Şayet sistem kilitlenirse, elektrik kesilirse veya yazma işlemini yapmak için beklerken disket sürücüde bulunan disket çıkarılırsa, genellikle önbellek içinde yapılan değişiklikler kaybolur. Belki de yazılamayan bilgiler çok önemli muhasebe/kayıt bilgileri içermekteydi. Bu yüzden de dosya sistemi tam kapasite kullanıma geçemiyor da olabilir.

Bu sebeplerden dolayı sistemi her zaman normal bir şekilde kapatın. (*Açılışlar ve Kapanışlar* (sayfa: 52) bölümüne de bakın.) Bağlı bir disketi **unmount** komutu ile sistemden ayırmadan yuvasından çıkarmayın, elektrik akımını sistem açıkken kapatmayın, disket sürücünün işini bitirmesini bekleyin ve işlem yapıldığını gösteren ışık sönmeden disketleri yerinden çıkarmayın. **sync** komutu önbelleğe baskı uygular ve bütün yazılmamış ama yazılması gereken bilgilerin disk üzerine yazılmasını sağlar. Bu komutla işlemin tam yapıldığından emin olabilirsiniz. Geleneksel Unix sistemlerinde her 30 saniyede bir **sync** komutunun çalışmasını sağlayan **update** isimli bir program bulunur. Bu program arka planda çalışır ve bu sayede **sync** komutunu kullanmanıza gerek kalmaz. Linux **sync**'in ağır disk G/Ç hatalarına sebep olması nedeniyle meydana gelen ani donmaları önlemek amacıyla daha etkili ve doğru **sync** sinyali gönderilmesini sağlayan **bdflush** isimli bir programı kullanır.

Linux altında **bdflush**, **update** tarafından başlatılır. Genellikle endişelenecek bir durum olmaz ama her hangi bir sebepten dolayı **bdflush** durur ise, çekirdek sizi uyaracaktır. Bunu **/sbin/update** kullanarak elle yeniden başlatabilirsiniz.

Önbelleğin etkili olamamasının sebebi onun boyutlarına bağlıdır. Çok küçük bir önbellek içinde bulunan bilgiler tekrar kullanılamadan, yeni gelenlere yer açmak için silinecektir. Kritik alan; aynı bilgiye hangi sıklıkla ulaşıldığı, ne kadar bilgi okunup yazıldığıdır.

Sabit boyutta bir önbellek alanınız varken onu çok büyük bir hale getirmek iyi olmayabilir. Çünkü bu gerçek belleğin boş alanının kısıtlanmasına ve çok yavaş olan takas işlemine ihtiyaç duyulmasına sebep olabilir. Gerçek belleğin daha verimli kullanılması için, Linux otomatik olarak boştaki bütün RAM alanını disk önbelleği olarak ayırır ve kullanır. Şayet bir program RAM alanına ihtiyaç duyarsa Linux önbellek alanını otomatik olarak küçültür.

Linux altında çalışırken önbelleği kullanmak için özel bir işlem yapmanız gerekmez, her şey otomatik olarak yapılmaktadır. Kapatma ve disket çıkarma işlemlerini düzgün uyguladığınız sürece, hiçbir şey hakkında endişelenmeniz gerekmez.

10. Açılışlar ve Kapanışlar

```
Start me up
Ah... you've got to... you've got to
Never, never never stop
Start it up
Ah... start it up, never, never, never
 You make a grown man cry,
  you make a grown man cry
(Rolling Stones)
```

Bu bölümde açılış ve kapanışlar esnasında Linux içinde neler olduğu ve bu işlemin düzgün bir şekilde nasıl yapılacağı anlatılmaktadır. Şayet işlemleri düzgün bir şekilde izlemezseniz dosyalar kaybolabilir veya bozulabilir.

10.1. Açılışlar ve kapanışlara genel bir bakış

Bilgisayar sistemine enerji verilmesi ve işletim sisteminin yüklenmeye başlaması olayına önyükleme (booting) denir. İsim, kendini kendini bir bilgisayardan kendi çabası ile çeken bir bilgisayar resminden gelmektedir, fakat eylemin kendisi biraz daha gerçekçidir.

Kendi kendini çekme çabası esnasında ilk önce, işletim sisteminin yüklenmesini ve başlatılmasını sağlayan küçük bir kod parçası yüklenir. Bu kodlara önyükleyici denir. Önyükleyici bir disket ya da sabit disk üzerinde sabit bir yerde bulunur. Bu iki basamaklı işlemin sebebi, işletim sisteminin büyük ve komplike olması, fakat bilgisayarın yükleyebildiği kodların ilk parçasının küçük olması (birkaç yüz byte) ve donanımsal yazılımların karmaşık hale gelmemesini sağlamaktır.

Bilgisayar mimarilerinde bu önyükleme işlemini farklılık gösterir. PC'ler için; BIOS, sabit disk veya disketten önyükleme sektörü diye adlandırılan ilk sektörü okur. Önyükleyici bu sektör içindedir. Böylece işletim sistemi disk üzerinde her nerede ise önyükleyiciden bu bilgi okunup o bölümden yüklenir.

Linux çekirdeği yüklendikten sonra donanım ve aygıt sürücüleri yüklenir ve sonra **init** çalıştırılır. Kullanıcıların bağlanması ve diğer işlerin yapılması için gereken süreçler **init** tarafından yüklenir. Bunun ayrıntıları aşağıda anlatılmıştır.

Bir Linux sistemini kapatmak için, ilk önce çalışan süreçlere; işlerini sonlandırıp, kendilerini durdurmaları söylenir. Daha sonra dosya sistemleri ve takas alanı ayrılır ve en sonunda sistemin kapatıldığını söyleyen bir mesaj belirir. Şayet bu süreç düzgün bir şekilde takip edilmezse korkunç şeyler olabilir ve genellikle olur; en önemlisi dosya sisteminin tampon belleği içinde bulunan bütün bilgi kaybolmuş ve sistem istikrarsız bir hale dönmüş olabilir. Ve bu nedenle kullanılmaz bir hale de gelebilir.

10.2. Önyükleme sürecine yakından bakalım

Linux işletim sistemini sabit diskten veya bir disketten başlatabilirsiniz. "Kurulum ve Başlatma Kılavuzu"nun (Installation and Getting Start Guide) Kurulum bölümü Linux'un nasıl kurulacağını ve nasıl istediğiniz şekilde açılış işlemini gerçekleştirebileceğinizi anlattır.

Bir PC açıldığında BIOS her şeyin sağlıklı bir şekilde çalışıp çalışmadığını anlamak için pek çok testler yapar[10] ve daha sonra asıl açılış işlemine geçilir. Bir disk sürücüsü seçip onun ilk sektörünü okur. Genel seçim sırası şu şekildedir: ilk önce disket sürücüye bakılır, orada bir disket yok ise sabit diske bakılır, orada da önyükleme sektörü yoksa CDROM sürücü ve diğer aygıtlara bakılır. Bu açılış sırası kullanıcıya göre değiştirilebilir. Bu ilk sektöre önyükleme sektörü denir, şayet bu önyükleme sektörü bir sabit disk üzerinde ise buna *Ana Önyükleme Sektörü* (Master Boot Sector – MBR) adı verilir, çünkü sabit disk üzerinde pek çok bölüm ve her bölümün kendine ait bir önyükleme sektörü olabilir.

Önyükleme sektörü, sorumluluğu gerçek işletim sistemini diskten okuyup başlatmak olan, küçük bir (ama bir sektörü doldurur) program içerir. Şayet disketten açılış yapılacaksa önyükleme sektörü içerisindeki kodlar sadece, bellek içine daha önceden kararlaştırılmış bir yerde bulunan ilk birkaç yüz bloğu okur. Tabii ki bu blokların sayısı çekirdeğin gerçek boyutuna dayanmaktadır. Bir Linux açılış disketi içinde dosya sistemi yoktur. Açılış işlemini basitleştirmek için ardışık bloklar içine yerleştirilmiş çekirdek kodları bulunur. Bununla beraber içinde LILO (LInux LOader – Linux Yükleyicisi) bulunan ve bir dosya sistemine sahip bir disket ile de sistemi açabilirsiniz.

Bir sabit diskten açılış yapılırsa önce MBR içindeki kodlar bölümleme tablosunu inceler, açılabilir olarak işaretlenmiş etkin bölümü bulur, o bölümden önyükleme sektörünü okur ve daha sonra bu önyükleme sektörü içindeki kodları çalıştırır. Bu önyükleme sektörü içerisinde bulunan kodlar, bir açılış disketinin önyükleme sektörü içinde bulunan kodlar ne yaparsa aynısını yapar: bölüm içerisindeki çekirdeği okur ve onu başlatır. Ayrıntılar çeşitli olmakla birlikte; sadece çekirdek için ayrı bir bölüme sahip olmak genellikle pek kullanışlı olmaz. Çekirdek için ayrı bir bölüm açılırsa, önyükleme sektörü içerisindeki kodlar gerekli bilgileri zincirleme bir halde okuyamaz ve dosya sistemi onları hangi sektörlere koydu ise oraları bulmak zorunda kalırlar. Bu sorunu çözmenin pek çok yöntemi vardır ama en genel kabul göreni LILO'dur. Bunun konumuzla bir alakası yoktur, ama LILO belgelerine bakmanız yararlı olabilir.

LILO ile açılış işleminde, doğrudan diskin başına gidilir ve buradan öntanımlı çekirdeğin yeri okunarak çekirdek yüklenir. LILO ile çeşitli yapılandırmalar mümkündür. Çekirdekler arasında seçim yapılabilir, farklı işletim sistemlerini yüklenebilir ve açılış esnasında kullanıcının bu işletim sistemlerinden birisini seçmesi sağlanabilir. LILO yüklenirken <**alt**>, <**shift**> veya <**ctrl**> tuşlarından birine basılıp, LILO'da öntanımlı yükleme seçeneğinin yerine, ne yüklenmesini istediğinizi soracak şekilde de ayarlamalar yapabilirsiniz. Alternatif olarak; LILO'yu seçenekler sunacak, şayet her hangi bir seçim yapılmazsa belirli bir süre sonra öntanımlı çekirdeği yükleyecek şekilde de ayarlayabilirsiniz.

LILO ile çekirdeğe, çekirdeğin ya da işletim sisteminin isminden sonra bir komut satırı seçeneği vermek mümkündür.

Disketten veya diskten açmak çeşitli getirilere sahip olmakla birlikte, sabit diskten açmak disketle uğraşma zahmetinden bizi kurtaracaktır. Ayrıca daha hızlıdır. Bununla birlikte, sistemi sabit diskten açılacak şekilde yüklemek bazen çok zahmetli olabilmektedir. Bu nedenle pek çok kullanıcı ilk önce disketten açılış yolu ile kurulum yapar ve her şeyin düzgün çalıştığını gördükten sonra LILO'yu ayarlayarak disk açılış kipine geçer.

Linux çekirdeği belleğe okunduktan sonra ya da adına ne dersin deyin, gerçekte başlatıldıktan sonra, kabaca aşağıdaki olaylar gerçekleşir:

- Linux çekirdeği diskte sıkıştırılmış olarak tutulur. Bu nedenle ilk önde kendi kendini çözer. Bu, çekirdeğin başlangıcındaki küçük bir programla yapılır.
- Şayet Linux'un tanıdığı ve çeşitli metin kipleri bulunan Super–VGA bir ekran kartınız varsa, Linux size hangi metin kipinde çalışmak istediğinizi sorar. Çekirdek derlemesi sırasında bir video kipi girilebilir ve böylece daima bu kip kullanılır. Bu ayrıca LILO veya **rdev** ile de yapılabilir.
- Bundan sonra çekirdek diğer donanımları kontrol eder (sabit disk, disket sürücü, ağ bağdaştırıcısı, vb) ve bazı donanım sürücülerini uygun bir şekilde yapılandırır ve bunu yaparken buldukları ile ilgili bir çıktı verir. Örneğin ben sistemi açtığım zaman, şuna benzer bir mesaj almaktayım:

```
LILO boot:
Loading linux.
Console: colour EGA+ 80x25, 8 virtual consoles
Serial driver version 3.94 with no serial options enabled
tty00 at 0x03f8 (irq = 4) is a 16450
tty01 at 0x02f8 (irq = 3) is a 16450
lp_init: lp1 exists (0), using polling driver
Memory: 7332k/8192k available (300k kernel code, 384k reserved, 176k
data)
Floppy drive(s): fd0 is 1.44M, fd1 is 1.2M
Loopback device init
Warning WD8013 board not found at i/o = 280.
Math coprocessor using irq13 error reporting.
Partition check:
  hda: hda1 hda2 hda3
VFS: Mounted root (ext filesystem).
Linux version 0.99.pl9-1 (root@haven) 05/01/93 14:12:20
```

  Bu çıktı sistemdeki donanımlara, kullanılan işletim sistemi çeşidine ve yapılmış olan ayarlara göre değişiklik gösterebilir.

- Daha sonra çekirdek kök dosya sistemini bağlamayı deneyecektir. Yer ayarlaması derleme esnasında veya herhangi bir zaman **rdev** komutu veya LILO ile yapılabilir. Dosya sistemi türü otomatik olarak bulunur. Kök dosya sisteminin bağlanması esnasında bir terslik olursa, örneğin: çekirdek içinde uygun dosya sistemi sürücüsü yoksa, çekirdek PANİKLER ve sistemi kapatır. Aslında yapabileceği başka da bir şey yoktur zaten.

  Kök dosya sistemi genellikle salt okunur bir şekilde bağlanır. Bu dosya sistemi bağlı durumdayken, dosya sisteminin sınanmasını sağlar. Oku–yaz kipinde bağlanmış bir dosya sistemi üzerinde sınama yapmak hiç de iyi bir fikir değildir. Burada bahsedilen sınama işlemi **fsck** ile yapılır.

- Bundan sonra çekirdek, **init** (/sbin/init) programını başlatır (daima süreç numarası olarak 1'i alır). **init** pek çok ufak tefek iş yapar. Aslında yaptıkları ne şekilde ayarlandığına bağlıdır. *init* (sayfa: 56) bölümünde daha ayrıntılı (henüz yazlmadı) bilgi bulabilirsiniz. En azından, çok gerekli olan birkaç artalan sürecini çalıştırır.

- **init** daha sonra çok kullanıcılı kipe geçer, sanal konsollar ve seri bağlantılar için **getty** programını çalıştırır. **getty**, kişilerin sanal konsollar veya seri hatlar üzerinden sisteme bağlanmasını sağlar. **init** ayrıca nasıl ayarlandığına bağlı olarak diğer programları da çalıştırır.
- Bütün bunlardan sonra açılış işlemi bitmiş olur, artık sistem açık ve normal çalışma kipine geçmiş haldedir.

10.3. Kapanışla ilgili ayrıntılar

Bir Linux sistemini kapatırken, doğru süreci izlemek çok önemlidir. Şayet bunu doğru yapmazsanız, dosya sisteminiz değersiz bir çöp yığınına dönebilir. Bunun sebebi, Linux'un bir kerede doğrudan değil de daha sonra yazan (write–back) bir disk önbellekleme sistemine sahip olmasıdır. Bu performans artışı sağlamakla birlikte bazı problemleri de yanında getirmektedir. Şayet bir kapris anında enerjiyi keserseniz, önbellekte bulunan ama henüz diske yazılmamış olan bütün bilgiler yok olup gidecektir.

Başka bir sebep de, galeyana gelip artalanda çalışan pek çok programın olduğu, çok işlevli bir sistemin açma/kapama düğmesine vurursanız, bu tam bir yıkım olabilir. Sistemi uygun bir şekilde kapatarak, artalanda çalışan bütün programların verilerini kurtardığından emin olabiliriz.

Bir Linux sistemini kapatmak için gereken komut **shutdown**'dır[11]. Genellikle bunu yapmak için iki yoldan birisi kullanılır.

Tek kullanıcısı olduğunuz bir sistemde alışılmış yöntem: çalıştığınız bütün programlardan çıkmak, bütün sanal konsollardan çıkıp, root kullanıcı olarak tekrar girmek (şayet hali hazırda root olarak bağlıysanız orada beklemek – ama kök dizininin veya root'un ev dizininin (`/root` dizini) dosya sistemlerinin ayrılması sırasında bir sorunla karşılaşmasını önlemek için, o dizinlerden başka dizinlere geçin) ve daha sonra **shutdown -h now** komutunu verin (`now` parametresi yerine bir artı işaretiyle birlikte dakikaları belirten sayılar da kullanabilirsiniz. Böylece, genellikle tek kullanıcılı bir sistemde olmasanız bile kapanma işlemi istenen süre kadar ertelenmiş olur).

Alternatif olarak **shutdown -h +süre uyarı** komutunu kullanabilirsiniz. `süre` sistemin ne kadarlık bir süre sonra kapatılacağını, `uyarı` ise ne sebeple kapatılacağını belirten küçük bir uyarı metnidir.

```
# shutdown -h +10 'Yeni bir disk ekliyoruz. Sistem 3 saat kapalı kalacak.'
#
```

Bu herkesi, sistemin 10 dakika sonra kapatılacağını ve verilerini kaybedebilecekleri konusunda uyarır. Bu uyarı X uçbirimleri dahil birilerinin bağlı olduğu bütün uçbirimlere gönderilir. Ve birkaç kez daha uyarı gider. Kapanma anı yaklaştıkça uyarılar sıklaşır.

```
Broadcast message from root (ttyp0) Wed Aug  2 01:03:25 1995...

Yeni bir disk ekliyoruz. Sistem 3 saat kapalı kalacak.
The system is going DOWN for system halt in 10 minutes !!
```

Erteleme süresi sonunda gerçek kapatma işlemi başlayınca; kök dosya sistemi hariç bütün dosya sistemleri ayrılır, halen bağlı kullanıcı olsa bile bütün kullanıcı süreç ve programları öldürülür, artalan süreçleri kapatılır, bütün dosya sistemleri ayrılır ve her şey tatlıya bağlanmış olur. Bu yapılırken, **init** makineyi kapatabileceğinizi söyleyen bir mesaj yayınlar. Bundan sonra yapmanız gereken tek şey parmağınızı enerji düğmesine dokundurmaktır.

Bazen, nadiren iyi sistemlerde bile, düzgün bir kapanış yapmak mümkün olmaz. Örneğin; şayet çekirdek paniklemişse, çöker ve hatalı davranmaya başlar, bu durumda yeni bir komut vermek imkansız olabilir. Düzgün bir şekilde kapatmanız mümkün olmayınca yapabileceğiniz tek şey, başınıza kötü bir şey gelmemesini ummak ve sistem enerji düğmesine basmaktır. Şayet sorun biraz daha büyükse (örneğin birisi baltası ile klavyenize vurduysa) ve çekirdekle **update** halen düzgün bir şekilde çalışıyorsa, **update**'in işlevini yapması ve önbellekte

bulunan bilgileri yazılmaya zorlaması için birkaç dakika beklemek daha uygun olacaktır. Ondan sonra enerjiyi kesebilirsiniz.

Bazı insanlar **sync** komutunu 3 kere ard arda kullanıp disk G/Ç'larının bitmesini bekleyip sistemi kapatmayı tercih ederler. Şayet çalışan bir program yok ise bu işlem **shutdown** komutu ile aynı işi yapmış olur. Bununla beraber bu işlem sırasında dosya sistemleri ayrılmadığı için ext2fs "temiz dosya sistemi" bayrağı ile ilgili hatalar ortaya çıkabilir. Bu nedenle üçlü **sync** komutu ile sistemi kapatmak *tavsiye edilmez*.

(Bilgi: Üçlü sync'in sebebi, komutların ayrı ayrı yazıldığı Unix'in ilk zamanlarında disk G/Ç işlemlerinin bitmesi için yeterli zamanı kazanmaktı.)

10.4. Sistemin yeniden başlatılması

Sistemi yeniden başlatmak için sistemi kapatıp, elektriği kesip, sonra tekrar vererek de yapabilirsiniz. Daha basit bir yol ise, sadece kapatmak yerine, **shutdown** komutunun sistemi yeniden başlatmasını sağlamaktır. Bunu `-r` seçeneğini kullanarak yapabiliriz. Örneğin; **shutdown -r now** yazmanız yeterlidir.

Pek çok Linux sistemi ctrl+alt+del tuşlarına aynı anda basılmasıyla yeniden başlatma işlemini yapabilmektedir. Bu **shutdown -r now** komutunu çalıştırmış olur. Bu ctrl+alt+del tuşları yapılandırılabilir. Çok kullanılıcılı bir sistemde bir miktar erteleme zamanı vermek uygun olacaktır. Başkalarının fiziksel olarak ulaşabilecekleri yerlerde bulunan sistemlerde bu tuşların işlevi kapatılabilir.

10.5. Tek kullanıcı kipi

shutdown komutu sistemi tek kullanıcılı hale getirmek için de kullanılabilir. Böylece sisteme sadece konsoldan root bağlanır ama başkaları bağlanamaz. Bu yöntem, sistem normal çalışırken yapılamayan, sistem yöneticisi görevlerini yapmak için yararlı olabilir.

10.6. Kurtarma Disketleri

Bilgisayarı sabit diskten açmak her zaman mümkün olmayabilir. Şayet LILO ayarları sırasında bir hata yaparsanız, sistemin önyükleme özelliğini kullanamayabilirsiniz. Bu nedenle, böyle durumlarla karşılaştığınızda kullanabileceğiniz ve donanımsal bir sorun olmadıkça her zaman işe yarayacak olan bu alternatif yola ihtiyacımız olacaktır. Tipik PC'ler için bu işlem disketten açılış olayına benzer.

Pek çok Linux dağıtımı yükleme esnasında bir Açılış Disketi oluşturmanıza izin verir. Bunu yapmak iyi bir fikirdir. Bununla birlikte, bu tür disketlerin bazıları sadece çekirdek içerir ve karşılaştığınız problemi çözmek için dağıtıcının yükleme diskindeki programları kullanacağınızı varsayarlar. Bazen bu programlar yeterli olmayabilir. Örneğin; bu yükleme diskleri içinde bulunmayan programlar ile yapılmış yedekleme yazılımlarına ihtiyacınız olabilir.

Bundan dolayıdır ki, özel bir kök dosya sistemli bir Kök Disketi yapmanız gerekli olabilir. Graham Chapman tarafından yazılmış olan *Bootdisk HOWTO* belgesi, bunun için gerekli yapılandırmaları anlatmaktadır. Tabii ki kök disketinizi ve açılış disketinizi güncellemeyi unutmayın.

Kök disketi için bağlı bir disket sürücüyü başka bir amaçla kullanamazsınız. Şayet tek bir disket sürücünüz var ise, bu durum büyük sıkıntılar yaratabilir. Bunun yanında, şayet yeterli belleğiniz varsa; açılış disketinizi, kök disketini RAM'e yükleyecek şekilde ayarlayabilirsiniz (açılış disketinin çekirdeği bunun için özel ayarlı olmalıdır). Kök disketi RAM'e yüklendikten sonra, disket sürücüyü diğer disketler için rahatlıkla kullanabilirsiniz.

11. init

"Uuno on numero yksi" (Bir Finlandiya dizi filmi sloganı.)

Bu bölümde çekirdek tarafından başlatılan ve ilk kullanıcı seviyeli süreç olan **init** anlatılmaktadır. **init** çok önemli görevlere sahiptir. Örneğin **getty** programını başlatır (bu sayede kullanıcılar sisteme bağlanabilir), kullanıcı seviyelerini ayarlar, öksüz süreçler ile ilgilenir. Bu bölüm **init**'in nasıl ayarlandığını ve çalışma seviyelerinin nasıl kullanılacağını anlatmaktadır.

11.1. İlk önce init gelir

init bir Linux işletim sistemi için kesinlikle gerekli olan programlardan birisidir. Fakat siz bunu halen önemsemiyor olabilirsiniz. İyi bir Linux dağıtımı, pek çok sistem altında çalışabilecek şekilde ayarlı bir **init** ile birlikte gelir; ve bu sistemler altında **init** ayarları ile uğraşmanız gerekmez. Genellikle, çalışma seviyesini değiştirmek istediğiniz zamanlarda, seri uçbirimler üzerinden veya modemler üzerinden bağlanmaya çalıştığınız zamanlarda **init** ile ilgilenmeniz gerekir.

Çekirdek kendini başlattığı (belleğe yüklendiği, çalışmaya başladığı ve aygıt dosyaları, veri yapıları ve benzeri şeyleri başlattığı zaman) ve kullanıcı seviyeli bir program olan **init** sürecini başlattığında, kendi üstüne düşen açılış işlemlerini bitirmiş olur. Bundan dolayı **init** her zaman için ilk süreçtir ve süreç numarası da daima 1'dir.

Çekirdek, **init** için daha önceden de kullanılmış olan birkaç yere bakar ama genellikle bir Linux sistemi altındaki en uygun yer `/sbin/init`'dir. Şayet çekirdek **init**'i bulamazsa `/bin/sh`'yi çalıştırmayı dener, eğer bunu da beceremezse sistem hatasını başlatır.

init başladığı zaman, idari görevlerce gerçekleştirilen; sistemlerinin kontrol edilmesi, `/tmp`'nin boşaltılması, çeşitli servislerin başlatılması ve kullanıcıların bağlanabileceği her uçbirim ve sanal konsol için **getty**'nin başlatılması vb. işlemlerin başlatılması ile açılış işlemi bitirilmiş olur (*Kullanıcı Giriş ve Çıkışları* (sayfa: 60) bölümüne bakınız).

Sistem düzgünce açıldıktan sonra, **init** kullanıcı tarafından terk edilen her uçbirim için ayrı ayrı bir **getty** süreci başlatılır. Böylece bir sonraki kullanıcı buraya bağlantı yapabilir. **init** ayrıca her öksüz süreci evlat edinir: bir süreç yavru bir süreç başlatır ve bu yavru süreçten önce ölürse, bu öksüz yavru süreç **init**'in yavrusu olur. Bu pek çok teknik sebepten kaynaklanır; bunun böyle olduğunu bilmek süreç listesi ve süreç yapı ağacını anlamayı kolaylaştırması açısından faydalıdır. [12] **init**'in kullanılabilir birkaç çeşidi vardır. Pek çok Linux dağıtımı, System V **init** tabanlı **sysvinit** (Miquel van Smoorenborg tarafından yazılmıştır) kullanırlar. Unix'in BSD versiyonu farklı bir **init** kullanır. Birinci fark çalışma seviyelerindedir: çalışam seviyeleri System V'de bulunurken, BSD'de bulunmazlar (en azından geleneksel olanlarında). Bu temel farklılık değildir. Biz sadece **sysvinit**'e bakacağız.

11.2. init'in getty'yi başlatmak için yapılandırılması: /etc/inittab dosyası

Başladığı zaman, **init** `/etc/inittab` yapılandırma dosyasını okur. Şayet HUP sinyali gönderirseniz, [13] bu dosyayı tekrar okur. Böylece değişikliklerin etkin olması için sistemin yeniden başlatılması gerekmez.

`/etc/inittab` dosyası biraz karışık bir yapıdadır. Biz **getty** programının basit yapılandırılması ile başlayalım. `/etc/inittab` satırları ikinokta üstüste işaretleri ile ayrılmış 4 sütun içerirler:

```
id:çalışma_seviyeleri:eylem:süreç
```

Bu alanlar aşağıda tanımlanmıştır. Ek olarak `/etc/inittab`, boş satırlar ve `#` işaretiyle başlayan satırlar içerebilir. Tüm bunlar göz ardı edilir.

id

Dosya içerisindeki satırı tanımlar. **getty** satırları için, üzerinde çalıştığı uçbirimleri belirtir (aygıt dosya isimlerinde `/dev/tty`'den sonra gelen karakterler). Diğer satırlar için bu önemli değildir (uzunluk sınırlamaları dışında), fakat bu eşsiz olmalıdır.

çalışma_seviyeleri
Bu satırın hangi çalışma seviyeleri için kullanılacağı belirtilir. Çalışma seviyeleri tek rakamlarla ayraçsız olarak yanyana verilir. Gelecek bölümde bu seviyeler açıklanmıştır.

eylem
Satırın ne iş yapacağını belirtir. Örneğin; `respawn` ile sonraki alandaki komutun çıkışta tekrar çalıştırılacağını, `once` ile ise sadece bir kez çalıştırılacağını tanımlarız.

süreç
Çalıştırılacak komut.

Birinci sanal uçbirim üzerinde (`/dev/tty1`), normal çok kullanıcılı seviyelerde (2–5) bir **`getty`** programı çalıştırmak için şu satırı yazmalıyız:

```
1:2345:respawn:/sbin/getty 9600 tty1
```

Birinci bölüm bu satırın `/dev/tty1` için olduğunu belirtir. İkinci bölüm ise 2,3,4 ve 5. çalışma seviyelerinin kullanılacağını belirtir. Üçüncü bölüm ise, çıkışta bu komutun tekrar çalıştırılması gerektiğini ve böyle bir kullanıcının ayrıldıktan sonra isterse tekrar bağlanabilmesinin sağlanacağını belirtir. Son bölüm ise bu uçbirim üzerinde **`getty`** komutunun çalıştırılacağını belirtir. **`getty`** programının sürümüne göre çalışma şekli değişik olabilir. Kılavuz sayfasını kontrol etmeyi unutmayın.

Şayet bir sisteme uçbirimler veya aranacak modem hatları eklemek isterseniz, `/etc/inittab` dosyasına daha fazla satır eklemeniz gerekecektir. Her uçbirim veya hat için bir satır yazmak zorundasınız. Ayrıntılı bilgi için **`init`**, `inittab` ve **`getty`** kıavuz sayfalarına bakmanız faydalı olacaktır.

Bir komut çalıştırılmaya başlarken hata verirse ve **`init`** onu tekrar başlatmak için ayarlanmışsa, bu büyük miktarda sistem kaynağı tüketir: **`init`** komutu başlatır, komut hata verir; **`init`** komutu tekrar başlatır, komut hata verir; init komutu bir daha başlatır, komut yine hata verir ve bu sonsuza kadar devam eder. Bunu önlemek için, **`init`** bir komutu ne sıklıkta tekrar başlatacağının izlerini tutar. Şayet çalıştırma sıklığı artarsa, tekrar başlatmadan önce 5 dakika geçmesini bekler.

11.3. Çalışma seviyeleri

Bir çalışma seviyesi **`init`**'in durumunu ve sistem servislerinin ne tür işlevler gerçekleştirdiğini tanımlayan bir göstergedir. Çalışma seviyeleri *Çalışma seviyesi numaraları* (sayfa: 58) de açıklanmıştır. 2 ile 5 arasındaki kullanıcı tanımlı seviyelerin nasıl kullanılacağına dair ortak bir karar yoktur. Bazı sistem yöneticileri, çalışma seviyelerini hangi alt sistemlerin çalıştığını göstermek için ayarlarlar. Örneğin; X çalışıyor, ağ etkin, vb. Başkaları bütün alt sistemlerin çalıştığını veya teker teker başlatılıp durduklarını (çalışma seviyesi değişmeden) göstermek üzere ayarlayabilirler (çalışma seviyeleri sistemlerini kontrol etme konusunda çok kabadırlar). Kendiniz için karar vermelisiniz. Ama Linux sisteminizin tanımladığı seviyeleri kullanmak daha kolay olabilir.

Çalışma seviyesi numaraları

0	Sistemi kapatır.
1	Tek kullanıcılı kip (sistem yöneticisine özel).
2–5	Kullanıcı tanımlı normal işlev
6	Sistemi yeniden başlatır.

Çalışma seviyeleri `/etc/inittab` içinde aşağıdaki satıra benzer şekilde ayarlanır:

```
l2:2:wait:/etc/init.d/rc 2
```

İlk bölüm isteğe göre verilmiş bir etikettir, ikinci alan 2.çalışma seviyesinin uygulanacağını söyler. Üçüncü bölüm ise **`init`**'in dördüncü bölümde yer alan komutu bir sefer çalıştırmasını ve çalışma seviyesine girildiğinde **`init`**'in

işlemin tamamlanmasını beklemesi söylenir. /etc/init.d/rc komutu ise 2. çalışma seviyesine girmek için gerekli olan servislerin

Dördüncü alandaki komut, bir çalışma seviyesinin başlaması için gerekli olan bütün tatsız işleri yapar. Hali hazırda çalışmayan servisleri çalıştırır ve yeni seviyede daha fazla çalışmasına ihtiyaç duyulmayan servisleri kapatır. Tam olarak komutun ne olduğu ve çalışma seviyelerinin nasıl ayarlandığı Linux dağıtımına bağlıdır.

init başladığı zaman /etc/inittab dosyası içinde geçerli çalışma seviyesinin tanımlandığı satırı arar:

```
id:2:initdefault:
```

single veya emergency gibi bir çekirdek komutu vererek, **init**'ten geçerli olarak tanımlanmış çalışma seviyesinden başka bir seviyede başlamasını isteyebilirsiniz. Örneğin, çekirdek komut satırı LILO üzerinden de verilebilir. Bu size 1. çalışma seviyesini (tek kullanıcılı çalışma seviyesini) seçme şansı verir.

Sistem çalışırken **telinit** komutu ile çalışma seviyesini değiştirebilirsiniz. Çalışma seviyesi değiştiğinde, **init** /etc/inittab dosyası içinde konuyla ilgili satırı çalıştırır.

11.4. /etc/inittab içinde özel ayarlamalar

/etc/inittab dosyası **init**'in özel durumlara tepki vermesini sağlayacak özel yetenekler içerir. Bu özel nitelikler üçüncü alanlar içerisindeki özel kelime/kelime grupları tarafından belirlenir. Örneğin:

powerwait
: Enerji kesintisi durumunda, **init**'in sistemi kapatmasına izin verir. Burada bir kesintisiz güç kaynağı ve bu kaynağı takip edip sonuçları **init** programına bildiren bir yazılım kullandığınız varsayılmaktadır.

ctrlaltdel
: Kullanıcının klavyeden ctrl+alt+del tuşlarına basması durumunda **init** programının sistemi yeniden başlatmasını sağlar.

Bilgi

Sistem yöneticisi ctrl+alt+del tuşlarını istediği başka bir şeyi yapması veya hiçbir şey yapmaması için ayarlayabilir.

sysinit
: Sistem açılışında çalıştırılacak komut. Bu komut örneğin /tmp dizininin temizlenmesini sağlayabilir.

Yukarıdaki liste ayrıntılı bir liste değildir. Bu nedenle inittab kılavuz sayfasının incelenmesi faydalı olabilir.

11.5. Tek kullanıcılı kipte açılış

Tek kullanıcılı kip önemli bir çalışma seviyesidir. Bu kipte iken sadece sistem yöneticisi ve pek az sayıda süreç çalışır. Tek kullanıcılı kip, yönetici görevlerinin yerine getirilmesi için gereklidir. Örneğin; /usr dizini üzerinde **fsck** çalıştırmak gibi... Bu komutu çalıştırırken disk bölümünün bağlanmamış olması gerekmektedir, üstelik kok dosya sistemi için bütün sistem süreçleri öldürülmeden bu yapılamaz.

Çalışan bir sistem, **telinit** ile 1. çalışma seviyesi istenerek tek kullanıcılı kipe çevrilebilir. Açılış esnasında çekirdek komut satırından emergency veya single kelimeleri girilerek de, **init** programının öntanımlı açılış seviyesini çalıştırmaması gerektiği belirtilebilir (Çekirdek komut satırı sistemin açılışının nasıl yapıldığına bağlı olarak değişik şekillerde kullanılabilir).

Tek kullanıcılı kip; bazı zamanlar herhangi bir dosya sistemi bağlanmadan önce **fsck** komutunun el ile çalıştırılması için gerekli olabilmektedir. Örneğin, bozulmuş bir /usr bölümü üzerinde yapacağınız her işlem

onun iyice bozulmasına sebep olabilir. Bu nedenle mümkün olan en kısa süre içinde **fsck** komutunun çalıştırılması gerekli olabilir.

Şayet başlangıçta otomatik devreye giren **fsck** hata verirse, açılış betikleri **init**'i tek kullanıcılı kipte açılmaya zorlarlar. Bu **fsck** komutunun otomatik olarak onaramadığı bir dosya sisteminin kullanılmasını önlemeye yönelik bir tedbirdir. Bu tür bozulmalar gerçekte çok nadiren görülürler ve bozulmuş bir sabit disk veya deneysel bir çekirdek tarafından meydana getirilirler.

Bir güvenlik önlemi olarak; düzgün ayarlanmış bir işletim sistemi, tek kullanıcılı kipte kabuk programını çalıştırmadan önce, root parolasının girilmesini isteyecektir. Aksi taktirde, LILO'dan uygun bir komut satırıyla sisteme root gibi girmek çok kolay olacaktı. (Dosya sistemi problemleri nedeniyle `/etc/passwd` dosyası bozulmuşsa, girişte parola istenmeyecektir. Tabii ki böyle bir durumda elinizin altında bir açılış disketinin bulunması iyi olacaktır).

12. Kullanıcı Giriş ve Çıkışları

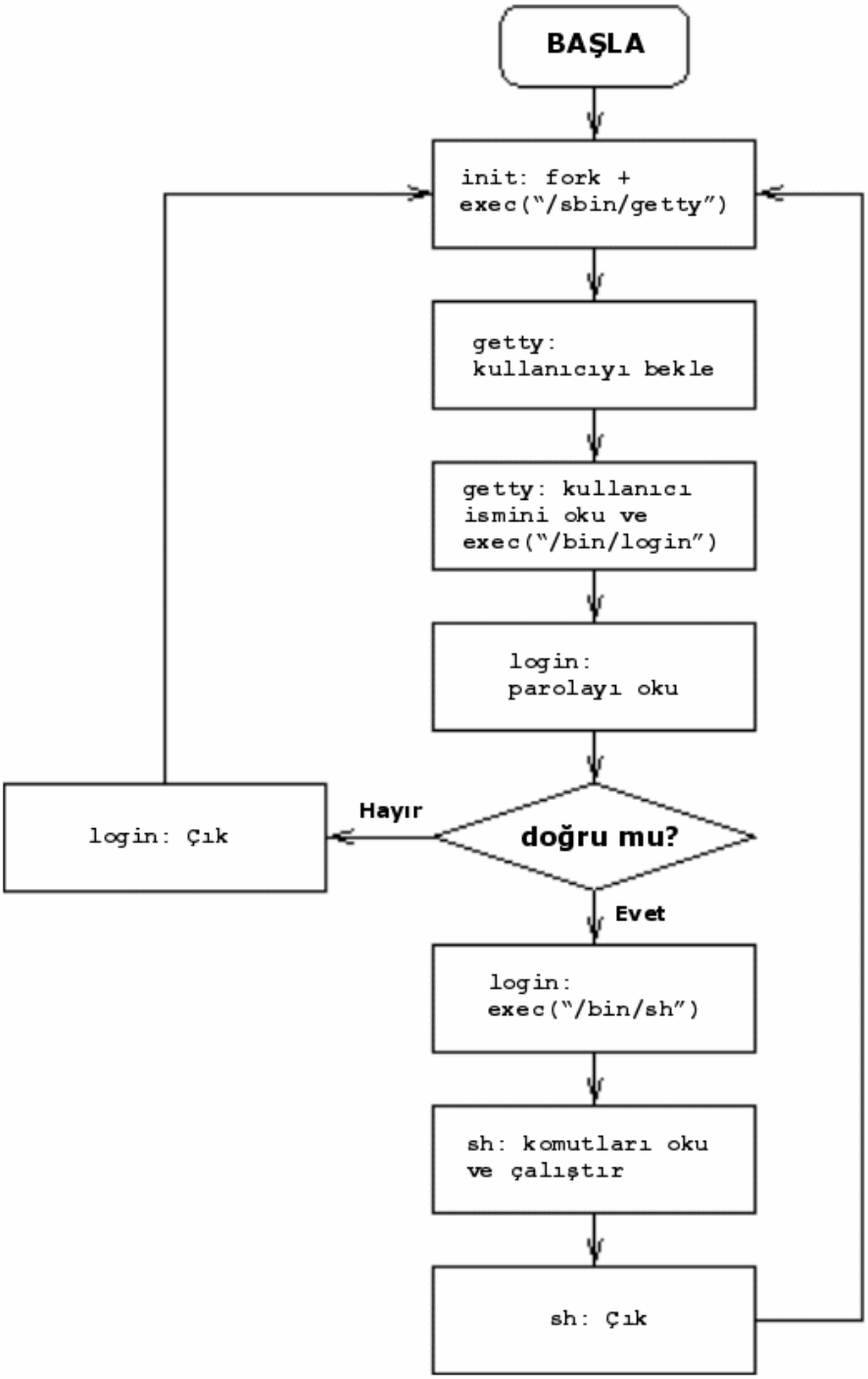
"benim gibi insanları üye olarak kabul eden klüplere girmekle ilgilenmiyorum." (Groucho Marx)

Bu bölümde bir kullanıcının sisteme girişi ve çıkışı esnasında neler olduğu açıklanmaktadır. Çeşitli etkileşimli artalan süreçleri, günlük dosyaları, ayar dosyaları ve benzeri şeyler ayrıntılı bir biçimde anlatılmıştır.

12.1. Uçbirim üzerinden giriş

Uçbirim üzerinden giriş (sayfa: 60) uçbirimler üzerinden bağlantıların nasıl yapıldığını göstermektedir. İlk önce, **init** uçbirim veya konsol üzerinden yapılacak bağlantıyı sağlamak için bir **getty** programını çalıştırır. **getty** uçbirimi dinler ve kullanıcının bağlantı için hazır olduğuna dair bir işaret göndermesini bekler. (Genellikle bu işaret kullanıcının klavyeden bir şeyler yazmasıdır). Bir kullanıcı tespit ettiği zaman, `/etc/issue` içerisinde bulunan "hoş geldin" mesajı ekrana yansıtılır ve bir kullanıcı adı girilmesini ister. Kullanıcı ismi alındıktan sonra **login** programını çalıştırır. **login** kullanıcı adını bir parametre olarak alır ve bir parola girilmesini bekler. Şayet bu bilgiler uyuşursa **login** programı bu kullanıcı için önceden tanımlanmış olan kabuğu çalıştırır, aksi taktirde süreci durdurur ve çıkar. (Belki de kullanıcıya yeni bir kullanıcı adı ve parolası girmek için bir şans daha verir). **init**, sürecin tamamlandığı uyarısını aldığında uçbirim için yeni bir **getty** programının çalıştırılmasını sağlar.

Şekil 7. Uçbirim üzerinden giriş



init, **getty**, **login** ve kabuk arasındaki etkileşim

Unutmayın ki **init**'in `fork` sistem çağrısını kullanarak yarattığı süreç bir tanedir. **getty** ve **login**, süreçte çalışan program ile yer değiştirilmektedirler (`exec` sistem çağrısı yardımıyla).

Seri hatlar üzerindeki kullanıcıları tespit edebilmek için ayrı bir programa ihtiyaç vardır, çünkü bu uçbirimlerin etkin hale geldiğini tespit etmek biraz daha karışıktır. **getty** parametrelerin çağrıdan çağrıya değiştiği modem bağlantıları için çok önemli olan hız ve diğer bağlantı ayarlarına kendini uyarlayabilir.

Sevapları ve günahları ile, kullanımda bulunan pek çok çeşit **init** ve **getty** programları vardır. Sisteminizde bulunan sürümünü ve diğerlerini öğrenmek iyi bir fikir olabilir (Bunun için Linux Software Map – Linux Yazılım Haritası kullanılabilir). Sisteme modem üzerinden erişim etkin değilse **getty** hakkında endişelenmenize gerek

kalmaz ama **init** önemini korur.

12.2. Ağ üzerinden giriş

Aynı ağ üzerindeki iki ayrı bilgisayar, genellikle tek bir fiziksel kablo üzerinden bağlanmıştır. Bu bilgisayarlar ağ üzerinden haberleşirken, bu haberleşme işleminde kullanılan ve her bir bilgisayarda ayrı ayrı bulunan programlar, bir sanal bağlantı üzerinden birbirlerine bağlanmışlardır. Bu bir çeşit hayali kablodur. Bu sanal bağlantı üzerinde bulunan her hangi bir bilgisayardaki programlar etkin durumda kaldığı sürece kendi kablosu üzerinde tekel konumuna sahip olur. Bununla beraber bu bağlantılar hayali olduğu için her iki bilgisayarın işletim sistemleri pek çok sanal bağlantıyı aynı fiziksel kablo üzerinde paylaştırabilir.

Bu yolla çeşitli programlar, diğer iletişimler ile uğraşmak zorunda kalmadan, tek bir kablo üzerinden iletişim sağlayabilirler. Aynı kablo üzerinde çeşitli bilgisayarlara sahip olmak mümkün olmasına rağmen, bu iki bilgisayarın kurduğu sanal hat dışında kalan bilgisayarlar, parçası olmadıkları bu iletişim işini yok sayarlar.

Bu gerçeğin oldukça karmaşık ve soyut ötesi bir açıklamasıdır. Belki de öyledir, ama bununla birlikte, ağ bağlantılarının normal bağlantılardan neden değişik olduğunu, daha kolay anlaşılır olmasını sağlamaktadır. Değişik bilgisayarlar üzerindeki programlar birbirleri ile iletişime geçmek istedikleri zaman sanal bağlantılar kurulur. Birağ üzerindeki her hangi bir bilgisayardan diğerine bağlanabilmek yöntem olarak mümkün olduğu için çok fazla sayıda potansiyel sanal iletişim mevcuttur. Bu nedenledir ki her potansiyel bağlantı için ayrı bir **getty** programı başlatmak hiç de pratik olmayacaktır.

Bütün ağ bağlantılarını denetleyen (ve **getty**'ye karşılık gelen) **inetd** isimli tek bir süreç vardır. Gelen bir ağ bağlantı sinyali aldığı zaman, bu tek bağlantıyı denetim altında tutmak için yeni bir süreç başlatır. Orjinal süreç bekler ve olası yeni bağlantıları dinlemeye devam eder.

Ağ bağlantıları için birden fazla iletişim protokolü olduğundan dolayı bu iş biraz karışıktır. En önemli iki tanesi **telnet** ve **rlogin**'dir. Bu bağlantılara ek olarak, FTP, Gopher, http gibi, yapılması muhtemel diğer bağlantılar da mevcuttur. Belirli bir bağlantı türü için ayrı bir süreç yaratmak etkisiz olacaktır. Bu nedenle, bağlantı türlerini tanımlayabilecek ve doğru servislerin çalışması için doğru programları yükleyebilecek tek bir dinleyici kullanılır. Bu dinleyici **inetd** olarak adlandırılır. Daha ayrıntılı bilgiyi Linux Ağ Yöntecisinin Kılavuzu[B36] isimli e–kitapta bulabilirsiniz.

12.3. `login` ne yapar?

login programı kullanıcı adı ile parolanın uyuşup uyuşmadığını kontrol eder ve kabuğun başlatılması, seri hatlara izin verilmesi gibi ayarların yapılmasını sağlayarak kullanıcıya gerekli ortamı hazırlar.

Başlangıç ayarlarının bir parçası da `/etc/motd` içindeki günün mesajının ve elektronik posta kutusunun kontrol sonuçlarının ekrana yansıtılmasıdır. Bunlar kullanıcının ev dizini içerisinde `.hushlogin` isimli bir dosya oluşturularak engellenebilir.

Şayet `/etc/nologin` isimli bir dosya mevcutsa kullanıcı girişleri engellenir. Bu dosya genellikle **shutdown** benzeri programlar tarafından yaratılır. **login** programı önce bu dosyayı kontrol eder ve şayet bulursa kullanıcıların giriş isteklerini reddeder. Şayet bu dosya var ise, **login** kapanmadan önce, bu dosyanın içeriğini uçbirime çıktılar.

login bütün hatalı bağlantı girişimlerini bir sistem günlük kayıt dosyasında tutar (**syslog** üzerinden). Ayrıca root kullanıcı tarafından yapılan bütün bağlantıların kayıtlarını tutar. Bunlar, davetsiz misafirlerin izini sürmek açısından oldukça faydalı olabilirler.

Sisteme giriş yapmış olan (ve halen sistemde bulunan) herkes `/var/run/utmp` içinde listelenir. Bu dosya sistem kapatılana veya yeniden başlatılana kadar geçerlidir; sistem tekrar açıldığında bu dosyanın içeriği silinmiş olacaktır. Bu dosya her kullanıcı ve uçbirimini (veya ağ bağlantısını) ve bazı yararlı bilgileri listeler.**who**, **w** ve diğer benzeri komutlar sistemde bağlı bulunan kullanıcıları listelemek için `utmp` içine bakarlar.

Bütün başarılı bağlantılar /var/log/wtmp içinde kayıtlıdır. Bu dosya sınırsızca büyüyecektir. Bu nedenle düzenli olarak temizlenmesi gerekir. Bunun için haftalık temizlik yapacak bir **cron** işi düzenleyebilirsiniz. **last** komutu wtmpiçerisini tarar.

Hem utmp hem de wtmp ikilik biçemdedir ve ne yazık ki özel programlar olmadan incelemeye uygun değillerdir. (Daha ayrıntılı bilgi için utmp kılavuz sayfasına bakınız.)

12.4. X ve xdm

Birileri, X aletleri xdm[14] üzerinden bağlanır gibi birşeyler karalayabilir mi; ayrıca; xterm –ls filan?

12.5. Erişim denetimi

Kullanıcı veritabanı geleneksel olarak /etc/passwd dosyası içerisinde saklanır. Bazı sistemler gölge (shadow) parolalar kullanırlar ve parolaları **/etc/shadow** dosyasına yönlendirirler. NIS kullanan hesapları paylaşan bilgisayarların bulunduğu alanlarda veya kullanıcı veri tabanını saklamak için diğer yöntemleri kullananlar; kullanıcı veritabanını merkezi bir yerden tüm diğer bilgisayarlara otomatik olarak kopyalıyor olabilirler.

Kullanıcı veritabanları sadece kullanıcı parolalarını değil, kullanıcılar hakkında; ev dizinleri, isimleri ve kullandıkları kabuklar gibi bazı ek bilgileri de içerirler. Bu diğer bilgiler topluma açık olmalıdır, böylece herkes okuyabilir. Ama parolalar şifreli bir şekilde depolanırlar. Bu, sistemde kayıtlı olmayan ama bir şekilde şifrelenmiş parolalara ulaşıp ve onları tahmin etmede çeşitli kriptografi yöntemleri deneyecek kişilere bir engel oluşturmak üzere gölge parolalar şifreli olarak sadece root kullanıcısının okuyabildiği ayrı bir dosyada saklanır. Bununla birlikte, gölge parolaların desteklenmediği bir sisteme sonradan aktarılması, zorluk çıkarabilir.

Şifreli veya değil bütün parolaların güvenilir ve kolay tahmin edilemez olduğundan emin olmalıyız. **crack** programı parolaları kırmak için kullanılır; tanımından bulunabilecek parolalar iyi değildir. **crack** izinsiz misafirlerce çalıştırılabileceği gibi, sistem yöneticisi tarafından kötü parolaları bulmak için de çalıştırılabilir. **passwd** programı uygun parolalar kullanmamız için bizi zorlar; bu gerçekte CPU turlarından daha etkilidir. Çünkü bir parolayı kırmak için pek çok hesaplamalar yapmak gerekmektedir.

Kullanıcı gruplarının veri tabanları /etc/group içerisinde tutulur; gölge parolalar kullanan sistemlerde bu /etc/shadow.group içerisinde olur.

Root genellikle pek çok uçbirimden ve ağ üzerinden bağlanamaz, /etc/securetty dosyası içinde listelenmiş uçbirimleri kullanmak zorundadır. Bu, adı geçen uçbirimlerden birisine fiziksel erişim kurulmasını zorunlu kılar. Bununla birlikte her hangi bir uçbirim veya ağ üzerinden normal bir kullanıcı gibi bağlanıp, root durumuna geçmek için **su** komutunu kullanmakta olasıdır.

12.6. Kabuk başlangıcı

Bir etkileşimli bağlantı kabuğu başladığı zaman; kabuk, daha önceden tanımlanmış bir veya daha fazla dosyayı çalıştırır. Değişik kabuklar değişik dosyaları çalıştırır. Ayrıntılı bilgi için her kabuğu kendi belgelerine bakmalısınız.

Pek çok kabuk ilk önce bazı genel dosyaları çalıştırır; örneğin: Bourne Shell (**/bin/sh**) ve türevleri /etc/profile ve ek olarak kullanıcı ev dizini içerisindeki .profile dosyasını çalıştırırlar. /etc/profile dosyası root kullanıcıya; kullanıcılar için genel çalışma ortamı hazırlama, PATH ortam değişkeni ile yerel komut dizinlerini diğerlerinin yanına ekleme gibi ayarlamalar yapma olanağı sağlar. Diğer yandan .profile dosyası ise kullanıcının kendi zevkine göre çalışma ortamı hazırlamasını sağlar ve şayet gerekiyorsa öntanımlı olarak atar.

13. Kullanıcı Hesaplarının Yönetimi

> "sistem yöneticiliği ile uyuşturucu satıcılığı arasındaki benzerlik: her ikisinde de ölçü biriminin kilo olması ve kullanıcıların bulunmasıdır." (eski bir bilgisayar şakası)

Bu bölümde yeni kullanıcı hesaplarının açılması, bu hesapların özelliklerinin nasıl düzenleneceği ve bu hesapların nasıl kapatılacağı açıklanmaktadır. Değişik Linux dağıtımları bunları yapmak için değişik araçlar kullanabilir.

13.1. Hesap nedir?

Bir bilgisayar çeşitli farklı kişiler tarafından kullanıldığı durumlarda, kullanıcılar arasında ayrım yapmak gerekli olmaktadır. Bu sayede kişiye özel dosyalar kişisel hale getirilebilir. Pek çok bilgisayarda olduğu gibi, tek bir kullanıcı varsa bile bunu yapmak faydalı olabilir. Sonuçta, her kullanıcı için benzersiz bir kullanıcı ismi verilir ve herkes bağlanmak için kendi ismini kullanır.

Bununla birlikte kullanıcı için bir isimden daha fazlası mevcuttur. Bir hesap, kullanıcının ismini, dosyalarını, kaynaklarını ve ona ait her şeyi temsil eder. Bu terim bankacılığa aittir ve bir ticari sistemde her hesapta biraz para vardır. Ve bu para kullanıcıların sisteme yaptığı baskıya göre değişen hızlarda eriyip gider. Örneğin; disk alanı günler ve Mb'lara göre bir fiyata sahip olabilir ve işlem süresinin de saniyelere göre bir fiyatı olabilir.

13.2. Bir kullanıcının oluşturulması

Linux çekirdeği, kullanıcıları sadece basit sayılar olarak algılar. Her kullanıcı için tam sayılardan oluşan benzersiz bir tanımlama yapılmıştır, çünkü bir bilgisayar için sayılarla uğraşmak harflerden oluşan isimler ile uğraşmaktan daha kolaydır. Bunlara kullanıcı kimliği (uid) ve *grup kimliği* (gid) denir. Çekirdek dışında ayrı oluşturulmuş veritabanlarında her bir kimlik için *kullanıcı ismi* olarak tanımlanan metinsel adlar tutulur. Ve tabii ki bu veritabanı bazı ek bilgileri de içerir.

Bir kullanıcı oluşturmak için, kullanıcı veritabanına bir kullanıcı hakkında bilgiler eklemelisiniz ve bu kullanıcı için bir ev dizini yaratmalısınız. Ayrıca kullanıcıyı eğitmek ve ona başlangıç için uygun bir ortam hazırlamak da gerekebilir.

Pek çok Linux dağıtımı kullanıcı hesapları oluşturmak için gereken programlar ile gelirler. Bu programların kullanışlı sürüm ve çeşitleri bulunur. Bunlardan komut satırı uygulaması olan ikisi **`adduser`** ve **`useradd`** dışında bir GUI araç da bulunabilir. Hangi program olursa olsun sonuçta, geriye el ile yapılması gereken pek az iş kalır. Ayrıntılar pek çok ve karmaşık olsa da bu programlar önemsiz görünen her şeyi yaparlar. Bununla birlikte *Kullanıcıların elle oluşturulması* (sayfa: 65) bölümünde bütün bu ıvır zıvır şeylerin el ile nasıl yapılacağı açıklanmaktadır.

13.2.1. `/etc/passwd` ve diğer bilgi dosyaları

Bir Unix işletim sistemindeki en basit veri tabanı; geçerli kullanıcıların ve onların birleştirilmiş bilgilerinin listelendiği `/etc/passwd` bir metin dosyasıdır. Bu dosya parola dosyası olarak anılır. Bu dosyada her kullanıcı için bir satır ve her satırda iki nokta üstüste işaretleri ile ayrılmış 7 sütun bulunur:

- Kullanıcı ismi.
- Şifrelenmiş olarak parola.
- Kullanıcı kimliği (bir numara).
- Grup kimliği (bir numara).
- Kullanıcının gerçek ismi ve heapla ilgili diğer açklamalar
- Ev dizini.
- Bağlantı kabuğu (başlangıçta çalışacak kabuk)

`passwd` kılavuz sayfasında bu dosya biçemi hakkında daha ayrıntılı bilgi bulunmaktadır.

Sistemdeki her hangi bir kullanıcı bu parola dosyasını okuyabilir ve böylece diğer kullanıcıların adlarını öğrenebilir. Bunun anlamı, ikinci bölümde bulunan parolaların da herkes tarafından görülebileceğidir. Bu parolalar şifreli olduğu için, teorik olarak ortada bir problem yoktur. Bununla beraber bu şifreler kırılabilir, özelliklede bir sözlükte bulunabilecek bir şeyse veya çok kısaysa. Bu nedenle, parolaları bir parola dosyasında saklamak iyi bir fikir sayılmaz.

Pek çok Linux sistemi gölge parolalar kullanır. Bu parola saklamanın alternatif bir yoludur: bu sistemde parolalar şifrelenmiş bir şekilde, sadece root tarafından okunabilen, `/etc/shadow` dosyasında tutulurlar. `/etc/shadow` dosyası ikinci alanda sadece özel bir işaret ihtiva eder. Kullanıcı kimliğini kanıtlaması gereken herhangi bir program setuid'dir ve bu nedenle gölge parola dosyasına erişimi vardır. Normal programlar diğer alanları kullanırlar ve parolalara erişemezler.

13.2.2. Sayısal kullanıcı ve grup kimliklerinin seçilmesi

Pek çok sistemde sayısal kullanıcı ve grup kimliklerinin ne olduğu pek de önemli değildir, fakat şayet bir ağ dosya sistemi (NFS) kullanıyorsanız, bütün sistemlerde aynı uid ve gid'ye sahip olmanız gerekmektedir. (uid=user id – kullanıcı kimliği, gid=group id – grup kimliği). Bu NFS'nin kullanıcıları sayısal kullanıcı kimlikleri ile tanımlamasından kaynaklanır. Şayet NFS kullanmıyorsanız, hesap oluşturma araçlarının bunları otomatik olarak sizin yerinize seçmesine izin verebilirsiniz.

Şayet NFS kullanıyorsanız, kullanıcı bilgilerinin birbirine uyması için bir mekanizma bulmak zorundasınız. Alternatif yollardan birisi NIS sistemidir (XXX network–admin–guide'a bakabilirsiniz).

Bunun yanında, sayısal kullanıcı kimliklerinin tekrar kullanılmasını önlemeyi denemelisiniz, çünkü kullanıcı kimliğinin yeni sahibi, eski sahibinin dosyalarına, postalarına veya her neyse onlara ulaşabilir.

13.2.3. Ortamın hazırlanması: `/etc/skel`

Yeni bir kullanıcı için ev dizini yaratıldığı zaman, bu `/etc/skel` dizinin içindeki dosyalarla oluşturulur. Sistem yöneticisi, kullanıcılar için daha hoş öntanımlı ortamlar sağlamak için, `/etc/skel` içinde dosyalar yaratabilir. Örneğin bir `/etc/skel/.profile` dosyası yaratıp, içinde, yeni kullanıcılar için EDITOR ortam değişkenininde belirterek daha kullanıcı dostu metin düzenleyiciler ayarlayabilir.

Bununla beraber, `/etc/skel` dizinini mümkün olduğunca küçük tutmak daha iyidir, çünkü, sıradaki, mevcut kullanıcıların dosyalarını güncellemesinin imkansızlaşması olabilir. Örnek: bu kullanıcı dostu metin düzenleyicinin adı değiştirilirse, bütün mevcut kullanıcılar kendi `.profile` dosyalarını düzenlemek zorunda kalacaklardır. Sistem yöneticisi bir betik kullanarak bunu otomatik olarak yapmayı deneyebilir, fakat bu yöntemle birilerinin dosyalarının bozulması neredeyse kesindir.

Mümkün olduğunca, genel ayarları genel dosyalarda yapmak daha iyi olacaktır, örneğin `/etc/profile` dosyası gibi. Bu yolla, hiç kimsenin dosyalarına ve ayarlarına zarar vermeden güncelleme yapmak mümkün olacaktır.

13.2.4. Kullanıcıların elle oluşturulması

Kullanıcıları elle oluşturmak için aşağıdaki adımları izleyin

- `/etc/passwd` dosyasını **`vipw`** ile düzenleyin ve yeni bir hesap için yeni bir satır ekleyin. Sözdizimleri konusunda dikkatli olun. *BİR SATIRI BİR METİN DÜZENLEYİCİ İLE DOĞRUDAN DOĞRUYA ASLA DÜZENLEMEYİN!* **`vipw`** dosyayı kilitleyeceği için diğer komutlar aynı anda güncelleme yapmayı deneyemezler. Parola alanını '`*`' yapmalısınız, böylece sisteme bağlanmak imkansız olur.
- Şayet yeni bir grup oluşturmanız gerekiyorsa, aynı yöntemle `/etc/group` dosyasını **`vigr`** ile düzenleyin.

- **mkdir** kullanarak, kullanıcı ev dizinini yaratın.
- /etc/skel dizininin içeriğini yeni ev dizinine kopyalayın.
- **chown** ve **chmod** ile sahipleri ve izinleri sabitleyin. -R seçeneği oldukça kullanışlıdır. Doğru izinler durumdan duruma göre biraz farklılıklar gösterir fakat genellikle aşağıdaki komutlar doğru işlemleri yaparlar:

```
# cd /home/kullanici
# chown -R kullanici.grup .
# chmod -R go=u,go-w .
# chmod go= .
```

- **passwd** ile parolayı belirleyin.

Son adımda parolayı düzenledikten sonra kullanıcı hesabı çalışmaya başlar. Her şey bitmeyen parolayı düzenlemeyin; yoksa kullanıcı bir dikkatsizlik sonucu, siz hala dosyaları kopyalarken, sisteme bağlanabilir.

Bazen hiç kimse tarafından kullanılmayan "aptal" hesaplar oluşturmak gerekebilir. Örneğin; anonim bir FTP sunucusunu düzenleyebilmek için ftp adında bir kullanıcı hesabı açmanız gerekir (böylece herkes,bir kullanıcı hesabı açmadan buradan dosya indirebilir). Bu gibi durumlarda, genellikle parola ayarlamak gerekmez. Gerçekte, bunu yapmamak daha iyidir, böylece bu hesabı, root herhangi bir kullanıcı haline gelebildiğinden root dışında hiç kimse kullanamaz.

13.3. Kullanıcı özelliklerinin değiştirilmesi

Bir hesabın çeşitli özelliklerini (/etc/passwd dosyasında konuyla alakalı alanları) değiştirmek için bir kaç komut vardır:

chfn
: Kullanıcının gerçek ismini değitirir.

chsh
: Kullanıcı girişinden sonra çalıştırılan kabuğu değiştirir.

passwd
: Kullanıcı parolasını değiştirir.

Süper kullanıcı herhangi bir hesabın özelliklerini değiştirmek için bu komutları kullanır. Normal kullanıcılar ise sadece kendilerine ait özellikleri değiştirebilirler. Bazen **chmod** komutunu kullanarak bu komutları normal kullanıcılar için yasaklamak gerekebilir. Özellikle acemi kullanıcıların çok olduu ortamlarda.

Diğer görevler el ile yapılmalıdır. Örneğin: kullanıcı ismini değiştirmek için, doğrudan /etc/passwd içinde düzenleme yapmak zorundasınız (**vipw** ile). Benzer şekilde, bir kullanıcıyı bir gruba eklemek ya da çıkartmak için /etc/group dosyasını düzenlemelisiniz (**vigr** ile). Çok nadiren bu tür işler ile uğraşmak zorunda kalırsınız, bununla beraber, bu işleri bir uyarı eşliğinde yapmalısınız. Örneğin: şayet siz kullanıcı adını değiştirirseniz, posta kutusuna bir takma ad tanımlanmadığı sürece, kullanıcıya gönderilen postalar ulaşılmaz olur.

13.4. Bir kullanıcının silinmesi

Bir kullanıcıyı sistemden kaldırmak için, ilk önce bu kullanıcıya ait ne varsa: dosyaları, posta kutuları, yazdırma işleri, **cron** ve **at** işleri ile kullanıcıya ait diğer her şey silinmelidir. Daha sonra /etc/passwd ve /etc/group dosyalarından ilgili satırlar kaldırılır (kullanıcının dahil olduğu bütün gruplardan ismini silmeyi unutmayın). Kullanıcının sisteme bağlanıp, silme işleri sırasında hesabı kullanmasını önlemek için; silme işlemlerine

başlamadan önce, hesabı kapatmak daha iyi olabilir (Aşağıya, *Bir kullanıcı hesabının geçici olarak kapatılması* (sayfa: 67) bölümüne bakınız).

Unutmayın ki kullanıcıların kendi ev dizinleri dışında da dosyaları olabilir. Bunları bulmak için **find** komutunu kullanın.

```
# find / -user username
```

Bunun yanında, üstteki komut, şayet çok büyük bir diskiniz varsa, oldukça uzun bir zaman alacaktır. Şayet ağ diskleri de bağlıysa çok dikkatli olun, sunucuyu veya ağı değersiz bir çöp yığınına çevirmeniz işten bile değildir.

Bazı Linux dağıtımları bu iş için **deluser** ya da **userdel** gibi özel komutlarla gelirler. Ancak elle yapmak da kolaydır ve bu komutlar herşeyi yapmayabilir.

13.5. Bir kullanıcı hesabının geçici olarak kapatılması

Bazı zamanlar kimi kullanıcı hesaplarını, silmeden kapatmak gerekir. Örneğin; kullanıcı aidatını ödememiş olabilir veya sistem yöneticisi bir yabancının bu hesaba ait parolayı ele geçirdiğinden şüpheleniyor olabilir.

Bir hesabı kapatmanın en iyi yolu, o hesaba ait kabuğu, sadece bir mesaj veren özel bir program ile değiştirmektir. Böylece, hesaba bağlanmak isteyen her hangi birisinin bu eylemi durdurulacak ve nedeni kendisine bildirilecektir. Mesaj, kullanıcıya problemi çözmek üzere sistem yöneticisine başvurmasını söyleyebilir.

Kullanıcı ismi veya parolanın değiştirilmesi de mümkündür ama bu durumda kullanıcı neler olup bittiğini anlayamayacaktır. Kafası karışan kullanıcı daha çok uğraşacaktır.

Özel bir program yaratmanın en basit yolu bir "tail betiği" yazmaktır:

```
#!/usr/bin/tail +2
Bu hesap güvenlik nedeniyle askıya alınmıştır.
Hesabınızı açtırmak için lütfen müşteri hizmetleriyle görüşünüz.
```

İlk iki karakter (‘#!’) çekirdeğe bu satırın devamın bir komut olduğunu ve bu komutun bu dosyayı yorumlamak için çalıştırılması gerektiğini söyler. Bu durumdaki **tail** komutu ilk satır hariç her şeyi standart çıktıya gönderir.

Şayet aliveli isimli kullanıcısının hesabında bir güvenlik açığından şüpheleniliyorsa sistem yöneticisi şuna benzer bir şeyler yapmalıdır:

```
# chsh -s /usr/local/lib/no-login/security aliveli
# su - tester
Bu hesap güvenlik nedeniyle askıya alınmıştır.
Hesabınızı açtırmak için lütfen müşteri hizmetleriyle görüşünüz.
#
```

Burada **su** komutunun kullanılmasındaki amaç; değişikliklerin düzgün bir şekilde çalışıp çalışmadığını görmektir.

Tail betikleri ayrı bir dizinde tutulmalıdır, böylece normal kullanıcı komutları ile karışması önlenebilir.

14. Yedek Alma

> Donanım belirlenmezciye göre güvenilirdir.
> Yazılım belirlenimciye göre güvenilir değildir.
> Toplum belirlenmezciye göre güvenilir değildir.
> Doğa belirlenimciye göre güvenilirdir.

Bu bölüm neden, nasıl ve ne zaman yedek almanız ve geri yüklemeniz gerektiğini anlatmaktadır.

14.1. Yedeklemenin önemi üzerine

Bilgileriniz değerlidir. Onları tekrar meydana getirmek; zaman, para veya en azından kişisel keder ve göz yaşına mal olabilir. Şayet bu bilgiler bazı deneylerin sonucu ise onları tekrar meydana getirmek mümkün olmayabilir. Bilgileriniz bir yatırım olduğuna göre, onları korumalı ve kaybetmemek için bazı adımlar atmalısınız.

Temel olarak bilgi kayıplarının 4 ana sebebi vardır: donanım arızaları, yazılım hataları, insandan kaynaklanan olaylar ve doğal afetler. Modern donanımların oldukça güvenilir olmasına rağmen, halen kendiliğinden arızalar çıkabilecekmiş gibi görünmektedir. Bilgi depolama donanımlarının en kritik parçası sabit disktir, ki bu diskler, elektromanyetik gürültü ile dolu bir dünya içine tam olarak kalabilen manyetik alanlara bel bağlamaktadır. Modern yazılımlar pek güvenilir gözükmese bile, kaya sertliğindeki bir program istisnadır, kural değil. İnsanlar kesinlikle güvenilmezdir, sırf keyif olsun diye veya bir amaç uğruna bilgilerinizi yok edebilirler ve yahut bir hata yapabilirler. Doğa kötü değildir, fakat her şey iyi giderken birden öfke kusabilir. Her şeyi hesaba katınca, her hangi bir şeyin doğru çalışmasının hiç de küçük bir mucize olmadığını görürüz.

Yedekleme bilgi yatırımımızı korumanın bir yoludur. Bilgilerimizin çeşitli kopyalarına sahip olursak, her hangi bir kopyanın bozulması bizim için problem teşkil etmez (en fazla yedeklerimizden yüklememizi gerektirir).

Düzenli olarak yedeklemek çok önemlidir. Fiziksel dünyaya ait her hangi bir şey gibi, yedeklerde yakında veya ileri bir tarihte hata verebilirler. İyi bir yedekleme yapmanın bir parçası da yedeklerin çalıştığından emin olmaktır; yedeklerinizin çalışmadığını görmek istemezsiniz her halde. Hasara tuz biber olarak, yedekleme işlemi esnasında bir hata meydana gelebilir; elinizde sadece yarım bir yedek varken sizi ortada bırakabilir, tek bir yedekleme aracınız vardır ve bozulabilir. Veya geri yüklemeyi denerken fark edersiniz ki: 15000 kullanıcılı alandaki önemli bir kullanıcı veritabanını yedeklememişsinizdir. Hepsinin en iyisi de; bütün yedeklerinin mükemmel bir şekilde çalışıyordur fakat sonuncu teyp sürücünüzün içine bir kova su dolmuş olabilir.

İş yedeklemeye geldiğinde, paranoya bu işin mayasında vardır.

14.2. Yedekleme ortamının seçimi

Yedekleme işindeki en önemli karar; yedekleme aracını seçmektir. Fiyatını, güvenilirliğini, hızını, kullanılabilirliğini ve bulunabilir olup olmadığını hesaba katmak zorundasınız.

Fiyat önemlidir, çünkü; tercihen, ihtiyacınız olan bilgilerden daha fazlasını yedeklemek isteyebilirsiniz. ucuz bir araç genellikle bir zorunluluktur.

Bozulmuş bir yedek, yetişkin bir adamı bile ağlatabileceği için: güvenilirlik çok önemlidir. Bir yedekleme aracı, bilgileri her hangi bir bozulma olmadan yıllarca saklayabilmelidir. Bir denetim gerektirmediği sürece yedekleme işinin kaç saate mal olduğu önemli değildir. Diğer yandan, bilgisayar boş durduğu sürece yedekleme işi başarılamayacaktır, bu nedenle hız da ayrı bir noktadır.

Bulunabilirlik kesinlikle çok önemlidir: piyasada olmayan bir yedekleme aracını kullanamazsınız. Gelecekte de kullanılabilecek, sadece sizin değil diğer insanların bilgisayarlarında da, yedekleme aygıtı seçmeniz önemlidir. Aksi taktirde her hangi bir problemden sonra yedeklerinizi yükleyememek gibi bir sorunla karşılaşabilirsiniz.

Kullanılabilirlik: ne kadar sıklıkla yedekleme yapılacağını belirleyen temel faktördür. En kolay yedekleme yapma yöntemi en iyisidir. Bir yedekleme aracının kullanımı zor ve sıkıcı olmamalıdır.

Geleneksel alternatifler teypler ve disketlerdir. Disketler: ucuz, güvenilir, kolay bulunabilir, kullanışlı, yavaş ama büyük boyutlu bilgileri saklamada yetersiz araçlardır. Teypler: ucuza göre biraz pahalı, güvenilir, nispeten hızlı, bulunabilir ve kullandığınız bandın boyutuna göre oldukça kullanışlı aygıtlardır.

Diğer alternatiflerde vardır. Genellikle pek bulunabilir olmasalar da, şayet bu sizin için problem değilse, diğer yollardan daha iyidirler. Örneğin, manyetik optik diskler, disketlerin ve teyplerin iyi yönlerini bir arada bulundururlar: hızlı, rasgele erişimli, tek bir dosyayı oldukça hızlı geri yükleyebilen, büyük boyutlu araçlardır.

14.3. Yedekleme aracının seçimi

Yedekleme yapmak için pek çok araç vardır. Yedekleme yapmak için kullanılan geleneksel Unix araçları **tar**, **cpio** ve **dump**'dır. Ek olarak pek çok üçüncü parti yedekleme yazılımları da mevcuttur. Yedekleme aygıtının seçimi, yedekleme aracının seçimini de etkileyebilir.

tar ve **cpio** türdeştirler ve yedeklemeye bakış açıları büyük benzerlikler içerir. Her ikisi de teyplere yedeklemek ve onlardan geri alabilmek yeteneklerine sahiptirler. Her ikisi de hemen hemen bütün saklama ortamlarını kullanmaya muktedirlerdir. Çünkü çekirdek aygıt sürücüleri alt seviye aygıtlarıyla ilgilenir ve onları kullanıcı seviyeli aygıtlarmış gibi davranmaya yönlendirirler. Bazı Unix dağıtımlarında **tar** ve **cpio** özel türdeki bazı dosyalarla bazı problemler çıkarabilirler (sembolik bağlar, aygıt dosyaları gibi) ama tüm Linux dağıtımlarında bu programların doğru çalışıyor olması gerekmektedir.

dump biraz farklıdır, dosya sistemi üzerinden okuma yapmak yerine, doğrudan doğruya dosya sistemini okur. Ayrıca özellikle yedekleme için yazılmıştır; **tar** ve **cpio** ise aslında dosyaları arşivlerler ama yedeklemeler için de iyi çalışırlar.

Dosya sistemini doğrudan doğruya okumanın bazı getirileri vardır. Dosyaların zaman izlerini etkilemeden onları yedeklemeyi mümkün kılar; **tar** ve **cpio** için, dosya sistemini salt okunur olarak bağlamak zorundasınız. Şayet her şeyin yedeklenmesi gerekiyorsa, dosya sisteminin doğrudan doğruya okunması daha etkili bir yoldur çünkü disk kafaları daha az hareket etmek zorunda kalırlar. Buradaki temel götürü ise yedeklerin sadece bir tek dosya sistemine özel hale getirilmiş olmasıdır. Linux **dump** programı sadece ext2 dosya sistemini tanır.

dump ayrıca, aşağıda tartışacağımız yedekleme seviyelerini desteklemektedir. **tar** ve **cpio**'da bunu diğer aygıtları etkinleştirmek yolu ile yaparız.

Üçüncü parti yazılımlar arası bir karşılaştırma bu kitabın konusu dışındadır. Linux Yazılım Haritası, bu programların freeware olanlarının pek çoğunu listelemektedir.

14.4. Basit yedekleme

basit yedekleme şeması her şeyin bir seferde yedeklenmesi ve daha sonra önceki yedekleme işleminden beri meydana gelmiş olan değişikliklerin yedeklenmesidir. Birinci yedekleme tam yedek, sonrakiler *fark yedekleri* olarak adlandırılır. Tam yedekler, fark yedeklerinden daha fazla işçilik gerektirirler, çünkü; teybe yazılması gereken daha çok bilgi vardır ve genellikle tek teyp veya diskete sığmazlar. Fark yedeklerinden geri yüklemek tam yedeklerden geri yüklemekten daha çok zamana mal olurlar. Geri yükleme eniyilenebilir, bu sayede daima bir önceki tam yedeklemeden itibaren olan her şeyleri yedeklersiniz. Bu yolla yedekleme biraz daha fazla iş ister fakat geri yükleme sırasında bir tam yedek ve bir fark yedeğinden daha fazlasını yüklemek gerekmez.

Şayet her gün yedek almak istiyor ve altı adet bandınız varsa; birinci bandı birinci tam yedek için ayırın (diyelim ki Cuma günü), 2 ile 5 arasındaki bantları her bir gün için (pazartesi–Perşembe arası) fark yedeklerinde ve altıncı banda yeni bir tam yedekleme (ikinci Cuma günü) yapın. Ve daha sonra 2–5 arasındaki bantları fark yedeklemesi için kullanmaya devam edin. Tam yedekleme yaparken her hangi bir şey olmasın diye, yeni bir tam yedekleme bandınız olana kadar 1. bant üzerine yazmayabilirsiniz. Altıncı banda tam yedekleme yaptıktan sonra, birinci tam yedekleme bandını başka bir yerlerde saklayabilirsiniz ve bu sayede yangında bütün yedek bantlarınız yansa bile geriye, en azından bir şeyler kalmış olur.

Şayet altı banttan fazlasına sahipseniz, diğerlerinitam yedekler için kullanabilirsiniz. Her yeni tam yedek yaptığınızda, daha eski olan bir bandı kullanırsınız. Bu yolla, çeşitli geçmiş haftalara ait kayıtlarınız olur ve şayet şu an silinmiş eski bir bilgiye ihtiyacınız olursa bunları kullanabilirsiniz.

14.4.1. **tar** ile yedekleme

Bir tam yedekleme **tar** ile kolayca yapılabilir:

```
# tar --create --file /dev/ftape /usr/src
tar: Üye isimlerinin sonundaki / kaldırılıyor
#
```

Yukarıda ki örnek **tar**'ın GNU sürümünü ve onun uzun seçenek isimlerini kullanmaktadır. **tar**'ın geleneksel sürümü sadece tek harflerden oluşan seçenekleri kullanır. GNU sürümü bir teypten daha fazlasını isteyen yedeklemeleri ve çok uzun dosya yolu tanımlamalarını da kontrol edebilir, fakat bütün geleneksel sürümler bu tür şeyleri yapamaz. Linux sadece GNU **tar** sürümünü kullanır.

Şayet yedekleriniz bir teybe sığmadıysa `--multi-volume` (`-M`) seçeneğini kullanmak zorundasınız:

```
# tar -cMf /dev/fd0u1440 /usr/src
tar: Üye isimlerinin sonundaki / kaldırılıyor
#2. bölümü /dev/fd0u1440 için hazırdıktan sonra return tuşuna basınız:
#
```

Disketleri, yedekleme işleminden önce biçemlemeyi unutmayın ya da başka bir pencere veya sanal uçbirimi tar ikinci disketi istediği zaman bunu yapmak için kullanın.

Bir yedekleme yaptıktan sonra düzgün çalışıp çalışmadığını anlamak için `--compare` (`-d`) seçeneği ile kontrol etmelisiniz:

```
# tar --compare --verbose -f /dev/ftape
usr/src/
usr/src/linux
usr/src/linux-1.2.10-includes/
....
#
```

Bir yedekleme sınamasını hata vermesi, orijinal bilgileri kaybedip yedeklerinizin çalışmadığı uyarısını almayacağınız anlamına gelir.

Bir fark yedeklemesi **tar**'ın `--newer` (`-N`) seçeneği ile yapılır:

```
# tar --create --newer '8 Sep 1995' --file /dev/ftape /usr/src --verbose
tar: Üye isimlerinin sonundaki / kaldırılıyor
usr/src/
usr/src/linux-1.2.10-includes/
usr/src/linux-1.2.10-includes/include/
usr/src/linux-1.2.10-includes/include/linux/
usr/src/linux-1.2.10-includes/include/linux/modules/
usr/src/linux-1.2.10-includes/include/asm-generic/
usr/src/linux-1.2.10-includes/include/asm-i386/
usr/src/linux-1.2.10-includes/include/asm-mips/
usr/src/linux-1.2.10-includes/include/asm-alpha/
usr/src/linux-1.2.10-includes/include/asm-m68k/
usr/src/linux-1.2.10-includes/include/asm-sparc/
usr/src/patch-1.2.11.gz
#
```

Şansızlık eseri, **tar** bir dosyanın düğüm bilgisinin değiştiğini algılayamaz, örneğin: değişmiş izin bitlerini veya isminin ne zaman değiştiğini. Bu **find** kullanılarak ve mevcut dosya sistemindeki dosyaların durumu ile daha önceki yedeklemenin karşılaştırılması şeklinde yapılabilir. Bunu yapacak betikler ve programlar Linux FTP sitelerinde bulunabilir.

14.4.2. **tar** ile dosyaların geri yüklenmesi

`--extract` (`-x`) seçeneği ile **tar** dosyaları geri alır:

```
# tar --extract --same-permissions --verbose --file /dev/fd0u1440
usr/src/
usr/src/linux
usr/src/linux-1.2.10-includes/
usr/src/linux-1.2.10-includes/include/
usr/src/linux-1.2.10-includes/include/linux/
usr/src/linux-1.2.10-includes/include/linux/hdreg.h
usr/src/linux-1.2.10-includes/include/linux/kernel.h
...
#
```

Komut satırında isimleri belirterek belirli dosya veya klasörleri ve onların alt klasörleri ve dosyalarını geri yükleyebilirsiniz:

```
# tar xpvf /dev/fd0u1440 usr/src/linux-1.2.10-includes/include/linux/hdreg.h
usr/src/linux-1.2.10-includes/include/linux/hdreg.h
#
```

Yedeklenmiş bilgiler içinde hangi dosyaların olduğunu görmek için `--list` (`-t`) seçeneğini kullanın:

```
# tar --list --file /dev/fd0u1440
usr/src/
usr/src/linux
usr/src/linux-1.2.10-includes/
usr/src/linux-1.2.10-includes/include/
usr/src/linux-1.2.10-includes/include/linux/
usr/src/linux-1.2.10-includes/include/linux/hdreg.h
usr/src/linux-1.2.10-includes/include/linux/kernel.h
...
#
```

Unutmayın ki **tar** yedek isimlerini ardışık olarak okur ve bu yüzden biraz yavaştır. Bununla birlikte, teyp sürücü veya sıralı erişim sağlayan aygıtlar ile rasgele erişimli veritabanı tekniklerini kullanmak mümkün değildir.

tar silinmiş dosyaları kontrol etmez. Bir tam ve fark yedeğinden bir dosya sistemini geri yüklemeniz gerekiyorsa, iki yedekleme arasında bir dosyayı sildiyseniz, yükleme sonrası bu dosya tekrar ortaya çıkacaktır. Şayet bu dosya sistemde bulunmaması gereken hassas bilgiler içeriyorsa, bu büyük bir problem yaratabilir.

14.5. Çok seviyeli yedekleme

Bir önceki bölümde anlatılan basit yedekleme yöntemi kişisel kullanımlar ve küçük siteler için yeterlidir. Daha ağır görevler için kullanılan sistemlerde, çok seviyeli yedekleme daha uygun olacaktır.

Basit yöntemin iki seviyesi vardır: tam ve fark. Bunlar her hangi bir seviye numarası kullanılarak genelleştirilebilir. Bir tam yedek 0 ve diğerleri 1, 2, 3 gibi numaralandırılabilir. Her bir fark yedeği seviyesinde, daha önceki yedeklemeden beri değişen her şeyi yeniden yedeklemeniz gerekir.

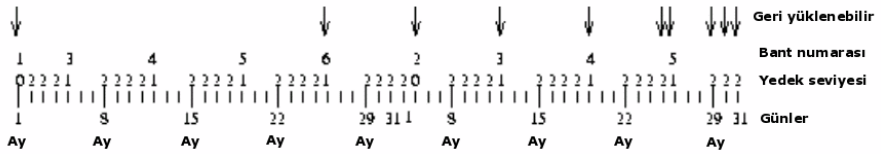
Bunu yapmamızın sebebi, daha ucuz bir şekilde bir yedekleme geçmişi oluşturmaktır. Bir önceki bölümde bulunan örnekte, yedekleme geçmişi birinci tam yedeğe kadar gitmektedir. Daha fazla banda sahip olarak bunu uzatabilirsiniz. Fakat her hafta için bir bant biraz pahalı olabilir. Uzun bir yedekleme geçmişi faydalı olabilir, çünkü bozulmuş veya silinmiş dosyalar uzun bir süre fark edilmeyebilirler. Dosyanın sürümü yeni olmasa bile, hiç olmamasından daha iyidir.

Çok seviyeli yedekleme sayesinde, yedekleme geçmişi daha ucuz bir şekilde uzatılabilir. Örneğin: şayet 10 adet bant alırsak; birinci ve ikinci bantları aylık olarak (her ayın birinci cuması) yedekleriz, 3–6 arası bantları haftalık (bir ayın diğer cumaları için, unutmadan bazı aylarda 5 Cuma olabilir, bunun için 4 adet fazladan banda

ihtiyacımız var) yedeklemeler için, 7–10 arası bantları günlük (pazartesi, Salı, Çarşamba, Perşembe) yedekleme için kullanılabilir. Sadece 4 adet yeni bant ile yedekleme tarihini 2 haftadan (bütün günlük teypler kullanıldıktan sonra) iki aya kadar uzatabiliriz. Bu iki ay esnasındaki dosyaların tüm sürümlerini saklayamayacağımız doğrudur ama yeterince sık aralıklar ile saklama yapabiliriz.

Çok seviyeli yedekleme tablosu örneği (sayfa: 72) her gün için hangi yedekleme seviyelerinin kullanıldığını ve ay sonunda hangi yedeklemelerin geri yüklenebilir olduğunu göstermektedir.

Şekil 8. Çok seviyeli yedekleme tablosu örneği



Yedekleme seviyeleri, dosya sistemlerinin geri yükleme süresini minimumda tutmak için de kullanılabilir. Şayet sıkıcı bir şekilde artan seviye numaralarına sahip pek çok fark yedeğiniz var ise bütün sistemi yeniden kurmak için, onların hepsini yeniden yüklemek zorundasınız.

Geri yüklenmesi gereken bant sayısını minimize etmek için, her bir fark yedeği bandı için daha küçük seviyeler kullanabilirsiniz. Bunun yanında, yedekleme için gereken süre artacaktır (her yedek bir önceki tam yedeklemeden itibaren her şeyi kopyalar). En iyi tablo **dump** kılavuz sayfasında önerilmektedir. Bu sıralı yedekleme seviyelerini kullanınız: 3, 2, 5, 4, 7, 6, 9, 8, 9... Bu hem yedekleme hem de geri yükleme süresini kısa tutacaktır. En fazla iki günlük işe eşit bilgileri yedeklemeniz gerekmektedir. Geri yüklemeniz gereken bant sayısı iki tam yedekleme arasını ne kadar uzun tuttuğunuza dayanır.

Çok sayıda yedekleme seviyesi ile verimli bir yedekleme örneği

Bant	Seviye	Yedekleme (gün)	Geri Yükleme (bant nr)
1	0	yok	1
2	3	1	1, 2
3	2	2	1, 3
4	5	1	1, 2, 4
5	4	2	1, 2, 5
6	7	1	1, 2, 5, 6
7	6	2	1, 2, 5, 7
8	9	1	1, 2, 5, 7, 8
9	8	2	1, 2, 5, 7, 9
10	9	1	1, 2, 5, 7, 9, 10
11	9	1	1, 2, 5, 7, 9, 10, 11
...	9	1	1, 2, 5, 7, 9, 10, 11, ...

Bu hayali tablo işçilerin sayısını azaltabilir, ama dikkatle izlemenizi gerektiren pek çok şey olduğu anlamına da gelmektedir. Değip değmeyeceğine siz karar vermelisiniz.

dump komutu yedekleme seviyelerini desteklemek için yaratılmıştır. **tar** ve **cpio** komutları için ek araçlar kullanmanız gerekir.

14.6. Neler yedeklenmeli?

Mümkün oldukça çok yedekleme yapmak isteyebilirsiniz. Temel istisna; kurulması daha kolay olan programlardır, yedeklenmesi gereken bazı ayar dosyaları olsa bile ve siz bunları tekrar tekrar düzenlemek istemeseniz bile hangi

yöntemin daha kolay olacağına siz karar vereceksiniz). Bir diğer temel istisna `/proc` dosya sistemidir. Çünkü burası çekirdek tarafından otomatik olarak yenilenen bilgiler ihtiva eder. Burayı yedeklemek hiç de akıllıca bir şey değildir. Özelliklede `/proc/kcore` dosyasının yedeklenmesi tamamen gereksizdir, çünkü burası fiziksel belleğinizin bir yansımasıdır ve genellikle çok büyüktür.

Yarı önemli bölümler: `/var` dizini içindeki haber havuzları, günlük kayıt dosyaları ve diğer şeylerdir. Neyin önemli olduğuna siz karar vermek zorundasınız.

Açık olan; kullanıcı dizinlerinin (`/home`) ve sistem ayar dosyalarının (`/etc` dizini ve mümkünse sistemde yayılmış olan diğerleri) yedeklenmesi gerektiğidir.

14.7. Sıkıştırılmış yedekler

Yedeklemek çok büyük alanlara ve çok paraya mal olabilir. Yer ihtiyacını azaltmak için yedekler sıkıştırılabilir. Bunu yapmanın çeşitli yolları vardır. Bazı programlar sıkıştırmayı desteklemek için yapılmışlardır. Örneğin; `--gzip` (`-z`) seçeneği, GNU **tar**'ı yedekleme ortamına yazmadan önce sıkıştırılmak üzere **gzip** sıkıştırma programına yönlendirir.

Maalesef sıkıştırma işlemleri problem yaratabilir. Sıkıştırmanın nasıl çalıştığına bağlı olarak, şayet bir bitlik alanda sorun varsa bütün sıkıştırılmış bilgi heba olabilir. Bazı yedekleme programları içlerinde hata düzeltmesi yerleşik olarak gelirler, fakat hiçbir sistem bu kadar çok sayıdaki hatayı kontrol edemez. Bunun anlamı bir yedekleme GNU **tar**'ın yaptığı gibi bütün çıktı tek bir birim olacak şekilde sıkıştırılırsa ve tek bir hata bile geri kalan bütün bilgiyi çöpe gönderebilir. Yedekler güvenilir olmalıdır ve bu sıkıştırma metotları hiç de iyi bir fikir değildir.

Alternatif bir yol ise, her bir dosyayı ayrı ayrı sıkıştırmaktadır. Bu yolla bir dosya zarar gördüğü halde, diğerleri herhangi bir zarara uğramamış durumda bulunurlar. Kayıp dosya her halükarda bozulmuş olacaktır, ama bu durum sıkıştırmayı hiç kullanmamaktan daha kötü değildir. **afio** programı (**cpio**'nun bir çeşidi) bunu yapabilir.

Sıkıştırma biraz zaman alan bir iştir. Bu da yedekleme programının banda yeterince hızlı bir şekilde yazamaması sonucunu ortaya çıkarır ve bu durumda teyp sürücüsü durur. Bu hem bant hem de sürücüsü için hiç iyi değildir. Bundan kaçmak için çıktının tamponlanması sistemi kullanılabilir, fakat bu da yeterince iyi çalışmayabilir. Bu sadece yavaş bilgisayarlar için bir problem teşkil edebilir.

15. Zaman Ayarları

> Zaman bir gözboyamadır. Öğle yemeği zamanı iki kere gözboyamadır." (Douglas Adams.)

Bu bölümde Linux sisteminin zamanı nasıl koruduğu ve sorunlardan kaçmak için ne yapmanız gerektiği anlatılmaktadır. Genellikle zaman hakkında endişe etmeniz gerekmez, ama nasıl çalıştığını bilmek iyidir.

15.1. Zaman dilimleri

Zamanın ölçümü en doğal ve düzenli olay üzerine kurulmuştur: dünyanın dönmesinden kaynaklanan gündüz ve gece değişim periyotları üzerine. Bu iki ardışık periyodun oluşturduğu toplam zaman sabittir. Sadece gece ve gündüzün uzunluğu değişir. Diğer bir sabit ise öğle vaktidir.

Öğle, Güneşin en yüksek noktada olduğu andır. Dünya yuvarlak olduğu için değişik bölgelerde değişik zamanlarda öğle olmaktadır. Bu yerel zaman kavramını ortaya çıkarmıştır. İnsanlar zamanı pek çok birim kullanarak ölçerler, pek çoğu doğal bir olgu olan "öğle" ye bağlıdır. Aynı yerde bulunduğunuz sürece yerel zamanın ne olduğu önemli değildir.

Uzak yerlerle haberleşme ihtiyacı ortaya çıkar çıkmaz, genel zaman kavramının zaruretini anlayacaksınız. Modern zamanlarda, dünyanın pek çok yeri diğer yerleri ile iletişim halindedir ve bu nedenle zamanı ölçmenin küresel

bir standardı tanımlanmıştır. Bu zamana evrensel zaman denir (UT ya da UTC – evrensel zaman, resmi olarak Greenwich Mean Time, GMT olarak geçer, çünkü İngiltere'de ki Greenwich bölgesinin yerel zamanına göre ayarlanmıştır). Değişik yerel zamanlara sahip insanlar birbirleri ile haberleşme ihtiyacı duydukları zaman, zamanı evrensel saate çevirir ve böylece neyin ne zaman yapılması gerektiği ortaya çıkar.

Her bir yerel zaman, zaman dilimi diye adlandırılır. Coğrafi şartlar, aynı anda öğle olan yerlerde aynı zaman dilimini kullanmaya imkan verirken, politikalar bunu zorlaştırmaktadır. Çok çeşitli sebeplerden dolayı, pek çok ülke yaz saati kullanmaktadır. Bu çalışırken daha fazla doğal ışık alabilmek için kendi saatlerini ileri almaları ve kış gelince geri çekmeleri anlamına gelir. Diğer ülkeler bunu yapmazlar. Bunu yapanlar, saatlerin ne zaman değiştirilmesi gerektiği konusunda anlaşamazlar ve kuralları her yıl değiştirirler. Bu da zaman bölgeleri değişimini çok önemsiz bir hale getirir.

Zaman bölgeleri, en iyi, yerleşim bölgesine göre ya da evrensel zamandan olan farkı söylenerek adlandırılır. ABD ve diğer bazı ülkelerde yerel zaman isimlere ve üç harften oluşan kısaltmalara sahiptir. Kısaltmalar benzersiz değildir ve ülkenin adı belirtilmeden kullanılmamalıdır. En iyisi konuşurken Helsinki, Doğu Avrupa diye söylemektir. Çünkü bütün Doğu Avrupa ülkeleri aynı kuralları kullanmazlar.

Linux bir zaman dilimleri paketine sahiptir ve bunun sayesinde bütün varolan zaman bölgelerini bilir ve gerekirse kolayca güncellenebilir. Bütün sistem yöneticileri kendilerine uygun zaman bölgesini seçmek zorundadırlar. Ayrıca, kullanıcılar kendi zaman ayarlarını yapabilirler, bu özellikle Internet üzerinden tek bir bilgisayarda çalışan ama farklı ülkelerde yaşayan kişiler için uygun olacaktır. İçinde bulunduğunuz zaman bölgesinde kullanılan yaz saati kuralları değiştiğinde, Linux sisteminin en azından bu bölümünü güncellediğinizden emin olun. Sistem zaman dilimini ayarlamak ve güncellemekten başka, zaman için endişelenmenizi gerektirecek çok az durum meydana gelir.

15.2. Yazılım ve donanım saatleri

Bir kişisel bilgisayar donanım saatini çalıştırmak için bir pil kullanır. Bu pil sayesinde, bilgisayarda elektrik olmasa bile saatin çalışması sağlanır. Donanım saati BIOS ayar ekranından veya işletim sisteminin her neresinde bulunuyorsa oradan ayarlanabilir. Linux çekirdeği zamanı, donanım saatinden bağımsız olarak saklar. Açılış esnasında Linux kendi saatini, donanım saatine göre ayarlar. Bundan sonrasında her iki saatte birbirinden bağımsız olarak çalışır. Linux kendi saatini korur, çünkü her seferinde donanım saatine bakmak yavaş ve karışık bir iştir.

Çekirdek saati daima evrensel zamanı gösterir. Böylece çekirdek bütün zaman dilimleri hakkında bilgi sahibi olmak zorunda kalmaz. En basit sonuçları daha yüksek güvenilirlik ve güncelleme esnasında kolaylıktır. Her süreç, kendi zaman değişimini kendi kontrol eder (timezone paketinin parçası olan standart araçları kullanarak).

Donanım saati evrensel veya yerel saatte olabilir. Bunu evrensel zamanda tutmak genellikle daha iyidir, çünkü yaz saati uygulaması başladığı ya da bittiği zaman donanım saatini değiştirmek zorunda kalmazsınız. (UTC, yaz saati farkını içermez.) Şansızlık eseri, bazı PC işletim sistemleri; MS–DOS, Windows ve OS/2 dahil, donanımın yerel zamanı gösterdiğini varsayarlar. Linux birinden birini kullanabilir, fakat şayet donanım zamanı yerel saati gösteriyorsa yaz saatinin başlangıç ve bitimlerinde sistemi yeniden ayarlamanız gerekir (aksi taktirde saat yerel zamanı göstermez).

15.3. Zaman gösterimi ve ayarlanması

Linux Sistemlerinde, sistem zaman dilimi ayarı `/etc/localtime` sembolik bağı ile ayarlanır. Bu bağ yerel zaman dilimini gösteren bir zaman bölge bilgi dosyasını işaret eder. Zaman dilimi bilgi dosyaları, kullandığınız dağıtıma bağlı olarak ya `/usr/lib/zoneinfo` ya da `/usr/share/zoneinfo` içerisinde saklanır.

Örneğin New Jersey'de bulunan bir SUSE sisteminde `/etc/localtime` bağı `/usr/share/zoneinfo/US/Eastern` dosyasını işaret eder. Bir Debian sisteminde ise

`/etc/localtime` bağı `/usr/lib/zoneinfo/US/Eastern` dosyasını işaret eder.

Eğer `zoneinfo` dizinini ne `/usr/lib` nede `/usr/share` dizinleri altında bulamıyorsanız; ya **`find /usr -print | grep zoneinfo`** komutunu kullanın ya da dağıtımınızın belgelerine bakın.

Farklı bir zaman diliminde yerleşik bir kullanıcınız varsa ne olur? Bir kullanıcı kendi zaman dilimi ayarını TZ ortam değişkeninde belirterek değiştirebilir. TZ ortam değişkeni ayaranmamışsa, sistem zaman dilimi geçerli olur. TZ değişkeninin sözdizimi **`tzset`** kılavuz sayfasında açıklanmıştır.

`date` komutu o anki tarih ve saati gösterir. [15] Örneğin:

```
$ date +%c
Prş 16 Oca 2003 18:10:26 EET
$
```

Günlerden Perşembe, 2003 yılının 16 Ocağı, akşam 6'yı 10 geçiyor (Saat EET yani Doğu Avrupa zaman dilimine göre ayarlı, `+%c` seçeneği tarih ve saati yerel biçemde göstermek içindir). **`date`** aynı zamanda evrensel zamanı da gösterebilir:

```
$ date -u +%c
Prş 16 Oca 2003 16:10:26 UTC
$
```

Ayrıca **`date`** çekirdeğin yazılım saatini ayarlamak için de kullanılabilir:

```
# date 1042734297
Prş Oca 16 18:24:57 EET 2003
# date +%c
Prş 16 Oca 2003 18:25:10 EET
#
```

Ayrıntılı bilgi için **`date`** kılavuz sayfasına bakabilirsiniz. Sözdizimleri biraz esrarlı olabilir. Sadece root saat ayarı yapabilir. Her kullanıcı kendi için zaman dilimi ayarı yapabilirken, saat herkes için aynıdır.

`date` sadece yazılım saatini gösterir veya ayarlar. **`clock`** komutu yazılım ve donanım saatleri arasındaki uyumu sağlar. Sistem açılış esnasında donanım saatini okumak ve yazılım saatini ayarlamak için kullanılır. Şayet her iki saati de ayarlamanız gerekirse önce yazılım saatini **`date`** ile ayarlayın ve daha sonra **`clock -w`** ile bunu donanım saatine yazın.

`clock` ile kullanılan `-u` seçeneği, donanım saatine evrensel zaman içinde olmasını söyler. `-u` deçeneğini doğru bir şekilde kullanmalısınız. Aksi taktirde zaman konusunda, bilgisayarınızın kafası biraz karışabilir.

Saat dikkatli bir şekilde değiştirilmelidir. Unix sisteminin pek çok parçası, saatin düzgün bir şekilde çalışmasına bağlıdır. Örneğin: **`cron`** süreçleri komutları periyodik olarak çalıştırırlar. Şayet saati değiştirirseniz, komutları çalıştırıp çalıştırmama konusunda karışıklık ortaya çıkabilir. Daha eski bir Unix sisteminde, birileri saati yirmi yıl sonraki gelecek bir zamana ayarladı ve cron bu yirmi yıllık periyodik komutların hepsini birden yapmaya kalktı. **`cron`**'un şu anki sürümü bunu kontrol edebilir ama siz yine de çok dikkatli olmalısınız. İleri veya geriye doğru sıçrayışlar, küçüklerden ve ileri doğru sıçramalardan daha tehlikelidir.

15.4. Saat yanlışsa

Linux sistem saati daima doğru değildir. PC donanımı tarafından üretilen periyodik bir zaman kesmesinin çalıştırılması ile tutulur. Şayet sistemin yapması gereken pek çok iş var ise zaman kesmesinin kullanılması çok gecikebilir ve sistem saati geri kalabilir. Donanım saati bağımsız çalışır ve çok daha doğrudur. Şayet sisteminizi sık sık açarsanız (sunucu harici bilgisayarlarda olduğu gibi) saatiniz daha doğru olacaktır.

Şayet donanım saatini ayarlamak isterseniz, sistem enerjisini kapatıp BIOS ayarlarına girip orada gerekenleri yapın. Bu sistem saatini değişmesi sonucu ortaya çıkabilecek bütün sorunlardan sizi kurtarır. Şayet bu ayarları

BIOS üzerinde yapma şansınız yoksa, sırasıyla **date** ve **clock** komutlarını kullanın ve bazı süreçlerin komik hareketlerine karşı sistemi yeniden başlatmaya hazır olun.

Donanım saatinizin, sistem saatinize ayarlanmasını diğer bir yolu ise, **hwclock -w** veya **hwclock --systohc** komutlarından birisini kullanmaktır. Eğer sistem saatinizin donanım saatinize göre ayarlanmasını istiyorsanız; **hwclock -s** veya **hwclock --hctosys** komutlarından birisini kullanmalısınız.

15.5. Ağ Zaman Protokolü

Bir ağ bilgisayarı (bir modemin ucunda bile olsa) kendi saatini otomatik olarak ayarlayabilir. Bunu diğer bilgisayarların zamanı ile karşılaştırarak yapar. NTP (Ağ Zaman Protokolü)'nin yaptığı şey budur. Sisteminizin saatini, test eder ve uzak sunucudaki saat ile uyumlu hale getirir. NTP ile, sisteminiz milisaniyeler içinde, Evrensel Zamana Uyumlu (Coordinated Universal Time/ UTC) hale getirilir.[16]

Pek çok Linux kullanıcısı için bu oldukça hoş bir lükstür. Benim evimdeki bütün saatler, Linux sistemine göre ayarlıdır. Daha büyük organizasyonlar için bu lüks, kaçınılmaz bir gereklilik olabilir. Zamanlarına göre ayrılmış olaylar için kütük dosyalarında arama yapabilmek, işleri oldukça kolaylaştırır ve hata ayıklamada "tahmini işlerle" uğraşmak zorunda kalmanızı önler.

NTP'nin önemini gösteren diğer bir örnekte SAN'dır. Bazı SAN'lar zaman dilimi kontrolleri ve dosya sistemi uyumunun düzgün ayarlanması için NTP ayarına ihtiyaç duyarlar. Bazı SAN'lar ve bazı uygulamalar ileri tarihli bir zaman etiketine sahip dosyalar ile uğraşırken, kafaları karışabilir.

Pek çok Linux dağıtımı bir NTP paketi ile gelir. Bu ya `.deb` ya da `.rpm` paketi olabilir. Bu paketleri NTP kurmak için kullanabilirsiniz ve yahut kaynak kodlarını `http://www.ntp.org/downloads.html` adresinden alabilirsiniz ve kendiniz derleyebilirsiniz. Her halükarda temel ayarlar aynıdır.

15.6. Temel NTP Ayarları

Hangi dağıtımı kullandığınıza bağlı olarak NTP ayarları ya `/etc/ntp.conf` ya da `/etc/xntp.conf` dosyasından yapılır. NTP ayarlarının ayrıntılarına çok fazla girmeyeceğim. Burada sadece temel ve basit ayarları anlatacağım.

Temel bir `ntp.conf` dosyası şuna benzer:

```
# --- GENEL YAPILANDIRMA ---
server  aaa.bbb.ccc.ddd
server  127.127.1.0
fudge   127.127.1.0 stratum 10

# Hedef dosya.

driftfile /etc/ntp/drift
```

Pek çok temel yapılandırmalı `ntp.conf` dosyasında iki adet sunucu ismi mevcuttur. Birisi, saat ayarının yapılmasını istenen sunucunun adı ve sahte bir IP adresinden oluşur (bu örnekte 127.127.1.0). Sahte IP adresi ağ problemleri olması durumunda veya NTP sunucunun kapalı olması/çökmesi durumunda kullanılır. Sisteminizdeki NTP uygulaması, uzak NTP sunucusu ayağa kalkınca, sistem saatini tekrar ona göre ayarlayacaktır. Bir iki sunucudan birincisi asıl sunucu olarak işlem yapar, ikincisi ise yedek amaçlıdır: Kötü günlerinizde kullanın diye.

Ayrıca bu hedef dosyanın yerini de belirtmelisiniz. Zamanla NTP, sistem saatindeki hata oranını "öğrenecek" ve kendini buna göre ayarlayacaktır.

Daha iyi bir kontrol sağlamak ve güvenliği elden bırakmamak için NTP'de kısıtlama/restrict seçeneği kullanılabilir. Örneğin:

```
# Bu servise genel erişimi yasakla.
resrict default ignore

# Bu ağdaki sistemlere, şu zaman servisleri ile ayarlama
# yapmalarına izin ver. Fakat bizim zaman ayarımızı değiştirme.
restrict aaa.bbb.ccc.ddd nomodify

# ntpd'ye aşağıdaki şu kısıtlanmamış erişime izin ver.

restrict aaa.bbb.ccc.ddd
restrict 127.0.0.1
```

Kısıtlama seçeneğini kullanmadan önce, düzgün çalışan bir NTP (Ağ Zaman protokolü) servisine sahip olmanız şiddetle tavsiye edilir. Kazara, uyarlama yapılmasına kısıtlama getirebilir ve nerede hata olduğunu bulmak için saatlerinizi harcayabilirsiniz.

NTP servisi, sistem saatinizi yavaşça ayarlayacaktır. Sabırlı olun!!! Basit bir test: Yatmaya gitmeden 10 dakika önce sistem saatini değiştirin ve yatın. Sabah kalktığınızda sistem saatinizin doğru olduğunu göreceksiniz.

15.7. NTP Araçları

NTP servisinin işni düzgün yapıp yapmadığını kontrol etmek için birkaç araç vardır. **ntpq -p** komutu, sisteminizin o anki zaman durumunu gösterecektir.

```
# ntpq -p
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
*cudns.cit.corne ntp0.usno.navy.  2 u  832 1024  377   43.208    0.361   2.646
 LOCAL(0)        LOCAL(0)        10 l   13   64  377    0.000    0.000   0.008
```

ntpdc -c loopinfo komutu; uzak sunucuya yapılan en son bağlantıdan beri sistem saatinin ne kadarlık bir hata yaptığını saniye cinsinden gösterir.

```
# ntpdc -c loopinfo
offset:               -0.004479 s
frequency:            133.625 ppm
poll adjust:          30
watchdog timer:       404 s
```

ntpdc -c kerninfo komutu, o anki kalan doğrulamayı gösterir.

```
# ntpdc -c kerninfo
pll offset:           -0.003917 s
pll frequency:        133.625 ppm
maximum error:        0.391414 s
estimated error:      0.003676 s
status:               0001  pll
pll time constant:    6
precision:            1e-06 s
frequency tolerance:  512 ppm
pps frequency:        0.000 ppm
pps stability:        512.000 ppm
pps jitter:           0.0002 s
calibration interval: 4 s
calibration cycles:   0
jitter exceeded:      0
```

```
stability exceeded:    0
calibration errors:    0
```

ntpdc -c kerninfo komutunun çok az değişik bir sürümü de **ntptime** komutudur.

```
# ntptime
ntp_gettime() returns code 0 (OK)
  time c35e2cc7.879ba000  Thu, Nov 13 2003 11:16:07.529, (.529718),
  maximum error 425206 us, estimated error 3676 us
ntp_adjtime() returns code 0 (OK)
  modes 0x0 (),
  offset -3854.000 us, frequency 133.625 ppm, interval 4 s,
  maximum error 425206 us, estimated error 3676 us,
  status 0x1 (PLL),
  time constant 6, precision 1.000 us, tolerance 512 ppm,
  pps frequency 0.000 ppm, stability 512.000 ppm, jitter 200.000 us,
  intervals 0, jitter exceeded 0, stability exceeded 0, errors 0.
```

NTP'nin nasıl çalıştığını görmenin bir diğer yöntemide **ntpdate -d** komutudur. Bu komutla sisteminiz bir NTP sunucuya bağlanır ve zaman sapması var ise bunu size gösterir ama sistem saatini ayarlamaz.

```
# ntpdate -d 132.236.56.250
13 Nov 14:43:17 ntpdate[29631]: ntpdate 4.1.1c-rc1@1.836 Thu Feb 13 12:17:20
transmit(132.236.56.250)
receive(132.236.56.250)
transmit(132.236.56.250)
receive(132.236.56.250)
transmit(132.236.56.250)
receive(132.236.56.250)
transmit(132.236.56.250)
receive(132.236.56.250)
transmit(132.236.56.250)
sunucu 132.236.56.250, port 123
stratum 2, precision -17, leap 00, trust 000
refid [192.5.41.209], delay 0.06372, dispersion 0.00044
transmitted 4, in filter 4
reference time:    c35e5998.4a46cfc8  Thu, Nov 13 2003 14:27:20.290
originate timestamp: c35e5d55.d69a6f82  Thu, Nov 13 2003 14:43:17.838
transmit timestamp:  c35e5d55.d16fc9bc  Thu, Nov 13 2003 14:43:17.818
filter delay:  0.06522  0.06372  0.06442  0.06442
         0.00000  0.00000  0.00000  0.00000
filter offset: 0.000036 0.001020 0.000527 0.000684
         0.000000 0.000000 0.000000 0.000000
delay 0.06372, dispersion 0.00044
ofset 0.001020

13 Nov 14:43:17 ntpdate[29631]: adjust time server 132.236.56.250 ¬
offset 0.001020 sec
```

Sistem saatinizin ayarlanma sürecini izlemek istiyorsanız **ntptrace** komutunu kullanın.

```
# ntptrace 132.236.56.250
cudns.cit.cornell.edu: stratum 2, offset -0.003278, synch distance 0.02779
dtc-truetime.ntp.aol.com: stratum 1, offset -0.014363, synch distance 0.00000, ¬
refid 'ACTS'
```

Şayet sistem saatinizin acilen ayarlanması gerekiyorsa, **ntpdate uzak-sunucu-ismi** komutunu kullanabilirsiniz. Ayar için beklemeniz gerekmez!

```
# ntpdate 132.236.56.250
13 Nov 14:56:28 ntpdate[29676]: adjust time server 132.236.56.250 ¬
offset -0.003151 sec
```

15.8. Bazı NTP sunucuları

Halka açık NTP sunucularının bir listesini http://www.eecis.udel.edu/~mills/ntp/servers.html/ adresinden alabilirsiniz. Bir sunucuyu kullanmadan önce sayfadaki açıklamaları dikkatlice okuyunuz. Bütün sunucular, kendisine bağlanan herkesin zaman ayarlarını yapacak kadar bant genişliğine sahip değildir. Bu nedenle, bir sistem yöneticisine başvurup, onun bilgisayarını bir NTP sunucu olarak kullanmak istediğinizi bildirmek daha uygun olabilir.

15.9. NTP Bağları

NTP'ler hakkında daha ayrıntılı bilgi NTP ana sayfasından elde edilebilir: http://www.ntp.org/

Veya http://www.ntp.org/ntpfaq/NTP-«a-«faq.htm sayfasını kullanabilirsiniz.

16. Yardım Bulmak

"Help me if you can I'm feeling down. And I do appreciate you being 'round." – The Beatles

16.1. Haber grupları ve eposta listeleri

Bu kılavuz size Linux hakkında her şeyi öğretemez. Bunun için yeterli yerimiz yok. Aynı konular üzerinde başka bilgilere ihtiyaç duymanız kaçınılmazdır. Ve bu LDP'deki herhangi bir kılavuzda açıklanmamış olabilir.

Linux'un hoş taraflarından biriside, kendini bu işe adamış pek çok forumun bulunmasıdır. Yeni başlayanlar için SSS'lardan çekirdek gelişimine kadar Linux'un her yönünü kapsayan forumlar mevcuttur. Bunlardan faydalanmak için yapmanız gereken birkaç şey vardır.

16.1.1. Doğru forumun bulunması

İlk önce uygun bir forum bulmalısınız. Pek çok haber grubu ve posta listeleri vardır, sizin sorularınıza en uygun olan birini bulmayı ve kullanmayı deneyin. Örneğin: Linux çekirdek gelişimine adanmış bir forumda sendmail hakkında soru sormak pek mantıklı değildir. En iyi ihtimalle birkaç cevap alırsınız ve insanlar sizin aptal olduğunuzu düşünür veya daha kötüsü aşağılayıcı pek çok cevap alabilirsiniz. Çabuk bir araştırma sonucu comp.mail.sendmail adında, sendmail ile ilgili soru sormaya uygun bir yer bulabilirsiniz. Haber istemcinizin size uygun haber gruplarını içeren bir listesi olabilir veya tüm haber gruplarının tam bir listesini şu adresten bulabilirsiniz[17]: http://groups.google.com/groups?group=*

16.1.2. Postalamadan önce

Şimdi uygun bir forum buldunuz ve sorunuzu yollamaya hazır olduğunuzu düşünüyorsunuz. Durun. Henüz hazır değilsiniz. Kendi kendinize sorunun cevabını bulmaya çalıştınız mı? Pek çok NASIL ve SSS belgesi mevcuttur; bunlardan birisi sizin sorununuzu içeriyorsa önce bu belgeyi okuyun. Sorunuzun cevabını içermese bile, sorununuzu daha doğru ve etkin bir şekilde aktarmak için gereken bilgiyi size kazandırabilirler. Ayrıca haber gruplarının ve posta listelerinin arşivleri vardır ve muhtemelen sizin sorunuz daha önce sorulmuş ve cevaplanmış olabilir. http://www.google.com ve benzeri arama motorları sorunuzu göndermeden önce bakabileceğiniz diğer yerlerdir.

16.1.3. Posta iletisinin yazılması

Tamam, doğru forumu buldunuz, ilgili SSS ve NASIL belgelerini okudunuz, web'i taradınız ama sorunuzun cevabını halen bulamadınız. Şimdi postanızı yazmaya başlayabilirsiniz. Aşağıdakine benzer bir şeyler söyleyerek, konu hakkında daha önceden bazı belgeleri okuduğunuzu açıklığa kavuşturmak iyi bir şey olacaktır: "WinModem HOWTO ve PPP FQA belgelerini okudum, ama hiç biri benim aradığım şeyi ihtiva etmiyor. 'WinModem Linux PPP ayarı' ile ilgili Google'da hiçbir şey bulamadım." Bu cümle, kaşıkla beslenmeye ihtiyaç duyan tembel bir gerzek yerine sizi biraz çaba sarf etmiş biri olarak gösterecektir. İlk şey, şayet cevabı bilen birisi varsa size bir cevap geleceğidir, sonraki ise; derin bir sessizlik ile karşılaşmak gibidir.

Açık seçik, imlâ ve yazım kurallarına dikkat ederek yazın. Bu çok önemlidir. Bu sizin düşünceli ve titiz birisi olduğunuz izlenimini verir. Aptal gibi görünmektense eğitimli ve akıllı birisi gibi görünmeyi deneyin. Söz veriyorum işinize yarayacaktır.

BENİM YAPTIĞIM GİBİ büyük harfler kullanmayın. Bu bağırdığınız ve kaba biri olduğunuz anlamına gelecektir.

Problemin ayrıntılarını ve bu problemi gidermek için neler yaptığınızı açıkça gösterin. "Linux çalışmayı durdurdu, ne yapmalıyım?" gibi bir soru hiçbir işe yaramaz. Nerede çalışma durdu? Ne şekilde çalışmayı durdurdu? Mümkün olduğunca kusursuz olmalısınız. Bunun yanında sınırlar da vardır. Konuyla alakasız bilgiler göndermeyin. Şayet posta alıcınız ile ilgili bir probleminiz varsa, bunun çekirdeğinizin açılış kayıtlarının (`dmesg`) aptallığı ile bir ilgisi olma ihtimali yoktur.

Asla özel eposta göndererek soru sormayın. Bu listelerdeki ana nokta, herkesin birbirinden bir şeyler öğrenmesidir. Özel soru iletileri, haber grupları veya posta listelerinin değerini ortadan kaldırır.

16.1.4. Epostanın biçimi

HTML biçeminde postalamayın. Pek çok Linux kullanıcısı, HTML iletileri rahatça okuyamayacakları posta alıcılarına sahiptirler Biraz çaba ile okuyabileceklerken, onlar bunu yapmazlar. Şayet HTML ileti yollarsanız genellikle okunmadan silinir. Düz yazı iletiler gönderin, bu yolla daha geniş bir kitleye ulaşabilirsiniz.[18]

16.1.5. Takip edin

Probleminiz çözüldükten sonra, problemin ne olduğunu ve nasıl çözdüğünüzü içeren kısa bir mesaj yollayın. Bu hem, gelecekte aynı problemle karşılaşacak birisine yardımcı olacaktır, hem de insanlar bu sorunla ilgili tartışmanın bittiğini anlayacaklardır. Haber grubu veya posta listesi arşivine baktıkları zaman sizin de bu problemle karşılaştığınızı, sorunuzu ve sorunun çözümünü göreceklerdir.

16.1.6. Daha fazla bilgi

Bu bölümdeki kısa kılavuz, Eric S. Raymond tarafından yazılmış daha ayrıntılı ve mükemmel bir kılavuz olan How To Ask Questions The Smart Way[B49]'in kısa bir özetidir. Herhangi bir şey postalamadan önce bu kılavuzu okumanız tavsiye edilir. Cevabını aradığınız sorunuz için cevap bulma olasılığınızı yükseltecek formüller içermektedir.

16.2. IRC

IRC (Internet Relay Chat), Eric Raymond'ın kılavuzunda anlatılmamaktadır ama IRC ihtiyaçlarınıza cevap verebilecek mükemmel bir seçenek olabilir. Bununla birlikte, bu doğru bir şekilde soru sorma hakkında biraz pratik gerektirir. Pek çok IRC #Linux kanalları oldukça meşguldür ve şayet sorununuzun cevabı kılavuz sayfalarında ve NASIL belgelerinde varsa, bunları okumanızın söyleneceğini umabilirsiniz. İmla ve yazım kuralları burada da geçerlidir.

Posta listeleri ve haber grupları ile ilgili söylenenlerin pek çoğu IRC içinde geçerlidir. Ek olarak aşağıdakiler vardır.

16.2.1. Renkler

Renkli, kalın, altı çizgili veya garip karakterler kullanmayın. Bunlar bazı eski istemcilerle uyumlu değildir ve okunması çok zor olan metinlerdir. Bir kanala girer ve hemen renkler veya kalınlık kusmaya başlarsanız dışarı atılmanız an meselesidir.

16.2.2. Kibar olun

Cevap almak size verilmiş bir hak değildir. Şayet düzgün bir şekilde sorarsanız, büyük ihtimalle bir cevap alırsınız ama bir cevap almak "hakkınız" yoktur. Linux IRC kanallarındaki insanlar kendi zamanlarını harcıyorlar ve hiç kimse onlara bunun için bir ödeme yapmıyor, özellikle de siz.

Kibar olun. Size nasıl davranılmasını istiyorsanız, diğerlerine de öyle davranın. Şayet insanların size karşı nazik olmadığını düşünüyorsanız bile, kibar olun, sinirlenmeyin ve insanları isimleri ile çağırmayın. Bu sizin onların seviyesine inmenizdense, onların aptal görünmesini sağlayacaktır.

Büyük bir alabalık gibi, gidip kimseye vurmayın. Bunun daha önceden bir veya iki kere yapıldığını ve komik bulunduğunu mu sanıyorsun?

16.2.3. Bulunduğunuz kanalda kullanılan dilde ve düzgün yazın.

Pek çok #Linux kanalı İngilizce'dir. Onlarla İngilizce konuşun. Büyük IRC ağlarının pek çoğu diğer dillerde de kanallar açarlar. Örneğin; Fransızca kanalının adı #Linuzfr, İspanyolca'sının adı #linuxes veya #linuxlatino olabilir. Şayet size uygun bir kanal bulamazsanız, ana #linux kanalında İngilizce olarak bunu sorun, büyük ihtimalle yardım bulabilirsiniz.[19]

"1337 Hhghgdka456745 kjfksj +%%&57677" şekline mesaj yazmayın. Diğer insanlar yapsa bile. Bu aptal görünür ve sizin de bir aptal gibi görünmenize sebep olur. En iyisi, sadece salak gibi görünmektir, daha da kötüsü kıçınıza tekmeyi yiyebilirsiniz.

16.2.4. Port taraması

Asla herhangi bir kişiden port taraması yapmasını veya sizi hacklemesini istemeyin. Bu çok ayıptır. Bağlantı yaptığınız IP'nin sizin olduğunun bilinmesine veya olduğunu söylediğiniz kişi olup olmadığınızın tespit edilmesine imkan yoktur. İnsanları, böyle bir isteğe hayır deme zahmetine sokmayın.

İstemiş olsalar bile, hiç kimsenin portlarını taramayın. Kim olduklarını veya kullandıkları IP'nin kendilerine ait olup olmadığını bilmenize imkan yoktur. Bazı yargılamalarda port taraması suç olarak kabul edilir ve pek çok ISS'nin anlaşmalarına aykırıdır. Pek çok kişi TCP bağlantılarının kayıtlarını tutar ve port taramasından geçirildiklerini anlayabilirler. Ve pek çok kişi sizi ISS'ye şikayet edecektir. Kim olduğunuzu bulmak çok kolay bir iştir.

16.2.5. Kanalda kalın

Sizden istenmedikçe kimseye /msg göndermeyin. Bu kanalın yararlılığını azaltmakta ve bazıları ise sadece, bunu tercih etmemektedir.

16.2.6. Konu içinde kalın

Konuda kalın. Bu kanal bir Linux kanalıdır. "İbo geçen hafta kimi dövdü" kanalı değildir. Başkalarının konu dışına çıkmış olması, sizin de çıkabileceğiniz anlamına gelmez. Onlar büyük ihtimal ile kanalın düzenli takipçileridir ve onlara farklı kurallar uygulanmaktadır.

16.2.7. CTCP'ler

Kanalı CTCP kalabalığı, sürümleri veya buna benzer bir şeyle doldurmak istiyorsan tekrar düşün. Çok çabuk atılmana sebep olabilir.

16.2.8. Hacking, Cracking, Phreaking, Warezing

Selamsız bir şekilde dışarı atılmak istemiyorsanız, istismar edici davranışlar hakkında soru sormayın.

Bir #Linux kanalında iken her hangi bir hacker/cracker/phreaker/warezer kanallarında bulunmayın. Bazı sebeplerden dolayı, #Linux kanallarındaki görevliler, başkalarının yazılımlarını çalmayı veya donanımlarına zarar vermeyi düşünen insanlardan nefret ederler[20] nedenini hayal edebilir misiniz?

16.2.9. Toparlayalım

Çok fazla YAPMA ve az sayıda YAP için özür dilerim. YAPlar, posta listeleri ve haber grupları bölümünde çokça kullanıldı.

Muhtemelen yapabileceğiniz en iyi şey; bir #Linux kanalına girmek ve doğru tonlamayı bulmak için hiçbir şey söylemeden bir yarım saat beklemek olacaktır.

16.2.10. Daha fazla okunacak kaynak

IRC #Linux kanallarından nasıl daha fazla yararlanılacağına dair mükemmel FAQ'lar vardır. Pek çok #Linux kanalı bir FAQ veya kurallar sayfasına sahiptir. Bunu nasıl bulacağınız genellikle kanal konusu içinde açıklanır veya /topic komutunu kullanabilirsiniz. Şayet kurallar var ise bunları okuduğunuzdan emin olun. http://linuxfaq.quartz.net.nz adresinde bulabileceğiniz "Undernet #Linux FAQ" belgesini tavsiye ederim.

A. Terim Dağarcığı

Burada Linux ve sistem yönetimi ile ilgili bazı terimlerin kısa tanımları bulunmaktadır. Terimlerin İngilizce karşılıkları yanlarında parantez içinde verilmiştir.

açılış [İng.: booting]
Bilgisayarın enerji düğmesine basılması ile işletim sisteminin komut kabul edebilir bir duruma gelmesi için geçen süre içinde olan her şeye açılış denir.

açılış disketi [İng.: emergency boot floppy]
Sabit disk üzerindeki dosya sisteminin zarar gördüğü durumlarda veya önyükleyici kullanılamadığı durumlarda işletim sistemini açmak için kullanılan diskettir. Pek çok Linux dağıtımı, kurulum esnasında bu disketi yapmayı önerir. Bu disketi oluşturmanız şiddetle tavsiye edilir. Şayet elinizdeki Linux dağıtımında bu özellik yok ise Açılış Disketi NASIL (Boot Floppy HOWTO) belgesini okuyun.

açılışta kendi kendini sınama [İng.: Power on self test (POST)]
Bilgisayara enerji verildiği zaman yapılan bir dizi teşhis amaçlı sınamalardır. Bu sınama genellikle; belleğin denetlenmesisini, donanım yapılandırılmasının en son kayıtlı şekilde olup olmadığını, BIOS'a kayıtlı herhangi çalışır bir disket sürücü veya sabit disk sürücünün denetlenmesi şeklindedir.

ağ dosya sistemi [İng.: Network File System (NFS)]
Sun Microsystems firması tarafından geliştirilmiş ve RFC 1094 olarak tanımlanmış; bir bilgisayarın ağ üzerinden sanki kendi yerel diskine ulaşırmış gibi erişim yapmasını sağlayan protokoldür.

ana önyükleme kaydı [İng.: Master Boot Record (MBR)]
Bir disk yüzeyindeki ilk mantıksal sektördür. Bilgisayarın açılabilmesi için BIOS buradaki küçük bir programı kullanır.

artalan süreci [İng.: daemon]
Genellikle farkedilemeyen ve arka planda gizlenen, kendilerini tetiklemek için bir şeylerin olmasını bekleyen süreçlerdir. Örneğin; **`sendmail`** birisi eposta gönderince uyanır, **`update`** her 30 saniyede bir tampon belleğin içeriğini diskin üzerine yazmak üzere uyanır.

biçemleme [İng.: formatting]
Basitçe; biçemleme bir disk yüzeyinin izler, sektörler ve silindirlere ayrılması ve düzenlenmesi işlemidir. Bazen, yanlış bir biçimde, bir dosya sisteminin disk yüzeyine yazılması olarak da kullanılır (MS Windows/MS DOS çevrelerinde) Dosya sisteminin disk üzerinde yazılması işlemine *dosya sisteminin oluşturulması* denir.

bozuk blok [İng.: bad block]
Güvenilir bir şekilde bilgi depolayamayan sektörler. Genellikle bir disk üzerindeki manyetik özelliklerini kaybetmiş bir sektör grubudur.

bozuk sektör [İng.: bad sector]
Güvenilir bir şekilde bilgi depolayamayan bir sektör. Genellikle bir disk üzerindeki manyetik özelliklerini kaybetmiş bir sektördür.

CMOS RAM
CMOS (**C**omplenmentary **M**etal **O**xide **S**emiconductor) "Eşlenik Metal Oksit Yarıiletken" in kısaltmasıdır. Karışık bir teknolojidir ama basitçe; elektrik akımı olmadığı durumlarda durumlarını değiştirmeyen transistör çiftleridir denebilir. Böylece bu yarıiletken, enerjisiz durumda iken bilgi saklayabilen bir durağan (statik) RAM olarak kullanılabilir.

çalışma seviyesi [İng.: run level]

Linux genellikle 7 tanesi tanımlanmış olan 10 adet (0–9) çalışma seviyesine sahiptir. Her bir çalışma seviyesi değişik servisler ve sisteme verilmiş değişik yapılandırmalar ile başlayabilir. Çalışma seviyesi 0 "sistemin durdurulması", çalışma seviyesi 1 "tek kullanıcılı kip", çalışma seviyesi 6 "sistemin yeniden başlatılması" olarak tanımlanmıştır. Teorik olarak geri kalan seviyeler sistem yöneticisi tarafından tanımlanabilir. Bununla birlikte pek çok dağıtım bunların bazılarını öntanımlı olarak bulundururlar. Örneğin; çalışma seviyesi 2 "çok kullanıcılı konsol" ve çalışma seviyesi 5 "çok kullanıcılı X Pencere Sistemi" olarak tanımlamış olabilir. Bunlar dağıtımdan dağıtıma göre farklılıklar gösterebilir, bu nedenle kendi sisteminizi kontrol etmeniz gerekir.

çekirdek [İng.: kernel]
Kaynakların paylaşımı ve donanımlar arası etkileşimi sağlayan işletim sistemi parçasıdır. Ayrıca *sistem programları* (sayfa: 86)na bakınız.

disk bölümü [İng.: partition]
Diskin mantıksal bölümlerinden her biri (mantıksal bölüm ile karıştırmayın). Her bölüm normalde kendi dosya sistemine sahiptir. Unix, bölümlerin sanki bağımsız fiziksel aygıtlarmış gibi davranmalarına izin verir.

disk denetleyici [İng.: Disk Controller]
İşletim sistemindeki disk erişimi hakkındaki bilgileri fiziksel diskin anlayabileceği şekle çeviren bir donanım devresidir. Bu soyutlanmış bir tabaka sağlayarak, işletim sisteminin değişik türdeki diskler ile nasıl iletişim sağlaması gerektiğini bilmesi zorunluluğunu ortadan kaldırır, sadece disk denetleyici türlerinin sayısının bilinmesini zorunlu kılar. Çok bilinen *disk denetleyici*ler IDE ve SCSI'dir.

dosya düğümü [İng.: inode]
Bir Unix dosya sisteminde, dosyalar hakkında bilgi saklayan veriyapısıdır. Her bir dosya için bir dosya düğümü vardır ve her bir dosyanın bulunduğu yer ile sistemdeki *dosya düğümü*numarası benzersizdir. Her *dosya düğümü* şu bilgileri içerir: *dosya düğümü*nün bulunduğu aygıt; kilitleme bilgileri; dosyanın türü, kipi ve boyu; dosyaya bulunan bağların sayısı; sahibinin kullanıcı ve grup kimlikleri; son erişim ve düzenleme tarihleri; *dosya düğümü*nün kendisinin en son düzenlediği tarih ve dosyanın disk üzerindeki blok adresi. Bir Unix dizini dosyaların isimleri ile dosya düğümleri arasındaki ilişkiyi sağlar. Şayet **`ls`** komutunu `-i` seçeneğiyle beraber kullanırsanız bir dosyanın *dosya düğümü* numarasını bulabilirsiniz.

dosya sistemi [İng.: filesystem]
İki ayrı amaç ve anlam için kullanılan bir terimdir. Bir sürücü üzerindeki hiyerarşik dosya veya dizin topluluğudur (örneğin; CD–ROM, disket, sabit disk üzerindeki dosyasistemleri.) veya işletim sisteminin dosyaları yazacağı yere karar vermesini sağlayan dosya düğümleri, bloklar ve süper bloklar gibi disk ortamı üzerindeki özel kayıtların oluşturduğu bütündür. Terimin anlamını cümle içindeki kullanımına göre değerlendirin.

düşük seviyeli biçemleme [İng.: low level formatting]
Biçemleme (sayfa: 83) ile eşanlamlıdır, bazan yanlış olarak biçemlemek olarak da bilinen dosya sistemi oluşturmaktan farklılığı belirlemek için MS–DOS ve Windows ortamlarında kullanılır.

fark yedekleme [İng.: incremental backup]
Son *tam yedekleme* (sayfa: 86)den beri dosya sistemi üzerinde değişen şeylerin yedeklenmesi. Bir yedekleme sisteminde düzenli bir şekilde kullanılırsa, bilgileri depolama zamanından ve çabasından önemli ölçüde tasarruf sağlar.

geometri [İng.: geometry]
Bir diskin kaç silindire, kaç kafaya ve her silindirde kaç sektöre sahip olduğu belirtmekte kullanılır. Disk geometrileri genellikle `silindir * kafa * sektör` sayıları olarak verilir ve bu çarpıma bazan kısaca CHS (Cylinders * Heads * Sectors) denir.

gölge parolalar [İng.: shadow passwords]
Unix sistemlerindeki parola dosyaları çok geniş bir kesim tarafından okunabildiği için, kullanıcı hesaplarının şifrelenmiş parolalarını bulundurmaz. Bir `shadow` dosyası herkes tarafından okunamadığı için kullanıcı hesaplarının şifrelenmiş parolaları bu dosyada bulundurulur.

hesap [İng.: account]
Bir Unix işletim sistemi her bir kullanıcıya bir hesap açar. Bu hesaba girebilmek için bir kullanıcı adı ve parola verir. Dosyaların depolanabilmesi için genellikle bir ev dizini tahsis edilir, donanım ve yazılımlara ulaşabilmek için gerekli izinler verilir. Bütün bunların toplamına *hesap* denir.

işletim sistemi [İng.: operating system]
Bilgisayar programlarının yürütülmesini sağlayan ve hata ayıklama, giriş–çıkış denetimi, sıralama, sayışım, derleme, bellek atama, veri düzenleme ve benzer hizmetlerle sistem kaynaklarını (işlemci, bellek, disk alanı, vb.) kullanıcı ve uygulama programları arasında paylaştıran yazılımların oluşturduğu bütündür. Güvenliği sağlamak için sistem erişimini de kontrol eder. Ayrıca *çekirdek* (sayfa: 84) ve *sistem programı* (sayfa: 86)na bakınız.

iz [İng.: track]
Disk dönerken ama kafa sabitken kafanın altından geçen yüzey parçasıdır. Her iz *sektör* (sayfa: 86)lere ayrılır ve her izin “dikey” toplamı bir *silindir* (sayfa: 86)i oluşturur.

kök dosya sistemi [İng.: root filesystem]
Bir Unix dosya sistemindeki bütün dosya sistemlerinin bağlı olduğu en tepedeki dosya sistemidir. "/" ismiyle bağlanır ve diğer sistemler onun üzerine "/usr" şeklinde bağlanır. Şayet kök dosyasistemi bağlanamazsa çekirdek panikler ve *açılış* (sayfa: 83) işlemi devam edemez.

kuyruklama, havuzlama [İng.: spool]
Bir dosyanın bir kuyruğa gönderilmesi. Bu terim IBM tarafından ilk zamanlarda “Simultaneous Peripheral Operation On–Line” sözcüklerinden türetilmiş bir kısaltma olarak sadece paralel porta yönlendirilen dosyaları sıraya sokmak anlamında kullanılmış olmakla beraber, günümüzde bu terim epostaların kuyruklanması (sözcüğün sözlük anlamındaki gibi epostaların gönderilinceye kadar havuzlanması), çiziciler ve diğer çizim aygıtları için işlerin sıraya sokulması gibi farklı işlemler için de kullanılmaktadır. Bu işlemi yapan programlara da kuyruklayıcı (spooler) denmekte ve genellikle bir artalan süreci olarak çalıştırılmaktadır.

mantıksal bölüm [İng.: logical partition]
Ek bölüm içerisindeki bölüm. Gerçekte *mantıksal bölüm* yoktur sadece yazılımın mantıksal yapısı içerisine bulunur.

oku/yaz kafası [İng.: Read–write head]
Bir diskin manyetik yüzeyine yazmak ve okumak için kullanılan ince bir elektromanyetik bobin ve metal nüvedir. Bu alet, plakaların dönme yönüne dik olarak hareket eder.

önyükleme sektörü [İng.: boot sector]
Genellikle herhangi bir atanmış disk bölümünün (diskin değil) ilk sektörüdür. İşletim sisteminin düzgün bir şekilde yüklenmesi ve çalışmasını sağlayacak çok kısa (birkaç yüz bytelık) bir program (*önyükleyici* (sayfa: 85)) içermek üzere ayrılmış özel bir sektördür.

önyükleyici [İng.: Bootstrap loader]
İlk bilgisayarlarda genellikle ROM içerisinde bulunan çok küçük bir programdı. Şimdilerde bir disk üzerindeki sabit/belili bir bölgenin okunup, oradaki programın kontrolü almasına izin verilir. Bu bölge genellikle MBR olur. Bu bölgedeki program genellikle karışık ve büyük hacimlidir ve daha sonra işletim sisteminin seçimi, yüklenmesi ve kontrolü ele alması için gerekli işlemlerin sorumluluğunu alır. Bu programa genel olarak önyükleyici denir. (LILO, GRUB ve benzerleri birer önyükleyicidir.)

parçalanma [İng.: fragmentation]
Bir dosyanın bir disk üzerine ardışık bloklar halinde yazılamaması durumudur. Şayet bir dosyanın tamamını ardışık bloklar halinde yazmak için gerekli boş disk yüzeyi olmaz ise, dosya disk yüzeyinde iki veya üç parçaya ayrılır. Buna parçalanma denir ve bu olay; dosyanın parçalarını aramasını gerektirdiği için, dosyanın yüklenme süresi uzar.

parola dosyası [İng.: password file]
Kullanıcı adlarını ve bu hesap hakkındaki bilgileri tutan dosyadır. Bir Unix işletim sisteminde bu dosya genellikle `/etc/passwd`'dir. Pek çok modern Linux sisteminde bu dosya parolaları ihtiva etmez. Güvenlik nedeniyle başka bir dosyaya (`/etc/shadow`) yönlendirilmiştir. Ayrıntılı bilgi için **passwd**(5) ve **shadow**(5) kılavuz sayfalarına bakınız.

plakalar [İng.: Platters]
Bir sabit disk içindeki fiziksel disklerdir. Genellikle bir sabit disk birbiri üzerine istiflenmiş pek çok fiziksel plakadan oluşur. Her bir bağımsız diske *plaka* denir.

posta aktarım aracısı [İng.: Mail Transfer Agent (MTA)]
Epostaları dağıtmakla görevli programdır. Başka bir MTA'dan mesaj alır veya onu geçici bir yerel bölgeye depolar, kime ait olduğunu analiz eder ve başka bir MTA'ya gönderir. Her durumda bir mesaj başlığı ekleyebilir veya düzenleyebilir. Unix'te genellikle, MTA olarak, **sendmail** kullanılır.

posta istemcisi [İng.: Mail User Agent (MUA)]
Kullanıcıların elektronik posta oluşturmalarını ve okumalarını sağlayan program. Kullanıcı ve MTA arasında arayüzdür. Dışarıya gönderilen mesajlar dağıtılmak için bir MTA'da tutulurken, gelen mesajlar MTA her nereye koydu ise oradan toplanır. MUA için örnek programalara olarak pine, elm, mutt sayılabilir.

sektör [İng.: sector]
Bilgi depolanabilecek en küçük boydaki ize verilen addır. Genellikle (her zaman değil)512 bayttır.

silindir [İng.: cylinder]
Çok kafalı bir sabit disk üzerinde, okuyucu kafaların bir anda üzerinde bulundukları izlerden oluşan göreli bölüm. Diğer bir deyişle plaka milinden eşit uzaklıktaki izler topluluğudur ve bu izler alt alta geldiğinde göreli bir silindir oluşur. Aynı *silindir* üzerine aynı anda ulaşılabilecek şekilde yerleştirilmiş bilgiler, oku/yaz kafasının disk dönüş hızından daha yavaş olmasından kaynaklanan erişim süresi uzunluğunu önemli bir miktarda kısaltırlar.

sistem çağrısı [İng.: system call]
Uygulama programları için çekirdek tarafından sağlanan servisler ve bunları çağırmanın yoludur. Kılavuz sayfalarının 2. bölümü sistem çağrılarını içerir.

sistem programı [İng.: system program]
Bir işletim sisteminde yüksek seviyeli görevleri yerine getiren programlardır. Örneğin donanıma dayanmayan işleri yaparlar. Bazı zamanlar çalışmak için özel durumlara ihtiyaç duyarlar; örneğin elektronik posta dağıtımı gibi. Ama genellikle sistemin bir parçası olarak düşünülebilirler (örneğin: derleyici). Ayrıca *uygulama programı* (sayfa: 87), *çekirdek* (sayfa: 84) ve *işletim sistemi* (sayfa: 85)ne bakınız.

takas alanı [İng.: swap space]
Sistemin bellek yerine kullanabildiği disk yüzeyidir. Bu genellikle ayrılmış bir bölümdür ama bir dosya da olabilir.

tam yedekleme [İng.: full backup]
Bütün bir dosya sisteminin bir yedekleme aygıtına (teyp, disket, CD gibi) kopyalanması.

tek kullanıcılı kip [İng.: single user mode]

Genellikle 1. çalışma seviyesini tanımlar. Sadece root kullanıcı bağlanabilir. Sistem onarımı (dosya sisteminin zarar gördüğü ama hala açılabildiği durumlarda) veya dosya sistemlerinin bölümler arası taşınmaları için kullanılır. Bunlar sadece iki örnektir. Bir sistemde disk üzerine tek bir kişinin yazmasını gerektirecek her durumda tek kullanıcılı kip kullanılır.

uygulama programı [İng.: application programs]
Yazılımlar faydalı işler yaparlar. Bir uygulama programını kullanmanın sonuçları, bilgisayarın alınma sebepleri ile aynıdır. Ayrıca *sistem programı* (sayfa: 86) ve *işletim sistemi* (sayfa: 85)ne bakınız.

yaz saati uygulaması [İng.: Daylight Saving Time]
Enerji tasarrufu için yılın belirli bir döneminde saatlerin bir saat ileri alınmasıdır. Yazın güneş ışığından daha fazla faydalanmak için dünya çapında uygulanan bir yöntemdir.

yazıcı kuyruğu, yazdırma kuyruğu [İng.: print queue]
Yazıcıyı kullanmak isteyen kullanıcıların, yazım işleminin bitmesini beklemeden kendi işlerine devam etmesini sağlayan bir yazıcı *artalan süreci* (sayfa: 83) tarafından kullanılan bir dosya veya dosyalar topluluğu. Ayrıca bir yazıcının pek çok kişiye paylaştırılmasına izin verir.

yerel zaman [İng.: local time]
Yerel bir bölge üzerinde yasalar, standartlar veya geleneklerle belirlenmiş resmî zaman.

yüksek seviyeli biçemleme [İng.: high level formatting]
Bir dosya sisteminin bir diske yazılmasını tanımlamak için kullanılan yanlış bir terimdir. Genellikle Windows ve MS–DOS ortamlarında kullanılır.

B. GNU Özgür Belgeleme Lisansı

GNU Özgür Belgeleme Lisansı ile lisanslanmış belgelerin bu lisansı içermesi gerektiğinden ve bu lisans kendisinin değiştirilmesine izin vermediğinden (buna tercüme de dahildir) lisans hiçbir değişiklik yapılmaksızın burada belgeye eklenmiştir.

(Ç.N. – GNU Özgür Belgeleme Lisansı bu özelliği sebebiyle dili İngilizce olmayan belgelerde kullanmak için uygun değildir; Türkçe belgenize İngilizce bir metin eklemek istemezsiniz, herhalde. Daha özgür –kendinin belgeye eklenmesini zorunlu kılmayan– lisanslar da var. Örneğin "Creative Commons Share Alike" kendinin belgeye eklenmesini zorunlu kılmaması dışında GNU ÖBL'ye hemen hemen eşdeğerdir.)

GNU Free Documentation License

Version 1.2, November 2002

```
Copyright © 2000,2001,2002 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA  02111-1307, USA

Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
```

1. PREAMBLE

   The purpose of this License is to make a manual, textbook, or other functional and useful document *free* in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

2. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world–wide, royalty–free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front–matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front–Cover Texts or Back–Cover Texts, in the notice that says that the Document is released under this License. A Front–Cover Text may be at most 5 words, and a Back–Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine–readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ascii without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard–conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally

available, and the machine–generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

3. VERBATIM COPYING

   You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

   You may also lend copies, under the same conditions stated above, and you may publicly display copies.

4. COPYING IN QUANTITY

   If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front–Cover Texts on the front cover, and Back–Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

   If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

   If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine–readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer–network location from which the general network–using public has access to download using public–standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

5. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front–matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front–Cover Text, and a passage of up to 25 words as a Back–Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front–Cover Text and one of Back–Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

6. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements."

7. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

8. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

9. TRANSLATION

   Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

   If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

10. TERMINATION

    You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

11. FUTURE REVISIONS OF THIS LICENSE

    The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

    Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
  Copyright (C)  year  your name.
  Permission is granted to copy, distribute and/or modify this document
```

```
  under the terms of the GNU Free Documentation License, Version 1.2
  or any later version published by the Free Software Foundation;
  with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
  A copy of the license is included in the section entitled "GNU
  Free Documentation License".
```

If you have Invariant Sections, Front–Cover Texts and Back–Cover Texts, replace the "with...Texts." line with this:

```
with the Invariant Sections being list their titles, with
the Front-Cover Texts being list, and with the Back-Cover Texts
being list.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Notlar

a) Belge içinde dipnotlar ve dış bağlantılar varsa, bunlarla ilgili bilgiler bulundukları sayfanın sonunda dipnot olarak verilmeyip, hepsi toplu olarak burada listelenmiş olacaktır.

b) Konsol görüntüsünü temsil eden sarı zeminli alanlarda metin genişliğine sığmayan satırların sığmayan kısmı ¬ karakteri kullanılarak bir alt satıra indirilmiştir. Sarı zeminli alanlarda ¬ karakteri ile başlayan satırlar bir önceki satırın devamı olarak ele alınmalıdır.

(1) Aslında bu sık sık karşılaşılan bir hatadır. Pek çok kişi çekirdeği işletim sisteminin kendisi sanmaktadır. Bir işletim sistemi düzenlenmiş bir çekirdekten daha fazlasını sunmaktadır.

(B10) http://www.pathname.com/fhs/

(B11) http://www.pathname.com/fhs/

(2) günümüzdeki BIOS'larda bu sınırlama söz konusu değildir. LBA ile 1024 silindir sınırı aşılmıştır. Daha fazla bilgi içim **lilo** kılavuz sayfasına bakınız.

(3) /proc dizini gerçekte disk üzerinde bulunmaz. /proc hakkında */proc dosya sistemi* (sayfa: 20) bölümünde daha ayrıntılı bilgi verilmiştir.

(B15) http://www.linux.doc.org/LDP/nag2/index.html

(4) Ç.N.: Burada entropy ile ifade edilen şey, farenin ekran koordinatları, diskin okunan ya da yazılan sektör numarası, klavyede basılan tuş numarası gibi sayılardır. Olay bir oyunda oyuncunun ganimet toplamasına benzetilebilir.

(5) LBA (Logical Block Addressing – Mantıksal Blok Adreslemesi) desteği olan yeni LILO sürümlerinde, bu 1024 silindir sorunu artık yoktur.

(6) Tabii ki **umount** komutunun aslı unmount'tur. Fakat 1970'li yıllarsa "n" esrarengiz bir şekilde ortadan kayboldu. Şayet "n" yi görürseniz lütfen Bell Laboratuarlarına haber verin.

(7) Yine de bu durum kullanıcının bir kaç saniye düşünmesi ile aşılabilen bir güvenliktir. Ancak, **sudo** komutu, kullanıcıların belirli komutları çalıştırabilmesi için sınırlandırılabilir. Bu konu hakkında ayrıntılı bilgiyi sudo(8), sudoers(5) ve visudo(8) kılavuz sayfalarında bulabilirsiniz.

(8) **debugfs** programı e2fsprogs paketi ile dağıtılır.)

(B27) http://www.go.dlr.de/linux/src/defrag-«0.73.tar.gz

(9) Ç.N.: Günümüzdeki sabit disklerin boyutları gözönüne alındığında takas alanı için MB hesabı yapmak yerine, izin verilen en büyük takas alanını ayırın (2GB). Diskte yere ihtiyacınız olana kadar zaten gerçek takas alanı ihtiyacınızı tespit etmiş olursunuz ve bu takas alanını küçülterek ihtiyaç duyacağınız disk alanını buradan kazanırsınız. RAM'ler için GB'ların, sabit diskler için onlarca gigabaytların konuşulduğu günümüzde ne kadar takas alanı lazım hesabı yapmak abesle iştigaldir. İşin doğrusu budur.

(10) Buna kısaca POST (Power On Self Test), açılışta kendi kendini sınama adı verilir.

(11) Ç.N.: Son çıkan bazı donanımlar açma–kapama düğmesine dokunmadan işletim sistemi tarafından tamamen kapatılabilmektedir. Bunun için kullanılan komut **poweroff**'tur

(12) **init** kendisinin öldürülmesine izin vermez. SIGKILL bile gönderseniz **init**'i öldüremezsiniz.

(13) **kill -HUP 1** komutunu root olarak çalıştırarak **init**'in /etc/inittab dosyasını yeniden okumasını sağlayabilirsiniz.

(B36) http://www.linux.doc.org/LDP/nag2/index.html

(14) Ç.N.: Burası henüz yazılmamış anlaşılan, X ve xdm hakkında yazılmış bir belge var, en azından bir fikir verebilir: XDM Terminal(B37)

(B37) ../howto/xdm-«terminal-«nasil.pdf

(15) Dikkat, **time** komutu o anki zamanı göstermez.

(16) Ayrıntılı bilgi için http://www.time.gov/about.html/ adresini ziyaret ediniz.

(17) Ç.N.: Türkçe yardımlaşabileceğiniz eposta listelerine nasıl erişebileceğinizi http://www.linux.org.tr adresinden öğrenebilirsiniz.

(18) Ç.N.: İletilerinizdeki Türkçe karakterler bazı liste sunucuları tarafından bazı kurallara uymazsanız bozulabilmektedir. Bu kurallara uyarsanız iletilerinizde Türkçe karakterleri herkesin doğru okuyacağını şahsen garanti ederim.

- Asla HTML biçemli ileti göndermeyin.
- Eposta istemcinizin gönderdiği postaların karakter kodlaması eğer ayarlanabiliyorsa, karakter kodlamasını "ISO–8859–9" ya da "UTF–8" seçin.
- Eposta istemcinizin gönderdiği postaların karakter dönüşüm kuralı ayarlanabiliyorsa, "Quoted Printable" ve "base64" kullanmayın. Sadece "8bit"'e izin verin.

Bu özellikleri doğru bir şekilde yapılandırılabilen eposta istemcilerinden benim bildiklerim, Mozilla ve KDE ile gelen KMail.

(B49) http://www.tuxedo.org/~esr/faqs/smart-«questions.html

(19) Ç.N.: Tabii Türkçe konuşulan IRC sunucuları da var. Genellikle belli başlı ISS'lerin IRC sunucularında bir #linux kanalı mutlaka vardır. Herhangi birine girip diğer sunucuların adreslerini de edinebilirsiniz.

(20) Ç.N.: bu nefret kelimesi bence çok hafif kalmış ama yine de orjinal metine uymak gerekiyor.

Bu dosya (sag.pdf), belgenin XML biçiminin TEXLive ve belgeler-xsl paketlerindeki araçlar kullanılarak PDF biçimine dönüştürülmesiyle elde edilmiştir.

27 Şubat 2007