TEMEL
LINUX SİSTEM YÖNETİMİ

KURS NOTLARI

Linux Kullanıcıları Derneği

http://www.linux.org.tr

Burak DAYIOĞLU <burak@linux.org.tr>

Berk DEMİR <berk@linux.org.tr>

Akademik Bilişim 2001 Konferansı

30,31 Ocak 2001, Samsun

ÖNSÖZ

Bu notlar, Akademik Bilişim 2001 Konferansı öncesinde gerçekleştirilecek iki günlük Temel Linux Sistem Yönetimi kursu için Linux Kullanıcıları Derneği (LKD) tarafından hazırlanmıştır.

Kurs içeriği Burak DAYIOĞLU tarafından belirlenmiş, notlar Berk DEMİR ve Burak DAYIOĞLU tarafından yazılmıştır. Kurs içeriğinin hazırlanması esnasında farklı bölümler için destek sağlayan LKD üyelerinin isimleri aşağıdaki listede sunulmuştur; katkılarından dolayı kendilerine teşekkür ediyoruz:

UNIX ve Linux nedir ?	Murathan Bostancı
UNIX Kabuğu ve Özellikleri	Barış Metin
Aygıt Dosyları	Sancar Saran
Bağ Dosyalaru	İlker Manap
Diskler ve Dosya Sistemleri	Devrim Seral
Kullanıcı Kotaları	Selami Aksoy
İleri Tarihe Görev Atama	Selami Aksoy

Kurs notları, gösterilen yoğun çabaya rağmen ancak kurstan birkaç saat önce tamamlanabilmiştir; bu nedenle eksiklerinin ve yanlışlarının olması kaçınılmazdır. Öneri ve eleştirileri yazarlara aşağıdaki e-posta adreslerinden ulaştırabilirsiniz.

Saygılarımızla,

Burak DAYIOĞLU <burak@linux.org.tr>

Berk DEMİR <berk@linux.org.tr>

29 Ocak 2001, Samsun

İÇİNDEKİLER

1. UNIX ve Linux nedir?

UNIX, 1970'li yılların başında AT&T Bell Laboratuvarları'nda geliştirilmesine başlanmış ve bugünkü durumu ile çok kullanıcılı (multi user) ve çok görevli (multi-tasking) hale ulaşmış bir işletim sistemidir. C programlama dilinin aynı yıllarda Dennis Ritchie tarafından bulunmasından sonra, 1973 yılında sistemin tümü bu dil ile yeniden yazılmış ve bu UNIX'in gelişimi açısından bir dönüm noktası oluşturmuştur. Bu yeni sistem, akademik çevrelerde büyük ilgi ile karşılanmış ve büyük bilgisayar sistemlerinin çoğunluğunu elinde bulunduran Amerikan akademik camiasının da desteği ile hızlı bir gelişme trendine girmiştir.

1977 yılında AT&T'nin daha önce ücretsiz olarak uygun gördüğü kuruluşlara sağladığı UNIX ticari bir ürün haline gelmiştir. Bu gelişmeler akademik camiada çalkantılara neden olmuş, University of California, Berkeley'deki iki öğrenci yazılıma ciddi eklentiler yazarak bu eklentilerini Berkeley Software Distribution (BSD) adı ile dağıtmaya başlamışlardır. Sonraki altı yıl içerisinde BSD çok hızlı bir gelişim göstermiş, AT&T UNIX'indeki mevcut bir çok kısıtlamanın kaldırılmasında önemli rol oynamıştır. BSD 4.2 dağıtımı ile birlikte AT&T UNIX'te mevcut olmayan ağ bağlantı desteği sisteme katılmıştır. Tüm bu gerekçeler ile akademik çevrelerde BSD dağıtımı oldukça popüler bir hale gelmiştir.

UNIX, uzun geçmişi nedeni ile oldukça kararlı ve oturmuş bir işletim sistemidir. Bu nedenle özellikle görev-kritik uygulamalarda neredeyse alternatifsizdir. Bir çok kurum ve kuruluş bu nitelikleri nedeni ile bilişim uygulamalarını UNIX ve türevi işletim sistemlerinde sürdürmektedir.

Linux, Finli bir öğrenci olan Linus Torvalds tarafından deney seti olarak geliştirimine başlanmış bir UNIX türevidir. Torvalds, tek disketlik açılış-uyumlu bir sistem ile Intel i386 uyumlu kişisel bilgisayarını başlatmayı deniyordu. Bu çalışmasını başarılı bir biçimde sonuçlandırınca, akademik çevrelerde popüler bir UNIX türevi olan MINIX sisteminden daha iyi bir sistem yazıp yazamayacağı konusunda kendisini sınamaya karar verdi. Oluşturduğu ilk taslakları 1991 yılında Internet üzerindeki muhtelif haber gruplarına göndererek, sistemin geliştirimi için destek istedi. Beklediğinin çok üzerinde destek gören projeye LINUX adı verildi ve Internet üzerinde her geçen gün artan sayıda geliştiricinin desteği ile bugün artık her gazete ve dergide hakkında yazılar görmeye başladığımız sistem durumuna ulaştı.

LINUX projesinin bugünkü başarısının en büyük nedenlerinden birisi Özgür Yazılım Vakfı (Free Software Foundation - FSF) tarafından hamiliği yapılan GNU projesidir. GNU projesi kapsamında yüksek kalitede bir çok geliştirme ve uygulama yazılımı açık kaynak kodu ile birlikte özgürce dağıtılmaktadır. Bu bağlamda, LINUX işletim sisteminin yalnızca çekirdeğini oluşturmaktadır. Sistemin tüm diğer bileşenleri GNU vb. kaynaklardan sağlanan özgür yazılım ürünlerinden oluşmaktadır.

1990'lı yılların ortalarından itibaren LINUX'un yaygınlığının hızla artması ticari yazılım üreticilerini de cezbetmiş ve sayısız ticari yazılım ürünü LINUX üzerinde de çalıştırılabilir duruma gelmiştir.

LINUX, bütün diğer UNIX türevlerinden beklenebilecek gerçek çok görevlilik, sanal bellek kullanımı, TCP/IP ve ağ desteği, bellek yönetimi gibi özellikleri yüksek kalite ile sağlamaktadır. Kaynak kodunun açık olması ve yazılımın özgürce dağıtılması nedeni ile hatalar kısa sürede giderilebilmekte, hatta bazı durumlarda potansiyel hatalar henüz ortaya çıkmadan giderilebilmektedir.

2. Kullanıcı ve Gruplar ile Çalışmak

UNIX işletim sisteminde "Kullanıcı", sistemden hizmet alan her gerçel kişinin sistem üzerindeki temsili karşılığıdır. Temel olarak bir UNIX sisteminden faydalanan her kişi için, bir UNIX kullanıcısı tanımlanmalıdır. Tanımlanan her kullanıcıya birer adet olmak üzere

- **Kullanıcı Adı (Login Name):** Kullanıcının sistem ile ilişkilerinde kendisini tanıtacağı kısa adı
- **Parola (Password):** Kullanıcının kimliğini sisteme ispatlamak için kullanabileceği ve yalnızca ilgili kullanıcı tarafından bilinmesi gereken harf-rakam-sembol karışımı
- **Kullanıcı Kimliği (User ID):** Kimlik kartı sıra numarası gibi düşünülebilecek bir sayı

- **Ana Grup Kimliği (Group ID):** Kullanıcının mensubu olduğu gruplardan öncelikli olanı
- **Ev Dizini (Home Directory):** Kullanıcının sisteme giriş yaptığında çalışmaya başlayacağı, kendisine özel dizin
- **Kabuk (Shell):** Kullanıcının sisteme girişi sonrasında çalıştırılacak ilk program; kullanıcının sistem ile tüm müteakip ilişkileri bu yazılım tarafından yönetilir.

atanır; kullanıcı tanımları sistemin sorumlu yöneticisi ya da yöneticileri tarafından yapılır. Sistem yöneticisi sistemin sağlıklı bir biçimde işletilmesinden ve bakımının yapılmasından sorumlu kişidir. UNIX bağlamında sistem yöneticisi geleneksel olarak `root` kullanıcı adı atanmış ve kullanıcı kimliği ve grup kimliği sıfır (0) olarak seçilmiş kullanıcıdır. `root` kullanıcısı, sistem üzerindeki en yetkili kullancıdır ve diğer kullanıcılar için geçerli olan yetki denetimleri `root` için sistem tarafından geçersiz kılınmıştır.

Sisteme kullanıcı eklemek üzere `useradd` programından faydalanılır. Bu programın kullanımı sistemden sisteme değişiklik göstermektedir; kullandığınız sistemin kılavuz sayfalarından detaylı bilgi edinebilirsiniz (`man useradd`). Örnek sistemimiz üzerinde program aşağıdaki gibi çalıştırılabilir:

```
# adduser -d /home/bdd bdd
```

Kullanıcı ekleme işlemi sırasında, `/etc/skel` dizini altında yer alan dosyaların tümü kullanıcının ev dizinine kopyalanır. Açtığınız her kullanıcı hesabı ile birlikte, kullanıcılarınızın ev dizininde bir takım düzenlemeler yapmak isterseniz `/etc/skel` dizini altında bu genel düzenlemeyi yaparak işlemi otomatik hale getirebilirsiniz.

Kullanıcı silmek üzere `userdel` programı kullanılmaktadır. Programın iki örnek kullanımı aşağıda verilmiştir:

```
# userdel burak (burak kullanıcısı sistemden silinir, ev dizini korunur)

# userdel -r burak (burak kullanıcısı sistemden ev dizini ile birlikte silinir)
```

Silme işleminden ÖNCE, kullanıcıya ait ve kullanıcının ev dizini dışında kalan diğer dosyaların silinmesi için "bul ve sil" işlemini yerine getirecek aşağıdaki komut satırı işletilebilir:

```
# find -user burak -exec rm {} \;
```

UNIX dosya ve dizin erişim denetimlerini grup bazında gerçekleştirir. Aynı dizine ya da dosyaya erişmesi gereken kullanıcı sayısı birden fazla ise bu kullanıcıların bir "grup oluşturduğu" öne sürülür ve bu kullanıcılar için uygun bir UNIX grup tanımının yapılması beklenir. Dosya ve dizinlerin grup bazında erişim denetiminin sağlanması sisteme esneklik getirir.

Bir üniversitenin mühendislik fakültesine bağlı farklı bölümlerinin birlikte kullandığı bir UNIX sistemini düşünelim. Bilgisayar Mühendisliği Bölümü'nün akademik personeli bölüm ile ilgili dosyalara erişmek isterken aynı dosyalara başka kimselerin (bölüm akademik personeli olmayan) erişmesini istememektedir. Benzer biçimde Elektronik Mühendisliği Bölümü personeli ise yalnızca kendi bölüm personelinin kullanımı için tahsis edilecek bir dizin talep etmektedir. Bu durumda doğal gruplama Elektronik Mühendisliği ve Bilgisayar Mühendisliği biçiminde olacaktır.

Grup mekanizması, daha girift problemlere de çözüm oluşturmaktadır: Bilgisayar ve Elektronik Mühendisliği Bölümlerinin ortak bir projesi olduğunu ve bu projenin her bölümden yalnızca ikişer akademisyen tarafından yürütüldüğünü düşünelim. Bu durumda ortak projede görevli çalışanların iki grubun birden üyesi olması gerekmektedir. Projede görevli Elektronik Mühendisliği personeli hem "Elektronik Mühendisliği" hem de "Ortak Proje", görevli Bilgisayar Mühendisliği personeli ise hem "Bilgisayar Mühendisliği" hem de "Ortak Proje" gruplarına üye edilecektir.

UNIX'te yeni bir grup tanımlamak üzere `groupadd` programından faydalanılır. Bu programın kullanımı sistemden sisteme değişiklik göstermektedir; kullandığınız sistemin kılavuz sayfalarından detaylı bilgi edinebilirsiniz (`man groupadd`). Örnek sistemimiz üzerinde program aşağıdaki gibi çalıştırılabilir:

```
# groupadd elektrik
```

Kullanıcı tanımı yapılırken bir kullanıcıyı birden fazla gruba dahil etmek mümkündür. `useradd` programı ile ilgili örnekler arasında bu türden bir örneği bulabilirsiniz. Bir kullanıcının hangi gruplara dahil olduğunu sorgulamak üzere `groups` programından faydalanılır. `groups` programı aşağıdaki gibi çalıştırılır:

```
$ groups burak
users, staff
```

Sistem üzerinde çalışan bir kullanıcı, herhangi bir anda yalnız ve yalnız bir grubun üyesi olarak davranabilir. Sisteme giriş anında, kullanıcının ana grubu olarak belirlenen grup üyesi olarak çalışmaya başladığı varsayılır. Kullanıcı, dilediği an `newgrp` programı yardımı ile aktif grubunu değiştirebilir. Örneğin `burak` kullanıcısı, ana grubu olan `users` grubundan staff grubuna geçiş yapmak için aşağıdaki komut satırını işletmelidir:

```
$ newgrp staff
```

Kullanıcılar, halihazırda bağlı bulunduğu grubu sorgulamak üzere `id` komutunu kullanır; bu komut kullanıcının kimliği ile ilgili bilgileri görüntüler:

```
$ id
uid=500(burak) gid=550 groups=500,550
```

UNIX sistemlerinde kullanıcı tanımlarına ilişkin bilgiler /etc/passwd dosyasında grup tanımlarına ilişkin bilgiler ise /etc/group dosyasında yer alır. Yeni nesil UNIX türevi işletim sistemleri, güvenliği arttırmak amacı ile /etc/passwd dosyası ile birlikte alternatif düzeneklerden de faydalanmaktadır. Linux dağıtımlarının büyük bir kısmı /etc/passwd dosyasının güvenlik ile ilgili hassas kesimlerini /etc/shadow dosyasında saklar.

/etc/passwd dosyasından bir satır aşağıdaki gibidir:

```
burak:J2HS4ks$wer9s:500:500:Burak DAYIOĞLU:/home/burak:/bin/bash
```

Bu satırdaki kolonlar sırası ile kullanıcı adını, şifrelenmiş biçimde kullanıcı parolasını, kullanıcı kimliğini, grup kimliğini, ev dizinini ve kabuk yolunu tanımlar. İstisnasız olarak her kullanıcı için sistem üzerinde bu biçimde bir satır ile gerekli tanım yapılmalıdır; `useradd` programı çok kabaca bu satırların doğru biçimde eklenmesinden sorumludur.

/etc/group dosyasından bir satır aşağıdaki gibidir:

```
staff::500:berk,murathan,burak
```

Bu satırdaki kolonlar sırası ile grup adı, grup erişim parolası, grup kimliği ve son kolonda ise virgül ile ayrılmış biçimde bu gruba dahil kullanıcı adları biçimindedir.

`root` kullanıcısı dilediği kullanıcıya yeni bir parolayı kullanıcının bilgisine dahi gerek olmaksızın atayabilir:

```
# passwd burak
```

Sıradan kullanıcılar, `passwd` programını parametresiz çalıştırmalıdırlar.

3. UNIX Kabuğu ve Özellikleri

Kabuk, işletim sistemlerinde kullanıcı ile işletim sistemi çekirdeği arasındaki iletişimi sağlamak üzere hazırlanmış bir arabirimdir. Örneğin, MS-DOS temelli işletim sistemlerinde `command.com` sistemin kabuğudur; sisteme hemen her türlü komut `command.com` programının sağladığı arabirim ile verilir.

UNIX sistemlerinde herhangi bir kullanıcının sisteme girişi ile birlikte kullanıcı için atanmış kabuk programının işletimi başlatılır ve kullanıcı sistemde çalıştığı sürece bu kabuk programı vasıtası ile sistem ile haberleşir. Kabuk istediğiniz programları çalıştıran, istemlerini uygun biçimde yönlendiren bir arabirim olarak ifade edilebilir.

UNIX altında kullanabileceğiniz çok sayıda ve farklı kabuk olmasına rağmen Bourne Again Shell (`bash`), Linux sistemlerde son derece popülerdir. Sistemdeki her kullanıcının kullandığı kabuk farklı olabilir, herkes kendi zevkine ve ihtiyaçlarına uygun olan kabuğu kullanabilmektedir ve UNIX'in bu özelliği önemli bir esneklik sağlamaktadır.

Bir çok Linux dağıtımı ön tanımlı kabuk olarak bash'i kullanmaktadır. Bu kabuk dışında Bourne Shell (sh), C Shell (csh) ve Korn Shell (ksh) gibi farklı kabukları saymak mümkündür.

Bir kullanıcının kabuğu genel amaçlı yazılmış bir kabuk programı olabileceği gibi, UNIX üzerinde çalışabilir herhangi bir program da olabilir. Örneğin kolaylıkla /usr/bin/pine yoluna sahip PINE e-posta istemcisini bir kullanıcı için kabuk olarak tanımlayabilirsiniz. Bu durumda kullanıcı sisteme girdiğinde yalnız ve yalnız PINE programını kullanabilecek, bu programın sunduğu arabirim ile sistem ile haberleşebilecektir.

Kullanıcılar diledikleri anda herhangi bir programı çalıştırırcasına bir genel amaçlı kabuk programını başlatabilir ve çalışmalarını bu kabuk altında sürdürebilirler. Örneğin bash kabuğunu kullanan bir kullanıcının, daha rahat edeceğini düşündüğü için kısa süreliğine csh kabuğuna geçmesi için aşağıdaki komutu yazması yeterli olacaktır:

```
$ /bin/csh
```

Geçici süre ile csh kabuğuna geçen kullanıcı işi bittiğinde

```
% exit
```

komutunu vererek eski kabuğuna dönebilir. bash kabuğunun yaygınlığını göz önünde bulundurarak, bu kabuk ile ilgili bir takım becerileri ve özellikleri inceleyeceğiz.

Kullanıcının sisteme girişi ile birlikte bash kabuğunun açılış-betikleri (intialization script) işletilir. Bu betikler sırası ile /etc/profile, ~/.bash_profile, ~/.bash_login, ~/.profile dosyaları olacaktır. Sisteme farklı bir kabuk ile giriş yapar ve bash'i daha sonra çalıştırırsanız, okunacak açılış betiği tektir ve ~/.bashrc dosyasıdır.

Dosya yollarının parçası olarak görünen ~ simgesi kullanıcı ev dizinlerini simgelemektedir. Bu anlamda burak kullanıcısının ev dizinine geçmek için

```
$ cd ~burak
```

yeterli olabilmektedir. Açılış betiği tabir edilen dosyalar içerisinde bash'in anlayabileceği biçimde yapılmış tanımlar ve komutlar yer almaktadır. Örnek bir ~/.bashrc dosyası içeriği aşağıdaki gibidir:

```
PATH=$PATH:/home/burak/bin
echo "Selam Burak!"
alias odtuweb lynx http://www.metu.edu.tr
```

İlk satırda PATH çevre değişkeni ile belirlenen yol arama listesine /home/burak/bin dizini eklenmektedir.Bu sayede bu satırın işletilmesinden sonra /home/burak/bin dizininde yer alan programlar tam yol verilmeden çalıştırılabilecektir. İkinci satırda ise echo komutu ile kısa bir karşılama mesajı görüntülenmektedir. Üçüncü satırda ise lynx programı ile ODTÜ web sitesinin görüntülenmesi için bir kısa yol verilmektedir; bu satırın işletilmesinden sonra odtuweb yazılması bağlantının gerçekleştirilmesi için yeterli olacaktır.

bash kullanımı oldukça kolay bir arabirime sahiptir; alıştıktan sonra çok hızlı bir biçimde çalışabilmenize imkan sağlamaktadır. bash'in bu türden özelliklerinin birisi tamamlama özelliğidir. Bir program adının ilk birkaç harfini girebilir ve TAB tuşuna iki defa ard arda basarak bu harfler ile başlayan çalıştırılabilir programların bir listesini görüntüleyebilirsiniz. Eğer bu harflerle başlayan program yalnızca bir tek ise, doğrudan program adı tamamlanacaktır. Sta ile başlayan programların bir listesi aşağıdaki gibi alınabilir:

```
$ sta <TAB><TAB>
startkde    startx      stat        statserial
```

TAB tuşunun kullanımı dosya isimlerinde de aynı şekildedir. Bir komutu girdikten sonra parametre olarak dosya adı gireceğiniz yerde dilediğiniz dosyanın ilk birkaç harfini yazabilir ve gerisini bash'in tamamlamasını sağlayabilirsiniz.

```
$ ls
butunleme.txt   baris.txt   ali.txt
$vi but<TAB>
$vi butunleme.txt
```

Komutun çıktısını bir dosyaya yazdırmak için > ve >> karakterleri kullanılabilir. Örneğin aşağıdaki örneği çalıştırdığınız dizinde `surecler` isimli bir dosya oluşturacak ve bu dosyanın içerisine "`ps auxw`" komutunun çıktısını yazacaktır; ekranda hiçbir çıktı görmeyeceksiniz çünkü bütün çıktıyı dosyaya yönlendirmiş olacaksınız:

```
$ ps auxw > surecler
```

Şimdi `surecler` isimli bir dosyaya sahibiz; bu dosyanın sonuna ekleme yapmak üzere >> karakterinden faydalanırız. Aşağıdaki örnek, `surecler` dosyasının sonuna bir satır eklemektedir:

```
$ echo "--- BİTTİ ---" >> surecler
```

UNIX felsefesinin önemli bir kısmı, küçük ve işlevleri iyi tanımlanmış programların ardı ardına çalıştırılması yolu ile karmaşık işlemlerin gerçekleştirilebilmesidir. Bu sayede, çok farklı ihtiyaçlara cevap vermek için yazılmış devasa programlara ihtiyaç duyulmamaktadır. UNIX araçlarının büyük bir kısmı oldukça az beceriye sahiptir ve bu nedenle öğrenilmeleri kolaydır. Sistem ile deneyiminiz ilerledikçe farklı programlar hakkında da bilgi edinir, bu programları da daha önce öğrendikleriniz ile farklı kombinasyonlarda çalıştırabilirsiniz.

Bu bağlamda, bir programın ürettiği çıktıyı bir diğerine girdi olarak verebiliyor olmak temel ve önemli bir kavramdır. Bu amacı gerçekleştirmek için `|` karakterinden faydalanılır. Çalışan görevler listesinden içerisinde "squid" geçenleri bulmak için aşağıdaki gibi bir komut çalıştırılabilir: Bu örneğimizde `ps` programının çıktısı grep programına girdi olarak verilmektedir.

```
$ ps aux|grep squid
root       619  0.0  0.0  3308    0 ?        SW    2000   0:00 [squid]
squid      622  0.0 18.7 51272 48388 ?       S     2000   3:03 (squid) -D
squid      630  0.0  0.1  1480  496 ?        S     2000   0:00 (dnsserver)
squid      631  0.0  0.1  1480  420 ?        S     2000   0:00 (dnsserver)
```

Daha sonra kullanmak üzere kabuk içerisinde bazı değişkenler (çevre değişkeni) tanımlayabilirsiniz. Değişken tanımlama için bir örnek aşağıda verilmiştir:

```
BEN="Barış Metin"
```

Böylesi bir tanımdan sonra `BEN` adında bir çevre değişkeni tanımlanmış olur. Değişkenin kullanımı esnasında önüne `$` işareti getirilmelidir:

```
echo $BEN
Barış Metin
```

Sisteme giriş yapıldığında bir dizi çevre değişkeni otomatik olarak kabuk tarafından tanımlanır. `printenv` komutu ile tanımlanmış çevre değişkenlerinin tam bir listesi alınabilir.

```
$printenv
HOSTNAME=ozgur.beykent.edu.tr
LOGNAME=baris
TERM=vt100
PATH=/usr/local/bin:/bin:/usr/bin:/usr/X11R6/bin:/usr/sbin:/home/baris/bin
HOME=/home/baris
SHELL=/bin/bash
USER=baris
```

Tanımlanan çevre değişkenlerinin, kabuktan çalıştırılacak tüm diğer programlar tarafından da erişilebilir olması için bu değişkenlerin ihraç edilmesi gerekmektedir:

```
export BEN
```

Sisteme giriş esnasında tanımlanan çevre değişkenlerinden birisi `HOME`'dur. Bu değişken kullanıcı ev dizininin yolunu tutar. Diğer bir önemli değişken `SHELL`'dir; `SHELL` kullanıcı kabuğunun yolunu tutar.

4. Dosya ve Dizin Erişim Denetimleri

UNIX dosya sistemi ikincil saklama ortamlarında yer alan dosyalara erişimi kontrol eder. Hangi kullanıcıların hangi dosyalara ve dizinlere nasıl ulaşabileceğinin kontrolü UNIX dosya sistemi tarafından yapılır. Bu nedenle dosya sistemi, UNIX üzerinde güvenliği sağlamak için denetlenmesi gereken en önemli noktalardan birisidir.

Bir `ls` çıktısında, çıktının ilk sütunu dosya ve dizin erişim haklarını gösterir. Örnek bir `ls` çıktısı aşağıda verilmiştir:

```
$ ls -l
-rw-r--r--   1   bdd   staff   20911     Jan   22   01:21       bsd.c
```

Erişim haklarının ilk karakteri, dosya ya da dizinin tipini belirler. Bu tip aşağıdakilerden birisi olabilir:

-: Sıradan dosya

d: Dizin

c: Karakter Aygıtı (character device)

b: Öbek Aygıtı (block device)

l: Sembolik Bağ (symbolic link)

s: Soket (socket)

p: FIFO

Erişim haklarının sonraki dokuz karakteri, üçerli gruplar halinde sistem üzerinde kimin neyi yapabileceğini belirler. Üç tür hak vardır; okuma (read), yazma (write) ve çalıştırma (execute). Buna paralel olarak üç farklı sınıf vardır; dosyanın sahibi (owner), dosyanın sahibi ile aynı kullanıcı grubundakiler (group) ve bu ikisinin dışında kalan tüm kullanıcılar (other).

Örneğin, dosya erişim hakları `rwxr-xr--` olan bir dosya, sahibi tarafından okunabilir, yazılabilir ve çalıştırılabilir.(`rwx`) Dosyanın sahibi ile aynı kullanıcı grubundakiler dosyayı okuyabilir ve çalıştırabilir (`r-x`). Geride kalan diğer kullanıcılar bu dosyayı sadece okuyabilecektir (`r--`).

Dosya ve dizinlere ilişkin erişim haklarını değiştirmek için `chmod` komutu kullanılır. Dosyanın erişim haklarını yalnızca dosyanın sahibi ya da `root` değiştirebilir; diğer kullanıcılar erişim denetim mekanizmasını herhangi bir biçimde geçersiz kılamaz. `chmod` programının kullanımına dair bir örnek aşağıda verilmiştir:

```
$ ls -l bsd.c
-rw-r--r--   1   bdd   staff   20911     Jan   22   01:21       bsd.c
$ chmod go-r bsd.c
-rw-------   1   bdd   staff   20911     Jan   22   01:21       bsd.c
```

chmod programına verilen ilk parametre, hangi sınıfa hangi yetkinin verileceğini belirler. Örnekte, grup (`g`) ve diğerlerinden (`o`) okuma (`r`) yetkisi geri alınmaktadır (`-`). Sonuç, örnekte görüldüğü gibi yalnızca sahibi tarafından okunabilen ve değiştirilebilen, başka herhangi bir biçimde herhangi birisi tarafından erişilemeyen bir dosya olacaktır. `chmod` programı ile bir dizin altında yer alan tüm dosya ve dizinlerin erişim haklarının bir seferde değiştirilmesi mümkündür; bu durumda `-R` parametresi kullanılmalıdır.

Bir dosyanın sahipliğini değiştirmek üzere `chown` programından faydalanılır. Kullanıcı dilediği taktirde aşağıdaki gibi bir komutla bu işlemi gerçekleştirebilir:

```
# ls -l bsd.c
-rw-r--r--   1   bdd   staff   20911     Jan   22   01:21       bsd.c
# chown burak bsd.c
-rw-r--r--   1  burak  staff   20911     Jan   22   01:21       bsd.c
```

Bir dosyanın bağlı bulunduğu grubu değiştirmek üzere `chgrp` programından faydalanılır. Kullanıcı dilediği taktirde aşağıdaki gibi bir komutla bu işlemi gerçekleştirebilir:

```
# ls -l bsd.c
-rw-r--r--   1   bdd   staff   20911     Jan   22   01:21       bsd.c
# chgrp users bsd.c
-rw-r--r--   1   bdd   users   20911     Jan   22   01:21       bsd.c
```

Anılan standart erişim denetim ayarlarının dışında genişletilmiş bir erişim denetim sınıfı daha mevcuttur. Bu sınıfta SUID, SGID ve Sticky ikilleri mevcuttur.

Dosya sistemi üzerinde yer alan bir program çalıştırıldığında işletim sistemi tarafından bir sürece (process) dönüştürülür. İşletime alınan süreç, programı çalıştıran kullanıcının yetkileri ile donatılır. Örneğin `root` kullanıcısı tarafından çalıştırılan bir program sahibinin yetkileri nedeni ile çalıştığı süre boyunca sistem üzerinde her türlü yetkiyi elinde bulundurur. Sıradan bir kullanıcı tarafından aynı program çalıştırıldığında, bu kullanıcının sistem üzerindeki yetkileri programa aktarılır.

Ancak bu yetkilendirme düzeneği belli durumlarda sistemi zora sokmaktadır. Örneğin parola değişikliği için tüm kullanıcılar tarafından `passwd` programının çalıştırılabilmesi gerekmektedir. `passwd` programı, `/etc/passwd` dosyası üzerinde değişiklik yapmak zorunda olduğu için `/etc/passwd`'nin sahibi olan `root` kullanıcısının yetkilerine ihtiyaç duyar. Özel durumu nedeni ile bu programın çalışmaya başladığı anda `root` kullanıcısının yetkileri ile çalışması (programı başlatan her kim olursa olsun) lazımdır. Bu gibi durumlar için set user-id (SUID) isimli erişim hakkından faydalanılır. Bu hakkın verildiği `/bin/passwd` programının `ls` çıktısı aşağıda verilmiştir:

```
$ ls -l /bin/passwd
-r-s--x--x    1 root     root        13536 Jul 12  2000 /usr/bin/passwd
```

Bir programa SUID hakkı vermek üzere `chmod` programı ile `u+s` hak belirleyicisi kullanılabilir:

```
# chmod u+s /bin/bash
-rwsr-xr-x    1 root     root       512540 Aug 22 19:46 /bin/bash
```

Yukarıdaki komutun `root` tarafından çalıştırılması halinde, komutun işletilmesinden sonra sisteme giriş yapan her kullanıcı `root` yetkileri ile donatılmış olacaktır. Bu örnek, bir üretim sisteminde denenmemeyi gerektirecek kadar tehlikelidir.

SUID ile benzer biçimde, bazı programların çalışmaları süresince belli bir kullanıcı grubunun üyesi olarak çalışması gerekli olabilir. Örneğin yazıcı hizmetleri ile ilgili yazılımlar bu türden haklara ihtiyaç duyarlar. Bir programa SGID hakkı vermek üzere `chmod` programı ile `g+s` hak belirleyicisi kullanılabilir:

```
# chmod g+s /bin/bash
-rwxr-sr-x    1 root     root       512540 Aug 22 19:46 /bin/bash
```

Tüm kullanıcıların yaz-boz tahtası olarak kullanabileceği dizin `/tmp`'dir. Kullanıcılar tarafından işletilen tüm programlar geçici dosyalarını bu dizinde yaratır ve işleri bittiğinde siler. Ancak bu dizine tüm kullanıcıların okuma ve yazma hakkı ile erişmesi durumunda kullanıcıların bu dizini hepten silebilmesi mümkün olacaktır. Kullanıcıların yalnızca kendilerine ait dizinleri ve dosyaları silebilmesini sağlamak üzere `/tmp` dizini Sticky erişim hakkı ile donatılmıştır:

```
$ ls -ld /tmp
drwxrwxrwt   11 root     root         4096 Jan 29 11:48 /tmp
```

Bir dizine sticky erişim hakkı vermek üzere `chmod u+t` hak belirleyicisi kullanılabilir.

5. Dizin Sıradüzeni ve Yapısı

UNIX dosya sistemi tek köklü bir ağaç yapısı biçimindedir; MS-Windows türevi işletim sistemleri bunun aksine çok köklü ağaç yapısında dosya sistemlerine sahiptir (c:, d:, e: ayrı birer köktür). UNIX'te tek bir kök vardır ve `/` sembolü ile gösterilir. Tüm dosyalar ve dizinler kök dizin referans alınarak işaretlenir. Tüm ağaç

tek bir dosya sistemi üzerinde olabileceği gibi, ağacın bazı dalları farklı dosya sistemleri üzerinde de depolanıyor olabilir.

Dizin hiyerarşisi içerisinde yer alan dizinlerden bir kısmı hemen her UNIX türevi için özel anlamlar ifade etmektedir. Tipik bir Linux sistemine ilişkin önemli dizinler ve bu dizinde depolanan dosya türleri aşağıdaki gibidir:

/tmp: Geçici dosyalar dizini

/lib, /usr/lib: Kitaplık dosyaları dizini

/include, /usr/include: C Başlık (header) dosyaları

/sbin,/usr/sbin: Sistem programları dizini

/bin, /usr/bin: Uygulama programları

/etc: Sistem ayar dosyaları

/var: Kuyruk dizinleri (e-posta, yazıcı vb.) ve kayıt (log) dosyaları

/usr/local: Yerel uygulamalar ve bileşenleri

/boot: Çekirdek ve açılış dizini

/proc: Çekirdek izleme ve ayar dizini

/dev: Aygıt dosyaları dizini

6. Aygıt Dosyaları

UNIX, programlama arabirimini olabildiğince kolaylaştırmak üzere tüm aygıt erişimlerini bir dosya benzetim yöntemi ile gerçekleştirmektedir. Bu benzetim sayesinde, programcı aygıt ile ilişkilendirilmiş dosyayı açar ve bu üzerinde dilediği gibi okuma ve yazma işlemlerini gerçekleştirir. İşlemler esnasında programcı eriştiği aygıtın denetim detaylarını bilmek zorunluluğundan kurtulmuş olur.

Aygıt dosyaları /dev dizini altında yer alır. Bu dizin altında yer alan dosyalar bilinen anlamı ile birer dosya değildir; yalnızca erişimlerin aygıta yönlendirilmesini sağlamak üzere bulunan birer yer tutacağıdır. Aşağıdaki örnekte bir grup aygıta ilişkin `ls` çıktısı görülmektedir:

```
$ ls -l /dev/hda*
brw-rw----    1 root     disk       3,   0 Sep 27 13:31 hda
brw-rw----    1 root     disk       3,   1 Sep 27 13:31 hda1
brw-rw----    1 root     disk       3,   2 Sep 27 13:31 hda2
```

Aygıt dosyaları öbek aygıt dosyaları ve karakter aygıt dosyaları olarak iki gruba ayrılır. Öbek aygıt dosyaları, öbek türü aygıtlara erişim için kullanılır. Bu aygıtlar giriş/çıkış işlemleri için tampon bellek kullanan aygıtlardır. Bu biçimde çalışan aygıtlar giriş/çıkış yapılan verileri bir öbek tamamen doluncaya kadar tampon bellekte tutar ve daha sonra bu öbeği bir seferde aktararak işlemi gerçekleştirirler. Bu türden aygıtlara disk ve teyp türü aygıtlar örnek gösterilebilir.

Karakter aygıt dosyaları, karakter (harf/sembol) bazında iletişim kurulan aygıtlar için kullanılırlar. Seri bağlantı noktasından bağlı yazıcılar bu türden bir örnek olarak görüntülenebilir.

7. Bağ Dosyaları *(Link Files)*

Sistem tarafından depolanan ve işlenen her dosya bir dosya adı ile ilişkilendirilir. Yeni bir dosya tanımlandığında bu dosya için bir de isim atanır. Kullanıcılar, bir dosyaya farklı konumlardan daha rahat erişebilmek için kısayollar tanımlamak ihtiyacı duyabilirler. UNIX bağlamında kısayol belirtmek için kullanılan dosyalar bağ dosyaları olarak anılırlar.

İki tür bağ dosyası mevcuttur; hard-link ve soft-link. Hard-link türü bağ dosyaları sistem üzerinde yalnızca sistem yöneticisi tarafından yaratılabilirler. Var olan bir dosyaya bir hard-link ile ikincil bir isim atandığı andan itibaren dosyaya iki farklı isim ile erişmek mümkün olacaktır. Bu aşamadan sonra isimlerden birisi ile gelecek bir dosya silme talebinde sadece dosyaya erişim için kullanılan iki isimden birisi silinmiş olur, dosyanın saklanması sürdürülür ancak artık yalnızca bir isim ile erişmek mümkün olacaktır. Bu durumda dosya yalnızca ve yalnızca kendisi ile ilişkilendirilen tüm isimleri için birer dosya silme işlemi gerçekleştikten sonra silinmiş ve hiçbir biçimde ulaşılamaz olacaktır.

UNIX sistemleri üzerinde bağ dosyaları `ln` programı yardımı ile tanımlanır:

```
# ln /raporlar/ocak-satis.txt  /cok-satanlar/2001-ocak.txt
```

Yukarıdaki örnekte, `/raporlar/ocak-satis.txt` yolu ile erişilen dosyaya erişmek için ikincil bir yol olarak `/cok-satanlar/2001-ocak.txt` tanımı yapılmaktadır. Bu noktadan sonra dosya isimlerinden birisi silinse de diğeri de silinene kadar dosya var oluşunu sürdürecektir.

Aşağıdaki örnekte php.ps adı ile verilen dosyaya bir hard-link oluşturulmaktadır:

```
# ls -l
total 239
-rw-r--r--   1 root     root       242783 Dec 13 10:50 php.ps
# ln php.ps php-dokuman.ps
# ls -l
total 478
-rw-r--r--   2 root     root       242783 Dec 13 10:50 php-dokuman.ps
-rw-r--r--   2 root     root       242783 Dec 13 10:50 php.ps
```

Görüldüğü gibi, hard-link türü bağın (`php-dokuman.ps`) oluşturulmasından sonra ls çıktısının ikinci kolonunda yer alan referans sayısı ikiye çıkmıştır. Dosyanın farklı adları için silme talebi geldikçe referans sayısı azalır ve sıfır olduğunda dosya sistemden silinir.

Sistem üzerindeki tüm kullanıcılar tarafından tanımlanabilen ve daha kolayca anlaşılıp idare edilebilen bağ dosyası türü soft-link'lerdir. Soft-link türü bir bağ, mevcut bir dosyaya farklı yollardan da ulaşılabilmesini sağlar. Ancak hard-link'lerden farklı olarak, kendisine ikincil isimler tanımlanan dosya için ilk ismine gelen bir silme talebi dosyanın tümü ile silinmesine yol açacak, ikincil isimler ile dosyaya verilen referanslar anlamsız olacaktır. soft-link türü bir bağ `ln` programına `-s` parametresi verilerek gerçekleştirilir. Aşağıdaki örnek soft-link'lerin kullanımını göstermektedir:

```
$ ln -s /raporlar/ocak-satis.txt /cok-satanlar/2001-ocak.txt
$ rm /raporlar/ocak-satis.txt
$ cat /cok-satanlar/2001-ocak.txt
No Such File or Directory
```

Örnekte, ilk satırda `/raporlar/ocak-satis.txt` yolu ile erişilen dosyaya erişmek için ikincil bir yol olarak `/cok-satanlar/2001-ocak.txt` tanımı yapılmaktadır. Ancak ikinci satırdaki dosyanın ilk adı için gelen silme talebinden sonra `/cok-satanlar/2001-ocak.txt` biçiminde verilen ikincil dosya adı ile dosyaya erişmek mümkün olamayacaktır; çünkü artık dosya tümüyle silinmiştir.

```
$ ln -s /home/httpd/html html
$ ls -l html
lrwxrwxrwx 1 webmaste users 16 Dec 13 08:56 html -> /home/httpd/html/
```

Yukarıdaki iki komutun sistem üzerindeki web sitesinin sorumlusu tarafından verildiğini düşünelim. Web sorumlusu ilk komutu kendi ev dizininde verdiğinde web sayfalarını düzenlemek için sisteme her girişinde gitmesi gereken `/home/httpd/html` dizinine kestirme bir yol tanımlamış olacaktır. Sisteme girişini yaptıktan sonra `cd html` diyerek `/home/httpd/html` dizinine daha az tuş vuruşu ile ulaşabilecektir. İkinci satırda yer alan `ls` komutunun çıktısında, dosya erişim haklarının görüntülendiği ilk kolonun ilk harfi olan "l" harfi, dosyanın bir bağ (link) dosyası olduğunu göstermektedir. Dosya adı kolonundaki ok işareti de dosyanın hangi dosyaya yönelmiş bir bağ olduğunu açıkça göstermektedir.

8. Süreçler ve Süreç Yönetimi

Sistem üzerinde depolanan bir program dosyası, işletime geçtiğinde işletimin başladığı andan sonlandığı ana değin geçen sürede sistem için işletilmesi gereken bir "süreç" olarak anılır. UNIX çok kullanıcılı ve çok görevli bir işletim sistemidir. Birden fazla kullanıcıya ait birden fazla süreç birbirine paralel olarak işletiliyor olabilir. Herhangi bir anda, genellikle sistemin sahip olduğu ana işlem birimi (CPU) sayısının çok üzerinde sürecin paralel olarak işliyor olması beklenir; ancak bir ana işlem birimi herhangi bir anda yalnız ve yalnız bir süreci işletiyor olabilir. Örneğin paralel işlemesi gereken yirmi süreç varsa ve bu süreçlere ev sahipliği yapan UNIX sisteminin dört işleyicisi var ise herhangi bir anda en çok dört süreç aktif olabilecek, diğer süreçler beklemek zorunda kalacaktır.

Modern işletim sistemlerinin büyük bir çoğunluğu bu problemi, zaman bölüştürme adı verilen bir teknik ile aşmaktadırlar. Bu teknik çerçevesinde her süreç uygun bir düzen çerçevesinde sıra ile uygun ana işlem birimlerinden birisine işletilmek üzere verilmekte, saniyeden kısa bir sürede bu sürecin işletimi beklemeye alınarak ana işlem birimine yeni bir sürecin işletimi görevi atanmaktadır. Her bir sürecin kısa sürelerde çalıştırılması ve bunun bir döngü biçiminde gerçekleştirilmesi sayesinde, kullanıcılar için bir yanılsama gerçekleştirilmiş olur; kullanıcılar çok sayıda sürecin paralel işletildiğini zannederler.

Sistemde işletilen her sürece bir süreç numarası atanır. Süreç ile ilişkili her türlü işlemde bu süreç numarası referans olarak kullanılır. Sistem üzerinde çalışan süreçlerin listelenmesi için `ps` (süreç durumu - process status) programından faydalanılır. Linux için örnek kullanımı ve çıktısı aşağıdaki gibidir:

```
$ ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.6   1292   524 ?        S    10:56   0:05 init [3]
root         2  0.0  0.0      0     0 ?        SW   10:56   0:00 [kflushd]
root         3  0.0  0.0      0     0 ?        SW   10:56   0:00 [kupdate]
root         4  0.0  0.0      0     0 ?        SW   10:56   0:00 [kpiod]
root         5  0.0  0.0      0     0 ?        SW   10:56   0:00 [kswapd]
root         6  0.0  0.0      0     0 ?        SW<  10:56   0:00 [mdrecoveryd]
root       287  0.0  0.6   1352   576 ?        S    10:56   0:00 syslogd -m 0
root       297  0.0  0.8   1648   728 ?        S    10:56   0:00 klogd
root       390  0.0  0.3   1316   344 ?        S    10:56   0:03 gpm -t ps/2
root       447  0.0  1.2   2208  1064 tty1     S    10:56   0:00 login – root
root      1202  0.0  1.4   2308  1272 pts/1    S    13:41   0:00 login -- bdd
bdd       1203  0.0  1.5   2312  1368 pts/1    S    13:41   0:00 -bash
```

Görüntülenen listede ilk kolonda süreci başlatan kullanıcı kimliği, ikinci kolonda süreç numarası ve sondan bir önceki kolonda da sürecin şimdiye değin kullandığı toplam ana işlem birimi zamanı yer almaktadır. Bu kolonda görüntülenen süre, sürecin işletimine başlanmasından `ps` sorgusunun işletildiği ana kadar geçen gerçek zamanı göstermez; görüntülenen sistemin bu sürecin işletimi için harcadığı net zamandır.

STAT başlığı ile verilen kolonda, süreçlerin halihazırdaki durumları görüntülenir. Kolonda, süreç durumlarına ilişkin tek harflik kısaltmaların bir kombinasyonu mevcuttur ve kısaltmalar ve açıklamaları aşağıdaki listedeki gibidir:

R: Süreç işletilebilir durumda

T: Süreç durdurulmuş durumda

P: Süreç diskten bir sayfa yüklemesi bekliyor

D: Süreç diskten bilgi yüklenmesini bekliyor

S: Süreç 20 saniyeden az bir süredir uyuyor

I: Süreç boşta (20 saniyeden fazla bir süredir uyuyor)

Z: Süreç sonlandı ancak süreç ölümü gerçekleşemedi (Zombi)

W: Süreç, diske aktarıldı (swapped-out)

>: Süreç kendisine uygulanan bellek limitini aştı

N: Süreç düşük öncelikle işletiliyor

<: Süreç yüksek öncelikle işletiliyor

Zombi süreçler sistem kaynaklarını kullanmazlar, bu nedenle sistem üzerinde çok sayıda zombi sürecin bulunması endişe edilecek bir durum değildir.

Bazı durumlarda işler durumda olan bir sürece müdahale gerekli olabilir. Temel müdahaleler arasında sürecin işleyişine son vermek ve süreci geçici olarak durdurmak sayılabilir. Bu amaçlar için kullanılmak üzere UNIX sistemlerinde "sinyal mekanizması" mevcuttur. Sinyaller süreçlere çok ilkel bir biçimde hükmedilebilmesine imkan vermektedirler.

UNIX sistemlerinde tanımlı onun (10) üzerinde sinyal vardır. Hangi sinyalin alınması durumunda ne biçimde davranılacağı tümü ile işleyen süreci geliştiren programcının seçimine başlıdır. Temel olarak bir süreç, aldığı bir sinyal sonrasında aşağıdaki davranışlardan birisini sergileyebilir:

Sinyali Görmezden Gelme: Süreç, sanki hiç sinyali almamışçasına işlemeyi sürdürebilir.

Toparlanma ve Çıkış: Süreç, sinyali aldığında sürdürdüğü işi hızla toparlar ve çıkar.

Dinamik Yapılandırma: Süreç, sinyali aldığında yapılandırma dosyalarını yeniden okuyacak çalışma düzeni ayarlarının son durumuna göre kendisini ayarlar; yazılımın durdurulup tekrar başlatılmasından kurtulunmuş olur.

İç Durum Raporlaması: Sinyali alan süreç, işleyiş durumuna ilişkin bir takım bilgileri bir rapor biçiminde dökebilir.

Hata Ayıklamayı Durdur/Başlat: Süreç, sinyali aldığında hata ayıklama kipine geçer ya da bu kipten çıkar

Hangi davranışın sergileneceği, daha önce de açıklandığı gibi, tümü ile yazılımı geliştiren kişinin/ekibin seçimidir. Sıkça kullanılan bazı sinyaller ve genel olarak bu sinyallere yüklenen anlamlar aşağıdaki gibidir:

HUP: Dinamik yapılandırma amacı ile yoğun biçimde kullanılır

TERM: Süreç işletiminin toparlanıp sonlandırılması sağlanır

KILL: Süreç işleminin toparlanmaya dahi müsaade edilmeden acilen sonlanması sağlanır

Süreçlere sinyal gönderebilmek üzere `kill` programından faydalanılır. Aşağıdaki örnekte 87 süreç numarasına sahip `inetd` sürecine HUP sinyali gönderilmektedir:

```
# kill -HUP 87
```

Sinyali alan süreç programcısının seçimi doğrultusunda sinyal sonrasında gereğini yapar. İşleyen bir süreci durdurmak üzere `kill` programı `-TERM` parametresi ile ya da parametresiz olarak çalıştırılmalıdır. Aşağıdaki iki satır tümü ile aynı işi yapmaktadır:

```
# kill -TERM 87
# kill 87
```

Tüm sinyallerin bir listesini edinmek üzere `kill -l` komut satırı işletilebilir.

9. Diskler ve Dosya Sistemleri

Bu bölümde diskler, disk bölümleri, dosya sistemleri ve bu kavramların UNIX ile ilişkisi irdelenecektir. Bir disk, veri depolama amacı ile kullanılan herhangi bir tür ortamdır. Disk bölümü (partition), bir diskin farklı amaçlar ile kullanımı için mantıksal anlamda bölünmesi işlemi sonucunda ortaya çıkan parçaların her birine verilen addır.

Bir dosya sistemi, en az bir disk bölümü üzerine yerleşen ve dosya ve dizinlerin belirli bir sistematik içerisinde depolanması için gerekli kataloglama ve erişim denetimi temel altyapı hizmetlerini sağlayan yapıdır. Zaman içerisinde ihtiyaçlar geliştikçe ve/veya değiştikçe çok farklı dosya sistemleri geliştirilmiş ve kullanılmıştır, ancak temel olan hemen her tür dosya sisteminin amacının bir biçimde depolanan dosyaların kataloglama ve erişim denetimi süreçlerini gerçekleştirmek olduğudur. İhtiyaçlar doğrultusunda her işletim sistemi ile uyumlu çalışabilen çok farklı dosya sistemlerinin olabilmesi mümkün olmaktadır; Linux işletim sistemi onun üzerinde farklı dosya sistemi ile uyumlu bir biçimde çalışabilmektedir.

Ext2 adı verilen dosya sistemi, Linux dağıtımları tarafından en yoğun biçimde kullanılan dosya sistemidir. Geleneksel UNIX dosya sisteminin tüm özelliklerini ve bir takım gelişmiş özellikleri içinde barındırmaktadır. Bu gelişmiş özellikler arasında yüksek performans için bir takım düzenlemeleri ve veri kaybını önlemek için alınmış tedbirleri saymak mümkündür.

Bir Ext2 dosya sistemi 4TB'lık (TeraBayt) bir büyüklüğe kadar genişleyebilmekte ve 255 harfe kadar uzun dosya isimlerine destek sunmaktadır.

Linux tarafından desteklenen bir dosya sistemi ailesi Microsoft MS-DOS ve Windows ailesi işletim sistemleri tarafından kullanılan dosya sistemleridir. FAT ve VFAT dosya sistemleri MS-DOS ve Windows 95/98 için geliştirilmiş birer dosya sistemidir. FAT sisteminde dosya isimleri 8+3 kuralına uygun biçimde verilebilmektedir; uzun dosya isimleri desteklenmez. VFAT sisteminde ise uzun dosya isimleri için destek sisteme eklenmiş durumdadır. Windows NT tarafından kullanılan NTFS dosya sistemi de Linux tarafından desteklenen dosya sistemleri arasındadır.

ISO9660 dosya sistemi CD-ROM'larda kullanılmak üzere geliştirilmiş bir sistemdir ve ISO9660 dosya sistemi de Linux tarafından desteklenmektedir.

Modern dosya sistemlerinin hemen tümü "transaction" temellidir. Journaling File System adı verilen bu tür sistemler elektrik kesintisi vb. durumlarda veri kaybını en aza indirmek üzere tasarlanmıştır ve gelişmiş veritabanı yönetim sistemleri tarafından kullanılan biçimde işlem kayıtları temelli çalışmaktadırlar. Bu türden dosya sistemlerinin kararlılığı, diğerleri ile mukayese edilemeyecek kadar yüksektir. Linux için geliştirilen Reiser FS, IBM'in JFS'i, SGI'ın XFS'i ve Ext3 bu kategoride Linux için çalıştırılabilir durumdaki dosya sistemleridir.

Linux dağıtımlarının büyük bölümü diskler üzerinde disk bölümlendirilmesinin gerçekleştirimi ve dosya sistemlerinin tanımlanması için `fdisk` programının kullanılmasına imkan verir. Bu programı kullanarak disk bölümlerinin ve disk bölümleri üzerine yerleşen dosya sistemlerinin idaresi sağlanır. Programı parametresiz başlattığınızda

```
# fdisk
Usage: fdisk [-l] [-b SSZ] [-u] device
E.g.: fdisk /dev/hda  (for the first IDE disk)
  or: fdisk /dev/sdc  (for the third SCSI disk)
  or: fdisk /dev/eda  (for the first PS/2 ESDI drive)
  or: fdisk /dev/rd/c0d0  or: fdisk /dev/ida/c0d0  (for RAID devices)
```

biçiminde kullanımına ilişkin bilgiler görüntülenecektir. Disk bölümlerinin ve dosya sistemlerinin her türlü yönetim yetkisi yalnızca sistem yöneticisine aittir; bu nedenle `fdisk` programı sistem yöneticileri dışında herhangi bir kullanıcı tarafından çalıştırılamaz.

Fdisk ile IBM PC uyumlu sistemimizde "Primary IDE Master" durumundaki sabit diskin üzerinde işlemler yapmak üzere uygun komut satırı

```
# fdisk /dev/hda
```

biçiminde olmalıdır. Programı başlattıktan sonra fdisk komut satırı görüntülenir:

```
Command (m for help):
```

Komut satırında iken yapılabileceklerin ilki mevcut disk bölümlerinin ve dosya sistemlerinin bir listesinin alınmasıdır. Bu amaçla `p` (print) komutu kullanılır:

```
Command (m for help): p
```

```
Disk /dev/hda: 255 heads, 63 sectors, 1582 cylinders
Units = cylinders of 16065 * 512 bytes
   Device Boot    Start       End    Blocks   Id  System
/dev/hda1   *         1       590   4891761   83  Linux
/dev/hda2           591       609    145585   82  Linux swap
/dev/hda3           610      1582   7815622+   f  Win95 Ext'd (LBA)
/dev/hda5           610      1096   3911796    b  Win95 FAT32
/dev/hda6          1097      1340   1959898+   b  Win95 FAT32
Disk /dev/hda: 255 heads, 63 sectors, 1340 cylinders
Units = cylinders of 16065 * 512 bytes
```

Yukarıdaki çıktıda disk üzerindeki bölümler ve bu bölümler üzerine yerleştirilmiş dosya sistemleri görüntülenmektedir. İlk kolonda disk bölümünün aygıt dosyası olarak adı, ikinci kolonda bu bölümde yer alan dosya sisteminin açılışın gerçekleştirileceği- dosya sistemi olup olmadığı, beşinci ve Blocks olarak adlandırılan kolonda bu disk bölümünün öbek cinsinden boyu ve son kolonda da bu bölüm üzerine yerleşen dosya sisteminin türü görüntülenmektedir.

Yeni bir disk bölümü tanımlamak üzere fdisk komut satırında `n` (new) komutunun kullanılması gerekir:

```
Command (m for help): n
Command action
      e   extended
      p   primary partition (1-4)
```

Bu ekranda yaratılacak olan disk bölümünün tipi sorulmaktadır.. Yaratılacak olan bölüm sayısı 4 veya daha az ise hepsi birincil (primary) bir disk bölümü olabilir.Ancak dörtten fazla bölüm yaratılacaksa o zaman extended bölüm tipini kullanmak gerekir. Extended tipinde yaratılan disk bölümlerinin numaraları 5'den başlayarak artmaktadır.

Birincil bir bölüm yaratmak üzere verilecek yanıt "p" olmalıdır.. Bu girişin ardından bölüm numarası sorulur:

```
Partition number (1-4):
```

Bu alana boş bölüm numaralarının birisinin atanması yeterli olacaktır. Bölüm numarasının girilmesinden sonra diskin büyüklüğüne göre değişmek ile birlikte aşağıdaki gibi bir ekran görüntülenecektir:

```
First cylinder (1-1340, default 1):
Last cylinder or +size  or +sizeM or +sizeK (1-1340, default 1340):
```

İlk satırda yeni bölümün başlayacağı silindir numarası, ikinci satırda ise bölüm boyu ya da bölümün son silindiri sorulmaktadır. Bölüme belirli bir büyüklük atamak ikinci satıra üzere aşağıdaki örneklerde görülen biçimde girişlerin yapılması mümkündür:

5MB'lık bir bölüm için +5000K

500MB'lık bir bölüm için +500M

5GB'lık bir bölüm için +5000M

Değer girildikten sonra disk bölümümüz tanımlanmış olacaktır. `fdisk` programı ile tanımlanan her disk bölümünün varsayılan dosya sistemi `ext2` olacaktır (Linux yeni tanımlanan her bölümün ext2 türünde olacağını VARSAYAR; bu davranışı değiştirmek mümkündür). Disk bölümü üzerinde tanımlanan dosya sisteminin türünü değiştirmek üzere `t` (type) komutu kullanılır. Yoğun biçimde kullanılan dosya sistemlerine ilişkin tür numaraları aşağıdaki gibidir:

82: Linux swap

83: Linux (ext2)

Mevcut bir disk bölümünü silmek için `d` (delete) komutu kullanılır:

```
Command (m for help): d
Partition number (1-4):
```

`fdisk` silme istemi sonrasında silinecek bölüm numarasını sorar ve belirtilen bölümün silinmesini sağlar.

UNIX dizin hiyerarşisinin temelini oluşturan kök dizininin üzerinde bulunduğu dosya sistemine kök dosya sistemi adı verilir. Bu dosya sisteminin üzerinde bulunduğu disk bölümünün açılışın yapılacağı disk bölümü olarak işaretlenmesi gerekmektedir. Bu işleme `fdisk` bağlamında aktivasyon denir. Bir disk bölümünü aktif hale getirmek üzere `fdisk` komut satırında `a` komutu kullanılır:

```
Command (m for help): a
Partition number (1-4):
```

Hangi bölümün aktif hale getirileceği seçildikten sonra bölüm aktif olarak işaretlenir. Aktif bölüm aşağıdaki gibi bir çıktı üzerinde Boot kolonunda `*` ile işaretlenmiş olacaktır:

```
Command (m for help): p
Disk /dev/hda: 255 heads, 63 sectors, 1582 cylinders
Units = cylinders of 16065 * 512 bytes
   Device Boot    Start       End    Blocks   Id  System
/dev/hda1   *         1       590   4891761   83  Linux
/dev/hda2           591       609    145585   82  Linux swap
/dev/hda3           610      1582   7815622+   f  Win95 Ext'd (LBA)
/dev/hda5           610      1096   3911796    b  Win95 FAT32
/dev/hda6          1097      1340   1959898+   b  Win95 FAT32
Disk /dev/hda: 255 heads, 63 sectors, 1340 cylinders
Units = cylinders of 16065 * 512 bytes
```

Görüntülenen örnek disk üzerinde /dev/hda1 bölümü aktif olarak işaretlenmiştir.

`fdisk` ile bir disk üzerinde yapılan değişikliklerin sonunda, değişikliklerin kalıcı duruma getirilmesi için `w` (write) komutu kullanılır. Herhangi bir anda `fdisk`'ten `q` (quit) komutu ile çıkılabilir. Çıkış komutu verildiğinde henüz kalıcı duruma getirilmemiş tüm değişiklikler iptal edilmiş varsayılır.

Bir disk üzerinde yeni bir disk bölümü ve ilgili dosya sistemi tanımlandıktan sonra bu dosya sisteminin, işletim sisteminin kullanımına hazır duruma getirilmesi gerekmektedir. Bu işlem sıkça "formatlama" olarak anılır. Yeni bir dosya sistemini kullanıma hazır duruma getirmek için kullanılan program `mkfs` programıdır. Örneğin /dev/hda5 disk bölümünün tanımlanmasından ve ext2 türü bir dosya sisteminin bu bölüm üzerinde çalışacak biçimde düzenlemesinin `fdisk` ile yapılmasından sonra `mkfs` programı aşağıdaki gibi çalıştırılmalıdır:

```
# mkfs /dev/hda5
```

Bu komut işletildiğinde sistem birkaç dakika süre ile dosya sisteminin tanımlanması için çalışacak ve bu işlemin sonrasında dosya sistemi kullanılabilir duruma gelecektir.

Sistem ana belleğinin yeterli olmadığı durumlarda kullanılmak üzere, belleği olduğundan daha büyükmüşçesine kullanmak üzere takas (swapping) işleminden faydalanılır. Ana belleğin az kullanılan bölümleri geçici olarak diske saklanır ve diğer süreçlerin kullanımı için yer açılması sağlanır. Diske saklanan bellek bölgesinin gerekli olması durumunda başka bir bölüm diskteki ile takas edilir. Linux işletim sistemi çalıştıran sistemlerde bu türden bir kullanım için takas dosya sistemi (swap file system) vardır. Takas dosya sistemi türünde bir dosya sistemi tanımlanmasından sonra dosya sisteminin kullanılabilir hale gelmesi için `mkswap` programından faydalanılır; `mkfs` takas dosya sistemleri için uygun değildir. Örneğin /dev/hda3 disk bölümü üzerinde bir takas dosya sistemi tanımlandığında

```
# mkswap /dev/hda3
```

komutu işletilerek dosya sisteminin kullanıma hazır hale getirilmesi sağlanabilir.

`mkfs` programı yeni bir dosya sistemi tanımlandığında yalnızca bir kez çalıştırılmalıdır. Üzerinde bilgilerin bulunduğu bir dosya sistemi için bu programın çalıştırılması dosya sistemlerinde depolanan verilerin tümüyle silinmesi anlamına geleceğinden dikkatli davranılmalıdır.

Kök dosya sistemi yada `/` Linux işletim sistemi için çok önemli bir yere sahiptir. Sistemin kök dosya sisteminin açılış esnasında bağlanamaması durumunda sistemin açılışı mümkün olmaz. Kök dosya sisteminin dolması durumunda sistemin davranışı kestirilemez.

Sistemde tanımlı dosya sistemlerinin ve bu dosya sistemlerine ilişkin seçimlerin bulunduğu yapılandırma dosyası `/etc/fstab` yolundadır. Doğal olarak diğer dosya sistemlerinin açılış esnasında bağlanabilmesi için bu dosyanın kök dosya sistemi üzerinde yer alması gerekmektedir. Dosyanın örnek içeriği aşağıdaki gibidir:

```
/dev/hda1       /              ext2        defaults          1 1
/dev/hda2       /home          ext2        defaults          1 2
/dev/hda3       swap           swap        defaults          0 0
/dev/cdrom      /mnt/cdrom     iso9660     noauto,owner,ro   0 0
/dev/fd0        /mnt/floppy    auto        noauto,owner      0 0
```

Bu tablonun ilk kolonu aygıt dosyasını belirler. İkinci kolon, bu dosya sisteminin dizin hiyerarşisinin hangi dalına bağlanacağına işaret eder. Üçüncü kolonda bu bölümde yer alan dosya sisteminin tipi belirlenir. Dördüncü kolonda belirtilen dosya sisteminin "bağlanması" sırasında geçerli kılınacak seçenekler virgül ile ayrılmış biçimde sunulur. Son iki kolon ise dosya sistemi sınanması esnasında etkinleştirilecek seçenekleri belirler.

Bir dosya sistemi ancak bağlama işlemi sonrasında erişilebilir. Bağlı dosya sistemlerinden birisinin sistemden "ayrılmasından" sonra bu dosya sistemine yeniden bağlanmasına değin ulaşmak mümkün olmayacaktır. UNIX'te bağlama işlemi `mount` programı yardımı ile gerçekleştirilir. `mount` programının kullanımına ilişkin örnekler ve açıklamaları aşağıda verilmiştir:

```
# mount /dev/hda5 /digerdisk
```

Bu komut ile `/dev/hda5` yolundaki disk bölümü içinde yer alan dosya sistemi `/digerdisk` yolundan erişilebilecek duruma getirilmiştir. Bu dosya sisteminin içeriğini incelemek için `/digerdisk` dizinine geçmek yeterli olacaktır.

```
# mount /dev/cdrom /mnt/cdrom
```

Bu komut ile sisteme bağlı CD-ROM okuyucusuna takılı CD'nin içeriğine `/mnt/cdrom` dizininden ulaşılabilecektir.

```
# mount -t vfat /dev/hda2 /mnt/dos
```

`/dev/hda2` yolundaki disk bölümü içerisinde yer alan VFAT türü dosya sistemi (`-t vfat` ile belirtilmektedir) `/mnt/dos` yolu ile erişilebilir kılınmaktadır.

```
# mount -o ro /dev/hda5 /digerdisk
```

`/dev/hda5` yolundaki disk bölümü içerisinde yer alan dosya sistemi salt okunur (`-o ro` ile belirtilmektedir) `/digerdisk` yolu ile erişilebilir kılınmaktadır.

Bağlı bir dosya sistemini ayırmak üzere `umount` programı kullanılır. Programa parametre olarak ayrılacak dosya sisteminin bağlama noktasını ya da ayrılacak dosya sisteminin üzerinde bulunduğu disk bölümünü temsil eden aygıt dosyasının yolunun verilmesi yeterli olacaktır: Örneğin

```
# mount -t vfat /dev/hda2 /mnt/dos
```

komutu ile bağlanan bir dosya sistemini ayırmak için aşağıdaki komutların herhangi birisinin çalıştırılması yeterli olacaktır:

```
# umount /dev/hda2
# umount /mnt/dos
```

Sistemden ayrılması istenen bir dosya sistemi üzerinde çalışan herhangi bir sürecin var olması durumunda, dosya sistemi sistemden ayrılamaz.

Dosya sistemlerinin bütünlüğünü sınamak üzere UNIX sistemlerinde `fsck` (file system check) programı bulunmaktadır. Bu program sistemin hatalı biçimde kapanması, elektrik kesintisi vb. durumlarda oluşabilecek dosya sistemi tutarsızlıklarını tespit etmek ve mümkün olduğu durumlarda da onarmak için kullanılır. Örnek kullanımı aşağıdaki gibidir:

```
# fsck /dev/hda3
```

`fsck` ile dosya sistemleri üzerinde denetim yaparken dikkat edilmesi gereken nokta, denetim yapılacak dosya sisteminin ya sisteme bağlanmamış olması ya da salt-okunur biçimde bağlanmış olması gereğidir. Hata ile okunur-yazılır biçimde bağlanmış bir dosya sistemi üzerinde `fsck` ile denetim yapılması durumunda dosya sistemi bütünlüğü bozulabilir.

Dosya sistemlerinin doluluğuna ilişkin bilgiler `df` (disk free) programı yardımı ile edinilebilir. Parametre olarak bir dosya sistemi verilebilir; parametre verilmemesi durumunda bağlı tüm dosya sistemlerine ilişkin boş alan bilgileri bir rapor biçiminde sunulur:

```
# df
Filesystem           1k-blocks      Used Available Use% Mounted on
/dev/hda1              1011928    734300    226224  76% /
/dev/hda2               806368       548    764856   0% /home
```

Dördüncü (Available olarak işaretlenmiş) kolonda dosya sistemlerinin boş alanları Kilobayt (KB) cinsinden verilmiştir.

Linux işletim sistemini daha etkin bir şekilde kullanabilmek için mevcut disklerin uygun biçimde disk bölümlerine ve dosya sistemlerine bölünerek dağıtılması gerekmektedir. Bu konuda, sistemin kuruluşu öncesinde strateji belirlenmeli ve bu stratejiye göre dağılım gerçekleştirilmelidir.

Ev kullanıcıları için mevcut ana bellek ile aynı büyüklükte bir takas dosya sisteminin tanımlanması ve ayrılabilecek tüm diğer alanın tek bir dosya sistemi biçiminde kök dosya sistemi olarak atanması uygun olacaktır. Sistem tek bir kullanıcı tarafından kullanılacağı için farklı dosya sistemlerine bölme konusunda özel bir çaba sarfedilmesine gerek yoktur.

Herhangi bir biçimde sunucu olarak kullanılacak bir sistemin (dosya sunucusu, e-posta sunucusu, uygulama sunucusu, web sunucusu vb.) öncelikle geçici dosyalar için kullanılan dosya sisteminin /tmp ayrılması gereklidir. Bu dizinin kök dosya sistemi üzerinde bulunması durumunda kök dosya sisteminin tümüyle dolması mümkün olacaktır ve böyle bir durumda sistemin işlevselliğini koruması beklenemez.

Sunucu olarak kullanılacak sistemlerde /var ve /usr dizinleri kaçınılmaz olarak ayrı dosya sistemlerine yerleştirilmelidir. Verilecek hizmetin özelliğine göre farklı dizinlerin de ayrı birer dosya sistemine yerleştirilebilmesi öngörülebilir. Örneğin Apache ile web hizmetleri veren büyük ölçekli bir sunucuda /usr/local/apache dizini için de ayrı bir dosya sistemi tanımlanması düşünülebilir.

10. Init Süreci ve Çalışma Düzeyleri

Init programı UNIX sistemleri için özel bir önem taşır. Bu program, sistemin açılışı sürecinde bir (1) numaralı süreç olarak işletime alınır ve sistemin kapanışına değin çalışmasını sürdürür. Bu süreç, açılış esnasında başlatılacak açılış betiklerini ve sonrasında işletimi denetlenecek süreçleri yönetir.

Bir sistemin çalışma düzeyi, kullanıcılara sunulan hizmetlerin ve kaynakların belirli bir kümesini tanımlar. Bir çalışma düzeyinde `X` hizmeti çalışıyor iken, bir diğerinde aynı hizmet çalışmıyor olabilir. Herhangi bir anda bir sistem ancak ve ancak bir çalışma düzeyinde bulunabilir. Linux sistemlerinde yedi farklı çalışma düzeyi tanımlanmıştır:

0	Sistem durdurulmuş	Sistemin güç anahtarının kapatılmasına uygun hali
1	Sistem yönetimi durumu	Kullanıcı girişlerine açık sistem bakımı durumu; bu durumda ağ hizmetleri çalışmamaktadır
2	Çok kullanıcılı durum	NFS hariç tüm ağ hizmetlerinin işler durumda olduğu çok kullanıcılı durum
3	NFS destekli çok kullanıcılı durum	NFS dahil tüm ağ hizmetlerinin işler durumda olduğu çok kullanıcılı durum
4	KULLANILMIYOR	KULLANILMIYOR
5	X Window destekli çok kullanıcılı durum	Sistem X Window pencereleme hizmeti sunmaktadır
6	Yeniden başlatma	Sistem önce çalışma düzeyi 0'a indirilir ve daha sonra yeniden başlatılması sağlanır

Sistemin herhangi bir anda hangi çalışma düzeyinde olduğunu öğrenmek üzere `runlevel` -programı ile sorgulanabilir, program sonuç olarak önce bir önceki çalışma düzeyini daha sonra ise güncel çalışma düzeyini görüntüleyecektir:

```
# runlevel
N 3
```

Sistemin çalışma düzeyi yalnızca sıfır (0) düzeyinde iken sorgulanamaz, bu düzeydeki bir sistem yalnızca güç anahtarı ile kapatılabilir.

Sistemin çalışma düzeyi `init` programı vasıtası ile değiştirilebilir:

```
# init 1
```

Sistem açılışında ya da `init` programı ile çalışma düzeyinin değişmesi esnasında, `/etc/inittab` dosyasının içeriği bu yazılım tarafından incelenerek işlenir. Bu dosya içerisinde `init` için kritik öneme sahip üç bilgi yer alır:

a. Sistemin açılışta varsayılacak çalışma düzeyi

b. Çalıştırılacak, takip edilecek ve durduğunda yeniden başlatılacak programlar

c. Sistem yeni bir çalışma düzeyine geçtiğinde çalıştırılacak programlar

Bu dosyada yer alan her satır `:` işareti ile ayrılmış dört temel bileşenden oluşur.

```
kimlik:çalışma düzeyi listesi:çalıştırma biçimi:süreç
```

Kimlik kolonunda işletilecek süreç için seçilmiş ve `/etc/inittab` dosyasında biricik olması sağlanmış bir tanımlayıcı bulunur. Birden fazla satırın kimlik kolonu aynı olmamalıdır. İkinci kolonda, belirtilen sürecin hangi çalışma düzeylerinde işletileceğinin bir listesi yer alır. Son kolonda sürecin başlatılmasında kullanılacak komut satırı belirtilmektedir. Üçüncü kolon olan çalıştırma biçimi aşağıdakilerden birisi olabilir:[1]

respawn: Çalışır durumda değil ise süreci başlat, durursa yeniden başlat

wait: Süreci başlat ve sonlanana değin bekle

once: Süreci bir seferliğine başlat; sonlanmasını bekleme, sonlandırsa da yeniden başlatma

sysinit: Bu süreci sistem konsollarını aktif hale getirmeden önce işletmiş ol ve sonlanmasını bekle

[1] Liste eksiksiz değildir, sadece sıkça kullanılan türleri göstermek için verilmiştir.

powerfail: Güç kaynağının kötü durumda olduğuna ilişkin sinyal aldığında işlet, sonucunu bekleme

powerwait: Güç kaynağının kötü durumda olduğuna ilişkin sinyal aldığında işlet, sonucunu bekle

initdefault: Bu satırda belirtilen çalışma düzeyinin varsayılan açılış çalışma düzeyi olmasını sağla

Örnek `inittab` satırları aşağıda verilmiştir:

```
acilis:3:initdefault:
si:S:sysinit:/etc/rc.d/rc.S
rc:23456:wait:/etc/rc.d/rc.M
c1:1235:respawn:/sbin/agetty 38400 tty1 linux
```

11. Sistem Açılış ve Kapanış Süreçleri

"Sistem açılış süreci", işleyicinin (CPU) elektrik ile beslenmeye başlaması ile başlar ve bir işletim sisteminin yüklenerek kullanıcı görevlerini kabul edebilir hale ulaşmasına kadar devam eden sürece verilen isimdir. IBM PC uyumlu sistemlerde bu süreç kapsamında işleyicinin beslemesinin sağlanması ile birlikte `BIOS` olarak bilinen açılış ROM' unda yer alan programın işletilmesi sağlanır. `BIOS` programının işletilmesi ve bu program tarafından sistem sınamalarının tamamlanmasından sonra, açılışa uygun cihazlar taranır ve `BIOS` ayarları çerçevesinde öncelik verilen ilk cihazdan başlayarak bir açılış kayıdı yüklemeye çalışılır. Açılışın yapılacağı cihaz, IDE ya da SCSI sabit disk, CDROM ünitesi ya da disket olabilir.

Linux dağıtımlarının büyük bir kısmında açılış kayıdını `LILO` adı verilen Linux yükleyicisi oluşturur. `BIOS`, `LILO`'yu yüklendikten ve başlattıktan sonra `LILO` kullanıcısının karşısına (arzu ediliyor ise) açılış mönüsünü getirir ve kullanıcıdan tercihlerini alır. Bu tercihler doğrultusunda `LILO`, Linux çekirdeğini yükler ve çekirdeğin başlatılmasını sağlar.

Çalışmaya başlayan çekirdek, kullanacağı kök dosya sistemini bilir; kök dosya sisteminin hangisi olacağı çekirdeğin içerisinde belirtilmektedir. Çekirdek, kök dosya sistemini salt okunur biçimde bağlar ve `init` programını yükleyerek çalıştırır. `Init` programı çalışmaya başladığında, sistem üzerinde başlatılan ilk UNIX sürecidir ve bir (1) numaralı süreç olarak kayıt altına alınır. `init` süreci, önceki bölümde anlatılan biçimde `/etc/inittab` ayar dosyasında belirlenen "sistem açılışı" işlevlerini yerine getirir. Çok kabaca, takas dosya sistemlerinin başlatılması ve kök dosya sisteminin `fsck` programı aracılığı ile tutarlılığının sınanması `init` için sistem açılışı işlevselliğini oluşturur. Tutarlılık sınaması esnasında bulunan hatalar otomatik olarak ya da bunun başarılamadığı durumlarda operatör desteği ile gerçekleştirilir. Hata sınamasının gerçekleştirilmesinden sonra kök dosya sistemi bu kez "yazılır-okunur" olarak yeniden bağlanır. Ve `/etc/inittab` dosyasında belirlenen "varsayılan açılış düzeyi'"ne uygun açılış betiklerinin çalıştırılmasına geçilir.

`/etc/rc.d/init.d` dizininde sistem açılışında başlatılabilecek hizmetler için birer betik bulunur. Bu betikleri çalıştırarak hizmetlerin başlatılması ya da durdurulması mümkün olacaktır. Bu dizinin bir listesi aşağıdakine benzer:

```
$ ls /etc/rc.d/init.d
alsasound       dhsd       kdcrotate       netfs       pvmd        snmpd
sobeled         functions  keytable        network     random      sshd
anacron         gpm        killall         nfs         rawdevices  syslog
apmd            halt       kudzu           nfslock     rhnsd       xfs
arpwatch        httpd      linuxconf       pcmcia      sendmail    xinetd
atd             identd     nylogd          portmap     single      ypbind
crond           ipchains   lpd             postgresql  smb
```

Bu betikler ile ilgili hizmetleri başlatmak için parametre olarak `start`, durdurmak için ise `stop` verilmelidir:

```
# /etc/rc.d/init.d/httpd stop
# /etc/rc.d/init.d/httpd start
```

ÖrneğinYkarıdaki iki satır ile web sunucu sisteminin durdurulması ve başlatılması mümkün olacaktır.

İşte bu dizinde yer alan betikler, dolaylı olarak `init` süreci tarafından işletilir. Çalışma düzeyine göre açılışta çalıştırılacak betiklerin bir listesi `/etc/rc.d/rc3.d` ve `/etc/rc.d/rc2.d` gibi adlandırılmış dizinlerde yer alır. Bu dizinlerde adı `S` ile başlayan betikler bu çalışma düzeyine geçişte, `K` ile başlayan betikler ise bu açılış düzeyinden bir başka açılış düzeyine geçerken işletilirler. Çalışma düzeyine geçişte ve çalışma düzeyinden çıkışta işletilen betikler sayı sırası ile işletilirler. Örneğin `/etc/rc.d/rc3.d` dizininde yer alan `S40network` `S10kudzu` betiklerinden önce `S10Kuzdu` betiği işletilecektir.

12. Yazılım Yönetimi ve RPM

Modern işletim sistemlerinin tümü, yeni yazılımların sistem üzerine birer "paket" olarak yüklenebilmesine imkan veren düzenekler içermektedirler. Bir paket, yazılım ile ilgili tüm dosyaları içerdiği gibi yazılıma ilişkin dosyaların hangi dizinlere ne biçimde yerleştirileceğini ve mevcut sistem ayarlarında ne gibi değişiklikler yapılması gerektiğine ilişkin bilgileri de içerir.

Kişisel bilgisayarları üzerinde MS-Windows işletim sistemini çalıştıranların kolayca anımsayabileceği gibi bu işletim sisteminin Denetim Masası altında "Program Ekle/Kaldır" seçeneği yer almaktadır. Yazılımların sisteme kurulması ve/veya sistemden kaldırılması bu program aracılığı ile gerçekleştirilir. Paket yöneticisi sayesinde "`X` kurulmadan `Y`'nin kurulmasına izin verme" biçiminde kuralların tanımlanabilmesi mümkün olmaktadır.

Yeni nesil UNIX'lerin hemen hepsi bir paket yöneticisi içermektedir. Paket yöneticisi temel olarak yeni yazılımların kolayca yüklenmesini, kurulu paketlerin bütünlüğünün sınanmasını ve artık gereksinim duyulmayan paketlerin sistemden silinmesini kolaylaştıracak bir yazılımdır. Bir sisteme paket halinde kurulacak yazılımların büyük bir bölümü `/usr` dizinine kurulmak isteyecektir. Bu nedenle paket yöneticisinin sistem yöneticisi dışında bir kullanıcı tarafından kullanılması ve yazılım kurulması genellikle mümkün olmaz.

Red Hat Linux ve bu dağıtımdan türetilen dağıtımlarda (Mandrake, Immunix vb.) Red Hat Package Manager (RPM) isimli paket yöneticisini kullanır. Bu yazılım kullanılarak paketlenmiş tüm yazılımlar `.rpm` uzantısına sahiptir. RPM sayesinde web üzerinde, ftp üzerinde ya da bir disk üzerinde yer alan paketlerin kurulması mümkün olmaktadır; paketlerin sistem ile uyumluluğu ya da uyumsuzluğu RPM tarafından denetlenebilmektedir.

Komut satırından `rpm` programının farklı parametreler ile çalıştırılması yolu ile farklı amaçlar gerçekleştirilebilir. KDE ve Gnome grafik arayüzleri ile birlikte RPM'in daha kolay kullanılabilmesine izin veren görsel arabirime sahip uygulamalar da dağıtılmaktadır.

Sistem üzerinde kurulu tüm paketlerin listelenmesini sağlamak üzere aşağıdaki komutun verilmesi yeterli olacaktır:

```
# rpm -qa
```

Adı belirli bir paketin sistemde kurulu olup olmadığını öğrenmek üzere `rpm`'i `grep` ile birlikte aşağıdaki gibi kullanmak mümkündür:

```
# rpm -qa|grep apache
```

Disk üzerindeki bir paketi sisteme kurmak için aşağıdaki gibi bir komut uygun olacaktır:

```
# rpm -i apache-1.3.14-i386.rpm
```

Ftp ya da web üzerinden bir paketi kurmak üzere paket adının yerine paket URL'si yazmanız yeterli olacaktır:

```
# rpm -i ftp://ftp.linux.org.tr/deneme/apache-1.3.14-i386.rpm
# rpm -i http://www.linux.org.tr/deneme/apache-1.3.14-i386.rpm
```

Kurulu bir paketi sistemden kaldırmak üzere aşağıdaki gibi bir komut yeterli olacaktır:

```
# rpm -e apache
```

Kurulu bir paket ile ilgili bilgi almak üzere aşağıdaki gibi bir komut satırı kullanılmalıdır:

```
# rpm -qi apache
```

Henüz kurulmamış bir paket ile ilgili bilgilere erişmek üzere aşağıdaki gibi bir komut satırı uygun olacaktır:

```
# rpm -qpi apache-1.3.14-i386.rpm
```

13. Kullanıcı Kotaları

Çok sayıda kullanıcıya hizmet veren bir sistemde kullanıcıların dosya sistemleri üzerinde depolayabileceği bilgi miktarının sınırlanması gereklidir; aksi durumda dikkatsiz ya da art niyetli bir kullanıcı dosya sistemini sonuna değin doldurabilir. Depolama alanı sınırlaması UNIX sistemlerinde çokça "kota" olarak anılır.

Kullanıcı ya da grup bazında kota tanımları dosya sistemi bazında verilebilir. Bu bağlamda, `/home` ve `/calisma` iki ayrı dosya sistemi ise, ikisi için de ayrı kota tanımlaması gerekli olabilir. Aksi halde `/home` dosya sistemi için tanımlanan kotalar `/calisma` dosya sistemi için geçerli olamayacaktır.

Kullanıcı ve gruplar için kotaları tanımlamadan önce ilk yapılması gereken kota desteğinin sisteme verilmesidir. Sistem ile birlikte gelen standart çekirdek çok yüksek ihtimalle kota desteğini içeriyor olacaktır. İçermemesi durumunda kota destekli yeni bir çekirdeğin hazırlanması gerekli olacaktır.

Dosya sistemi bazında kotaları aktif hale getirmek üzere `/etc/fstab` dosyası içerisinde küçük değişikliklerin yapılması gerekecektir. Kullanıcı kotalarını aktif hale getirmek için `usrquota`, grup kotalarını aktif hale getirmek için ise `grpquota` sözcükleri dosya sistemlerine ilişkin parametre kolonuna eklenmelidir:

```
# cat /etc/fstab

/dev/sda1          /                ext2     defaults,usrquota    1 1
/dev/sda3          /home            ext2     defaults,usrquota    1 2
/dev/sda4          swap             swap     defaults             0 0
none               /proc            proc     defaults             0 0
```

Örnekte, / ve /home dosya sistemleri için kullanıcı kotaları aktif hale getirilmiştir. Dosya üzerinde bu değişiklikler yapıldıktan sonra dosya sistemleri ya ayrılıp yeniden bağlanmalı ya da sistemin yeniden başlatılması sağlanmalıdır.

Bu işlemin ardından kotaları aktif hale getirmek için gerekli son adım olan quotacheck programı işletilmeli ve kota kayıtlarını tutacak dosyaların yaratılması sağlanmalıdır. Komut için uygun parametreler ile birlikte ilgili satır aşağıdaki gibi olmalıdır:

```
# quotacheck -auvg
```

Bu komutun işletiminden sonra kotanın aktif hale getirildiği dosya sistemlerinin kök dizinlerinde (örnekte `/` ve `/home`) `quota.user` ve/veya `quota.group` isimli dosyalar oluşacaktır.

Kotaların aktif hale getirilmesinden sonra `quotaon` programı ile kota düzeneğinin geçerli kılınması, `quotaoff` ile ise geçersiz kılınması sağlanabilir. Komut satırları aşağıdaki gibi olmalıdır:

```
# quotaon -auvg
# quotaoff -auvg
```

Kota düzeneğini açılışta otomatik başlatmak üzere `quotaon` programının uygun parametreler ile açılış betiklerinden birisinin içerisine yazılması önerilmektedir. RedHat Linux dağıtımında quotaon programı açılışta ön tanımlı olarak başlatılmaktadır.

Bir kullanıcının belli sürelerde kota sınırını esnetmesi gerekebilir. Örneğin 20MB'lık bir kota atadığınız kullanıcının üç günlüğüne kotasını 25MB'a esnetebiliyor olması her defasında kota arttırım talebi ile sistem yöneticisine gitmesini engelleyebilir. Bu esnekliği sağlamak üzere üç kavramdan faydalanılır: Esnek (soft) kota, katı (hard) kota ve esnetme süresi (grace period). Kullanıcılara atanan esnek kotalar, esnetme süresi kadar bir süre boyunca katı kota ile belirlenen sınıra değin aşılabilir.

Örneğin 5MB esnek kotası, 10MB katı kotası olan ve esnetme süresi 7 gün olan bir kullanıcı, 5MB'lık kotasını doldurduktan sonra 7 günü aşmamak kaydı ile 10MB'a kadar veri depolayabilir. Ancak kullanıcı 7

günlük esnetme sürecinin sonunda esnek kota sınırının altına inmediği taktirde dosya sistemine herhangi bir biçimde yazma hakkına sahip olmayacaktır.

Varsayılan esnetme süresi ayarlarını gözlemlemek ve değiştirmek üzere `edquota` programı aşağıdaki gibi çalıştırılmalıdır:

```
# edquota -t
Time units may be: days, hours, minutes, or seconds
Grace period before enforcing soft limits for users:
/dev/sda3: block grace period: 7 days, file grace period: 0 days
/dev/sda2: block grace period: 7 days, file grace period: 0 days
```

Verilen örnekte `/dev/sda3` ve `/dev/sda2` dosya sistemleri yedi günlük esnetme süreleri ile kullanılacak biçimde ayarlanmıştır.

Kullanıcı bazında esnetme süresi ayarlarını değiştirmek için `edquota` aşağıdaki gibi başlatılmalıdır:

```
# edquota -t selami
```

Kullanıcı bazında kota atamak ve atanan değeri güncellemek için `edquota` aşağıdaki gibi işletilmelidir:

```
# edquota selami
Quotas for user selami:
/dev/sda3: blocks in use: 153588, limits (soft = 10000, hard = 12000)
                    inodes in use: 3359, limits (soft = 0, hard = 0)
/dev/sda2: blocks in use: 8568, limits (soft = 10000, hard = 12000)
                    inodes in use: 7, limits (soft = 0, hard = 0)
```

Verilen örnekte `selami` kullanıcısı için her iki dosya sistemi üzerinde de 10MB'lık esnek ve 12MB'lık katı olmak üzere kotalar tanımlanmıştır.

Kotaları düzenlemek için özellikle bir düzenleyici yazılımı tanımlamadıysanız `vi` programı başlatılacaktır. Bu program ile rahat çalışamıyor olmanız durumunda aşağıdaki gibi iki satırın işletilmesini sağlayacak `pico` programının kullanılmasını sağlayabilirsiniz:

```
# EDITOR=/usr/bin/pico
# export EDITOR
```

Sistemdeki kotalar ile ilgili istatistiksel bilgilere erişmek üzere `repquota` programı kullanılır. Tüm dosya sistemleri ile ilgili kota raporu edinmek için aşağıdaki komut satırı uygun olacaktır:

```
# repquota -a
Block limits                File limits
User            used    soft    hard  grace    used  soft  hard  grace
root       --  175419       0       0          14679     0     0
bin        --   18000       0       0            735     0     0
uucp       --     729       0       0             23     0     0
man        --      57       0       0             10     0     0
selami     --   13046   15360   19200            806  1500  2250
burak      --    2838    5120    6400            377  1000  1500
```

14. Sistem Bilgileri ve Başarımı

Sistem yöneticisinin sistem kaynakları ve genel durumu hakkında bilgi alabilmesi için bir çok kaynak mevcuttur.

`/proc` dosya sistemi eğer çekirdekten destek verilmişse sistem her açıldığında oluşturulur. Sanal bir dosya sistemi olan `/proc`, sistem yöneticisinin bazı çekirdek parametrelerini çalışma sırasında değiştirebilmesine ve sistemin genel durumu ile ilgili olarak bilgi alabilmesine yardımcı olur.

`/proc` dizini altında yer alan bazı dosyalar ve içerikleri aşağıdaki listede verilmiştir:

`/proc/cpuinfo`: işlemci hakkında detaylı bilgi

`/proc/meminfo`: sistemin hafıza kullanımu hakkında detaylı bilgi

`/proc/modules`: o anda kullanımda olan çekirdek modülleri

`/proc/mounts`: o anda sisteme bağlı blok aygıtlar ve dosya sistemleri

`/proc/pci`: sistemin PCI kaynakları hakkında detaylı bilgi

Sistem yöneticisi veya normal bir kullanıcı sistemin hafıza kullanımı ile ilgili detayli bilgilere ulaşmak için `free` programını kullanabilir.

Sistemde çalışan süreçlerin sistem kaynaklarının kullanımı hakkında detaylı bilgi almak için `top` programı kullanılabilir. `top` programı varsayılan ayarlarına göre süreçlerin dizilimi en fazla işlemci gücü kullanan en yukarıda olmak üzere yukarıdan aşağıya doğru yapar.

`uptime` komutu ile sistemin açıldığından bu yana geçen zaman görülebileceği gibi, yük ortalaması (load average) adı altında sistemdeki işlemci gücünün kullanım oranı 1, 5, 15 dakikalık istatistikler ile verilmektedir. Yük ortalamasının 1.00 olması işlemci gücünün tam kapasite ile kullanıldığını belirtir, daha yüksek olması ise sistemin aşırı yüklenmeye başladığını gösterir. Yük ortalaması bilgisine `top` programı vasıtası ile de erişilebilir.

Sistem kaynaklarının görüntülenmesi ile ilgili bir diğer program `vmstat`'tır. Sanal hafıza kullanımı, giriş/çıkış işlemleri ve işlemci aktivitesi ile ilgili bilgiler verir. `vmstat`'ın onemli bir çıktısı son kolonunda yer alan işlemcinin boş geçirdiği zaman yüzdesidir (idle time). `vmstat` programı belirli aralıklarla çalıştırılırsa çok daha sağlıklı bilgi alınabilir. Özellikle CPU Idle Time için `vmstat` programının ardarda çalışması gerekmektedir. Bu ardarda çalışma periyodunu sağlamak için `vmstat` komutuna verilecek sayısal bir parametre, vmstat'ın kaç saniye aralıkla çalıştırılacağını belirtir:

```
$ vmstat 1
```

Yukarıdaki komut satırı ile vmstat'ın 1 saniye aralıkla sistem üzerinde ölçüm yapması sağlanmış olur.

15. Sistem Kayıt Sunucusu ve Kayıtların İncelenmesi

UNIX sistemlerinde çalışan sunucular durum raporu, uyarı mesajı ve hata mesajı çıktılarını, sistem kayıt sunucusuna gönderirler. Sistem kayıt sunucusu, kendisine gelen "sistem günlüğüne ekleme" başvurularını alarak ilgili sistem günlüğü dosyasına eklerler.

Sistem yöneticisi, hatalı çalışan bir sunucuyu tespit etmek için ya da hatalı çalıştığı bilinen bir sunucunun tam olarak nerede ve nasıl bir hata yaptığını görmek için sistem günlüğü dosyalarından faydalanır.

Çalışan sunucuların diledikleri bilgileri sistem günlüğüne yazabilmeleri için sistem kayıt sunucusu (`syslogd`) çalışır durumda olmalıdır. Sistem kayıt sunucusunun çalışmıyor durumda olması halinde aşağıdaki komut ile çalışmasının sağlanması mümkündür:

```
# /etc/rc.d/init.d/syslog start
```

Sistem kayıt sunucusu belirli hizmetlerin belirli öncelik gruplarına göre ayrıştırılması yolu ile sistem günlüğünü tutar. Örneğin kayıt sunucusu, posta sunucu yazılımına ilişkin hata mesajlarını filtreleyerek bunları ayrıca işleyebilir.

Sistem günlüğüne eklenen her kayıt hizmet türü ve hizmet önceliği bilgileri ile işaretlenir. Hizmet türlerinin bir listesi aşağıda verilmiştir:

auth: Kullanıcı tanımlaması (authentication)

authrpiv: Kullanıcı yetkileri yükseltmesi

cron: Cron ve at sunucusu ile ilintili

daemon: Hizmet sunucu yazılımlar ile ilintili

kern: Çekirdek ile ilintili

lpr: Yazıcı hizmetleri ile ilintili

mail: Posta sistemi ile ilintili

news: Haber öbekleri hizmetleri (NNTP) ile ilgili

local0 – local7: Yerel kullanım için ayrılmış hizmet türü belirteçleri

Olası hizmet önceliği belirteçlerinin bir listesi artan öncelik sırası ile aşağıda verilmiştir:

debug: Hata ayıklama bilgisi

info: Muhtelif bilgi

notice: Not edilmeye değer bilgi

warning: Uyarı niteliğindeki bilgi

error: Hatalar

crit: Kritik hatalar

alert: Alarmlar

emerg: Acil durumlar

`syslogd`'nin ayar dosyası `/etc/syslogd.conf`'tur. Bu dosyadaki her satır aşağıdaki biçimdedir:

```
hizmettürü.öncelik  kayıtdosyası
```

Aşağıda ayar dosyasından örnek satırlar ve açıklamaları yer almaktadır:

```
*.emerg       *
```

Her hizmet türü için acil durum önceliğine sahip günlük kayıtlarının sisteme giriş yapmış tüm kullanıcıların ekranlarında görüntülenmesi sağlanır.

```
authpriv.*    /var/log/secure
```

Yetki yükseltimine ilişkin her öncelikteki günlük kaydının `/var/log/secure` dosyasına yönlendirilmesi sağlanır.

```
kern.!=info         /dev/console
```

Çekirdek türü hizmetlerden gelen ve öncelik bilgisi info olmayan tüm mesajların `/dev/console` aygıtına yönlendirilmesi sağlanır.

```
Daemon.notice       /var/log/messages
```

Hizmet sunucu yazılımlar ile ilintili not edilmeye değer bilgiler ve daha öncelikli tüm bilgiler `/var/log/messages` dosyasına yönlendirilir.

`/etc/syslog.conf` dosyası üzerinde yapılan her değişiklikten sonra yazılıma `HUP` sinyalinin gönderilmesi gerekmektedir. Bu işlemi kolayca gerçekleştirmek üzere aşağıdaki gibi bir komut satırının işletilmesi uygun olacaktır:

```
# /etc/rc.d/init.d/syslog restart
```

16. İleri Tarihe Görev Atama

İleri tarihe görev ataması, kullanıcının bilgisayar başında olamadığı zamanlarda ya da periyodik olarak belli zamanlarda programların çalıştırılabilmesi için kullanılır. Geceleri İnternet bağlantısının boş olduğu saatlerde gerçekleştirilecek bir dosya transfer işlemi için ileri tarihe görev ataması istenebilir. Benzer biçimde her gece sistem üzerinde rutin kontrolleri gerçekleştirecek bir betik de sistem yöneticisi tarafından ileri bir tarihten itibaren periyodik olarak işletilmek üzere atanabilir.

İleri tarihte yalnızca bir seferliğine işletilecek komutlar için `at` programı kullanılır. Örneğin bir dakika sonra bir komut işletmek için

```
$ at now + 1 minute "cp /etc/hosts >/tmp"
```

biçiminde bir giriş yapılması yeterli olacaktır. Örnek komut, `/etc/hosts` dosyasının bir kopyasının `/tmp` dizini altında oluşturulmasını sağlamaktadır.

```
$ at 10am Jan 1 "/usr/local/bin/kartgonder burak@linux.org.tr"
```

Yukarıdaki gibi bir komut ile önümüzdeki ilk yılın ilk gününde saat 10:00'da `/usr/local/bin/kartgonder` programının belirtilen parametreler ile işletilmesi sağlanabilir. Örnekte, programın kendisine verilen adrese standart bir doğum günü kutlama mesajı gönderdiği varsayılmıştır.

`atq` programı çalıştırılmak üzere kuyrukta bekleyen işlerin bir listesini görüntüler; ilk kolonda göreve atanan sıra numarası görülmektedir:

```
$ atq
4       2001-01-27 20:17 a
```

İleri tarihte işletilmek üzere atanmış bir görevden vazgeçmek ve bu görevi iptal etmek için `atrm` programı kullanılır:

```
$ atrm 4
```

`at` ile ileri tarihe atanan görevlerin çalıştırılabilmesi için `atd` (at sunucusu) çalışır durumda olmalıdır. Çalışmaz durumda ise aşağıdaki gibi bir komut ile başlatılabilir:

```
/etc/rc.d/init.d/atd start
```

Düzenli işletilecek görevler için `cron` mekanizmasından faydalanılır. Cron görevleri, periyodik olarak işletilecek görevlerdir ve `crond` (cron sunucusu) tarafından işletilirler. Cron sunucusu çalışmaz durumda ise aşağıdaki gibi bir komut ile başlatılabilir:

```
/etc/rc.d/init.d/crond start
```

Düzenli olarak işletilecek görevlerin bir listesini almak üzere `crontab` programı `-l` parametresi ile işletilmelidir. Komutun kullanımına ilişkin örnek çıktı aşağıdaki gibidir:

```
# crontab -l
0 0 * * * /usr/local/bin/disktemizle
0,12 * * * 1 /usr/local/bin/denetle
```

Cron ile atanan görevlerin belirlenmesinde ve sorgulanmasında kullanılan format altı kolondan oluşan bir liste biçimindedir. Listenin son kolonu, önceki beş kolon aracılığı ile belirlenen tarih/saatte çalıştırılacak komut satırını göstermektedir. Zaman belirteci olarak kullanılan ilk beş kolonun açıklaması aşağıda verilmiştir:

Dakika: 0-59 arasında dakika belirleyici

Saat: 0-23 arasında saat belirleyici

Ayın günü: 1-31 arasında gün belirleyici

Ay: 1-12 arasında ay belirleyici

Haftanın günü: 0-6 arasında haftanın günü belirleyici (0=Pazar, 1=Pazartesi ...)

Zaman belirleyici olarak kullanılan alanlar boşluk sembolü ile ayrılmalıdır. Yıldız (`*`) işareti "olası tüm değerler" anlamında kullanılmaktadır. Birden fazla değer virgül (`,`) işareti ile ayrılabilir. Bir değer aralığı eksi (`-`) işareti ile belirtilebilir. Yukarıda verilen iki örnek satır açıklamaları ile birlikte aşağıda görülmektedir:

```
0 0 * * * /usr/local/bin/disktemizle
```

Bu satırda, `disktemizle` programının her gece yarısı (00:00) çalıştırılması sağlanmaktadır.

```
0,12 * * * 1 /usr/local/bin/denetle
```

Bu satırda `denetle` programının her hafta haftanın ilk günü (Pazartesi) gece yarısı (00:00) ve gün ortasında (12:00) çalıştırılması sağlanmaktadır.

Düzenli işletilecek görevler listesinde güncelleme yapmak üzere `crontab` programı `-e` parametresi ile çalıştırılmalıdır:

```
# crontab -e
```

Düzenli işletilecek görevleri düzenlemek için özellikle bir düzenleyici yazılımı tanımlamadıysanız `vi` programı başlatılacaktır. Bu program ile rahat çalışamıyor olmanız durumunda aşağıdaki gibi iki satırın işletilmesini sağlayacak `pico` programının kullanılmasını sağlayabilirsiniz:

```
# EDITOR=/usr/bin/pico
# export EDITOR
```

17. Ağ Ayarları

Modern bir işletim sisteminin sağlamak ile yükümlü olduğu hizmetlerin ilki ağ bağlantısının sağlanmasıdır. UNIX, tarihsel gelişimi çerçevesinde ilk TCP/IP gerçekleştiriminin hazırlandığı işletim sistemidir ve bu özelliği nedeni ile İnternet'in gelişiminde önemli rol oynamıştır.

Linux altında ağ ayarlarının yapılandırılması için bir çok farklı seçenek mevcuttur. Ağ ayarlarının yapılandırılmasında kullanılan dosyaları düzenleyen, kullanımı son derece kolay programlar tercih edilebileceği gibi, ayarları ilgili yapılandırma dosyalarının elle değiştirilmesi yolu ile gerçekleştirmek de mümkündür.

Ağ ayarlarını hızla ve kolayca gerçekleştirmek üzere `netconfig` programı kullanılabilir. `netconfig` programı her Linux dağıtımında bulunmayabilir, ancak popüler dağıtımlardan Red Hat ve türevlerinin tümü bu programı barındırmaktadır.

`netconfig` programı `/etc/sysconfig/network` dosyasını ve `/etc/sysconfig/network-scripts` dizini altında yer alan dosyaları kullanıcıdan alınan parametreler aracılığı ile yeniden düzenleyen bir aracı programdır.

Program başlatıldığında birincil ağ bağdaştırıcısına verilmek istenen IP adresini, bu IP adresi için ilgili ağ maskesini ve ağ geçidini soracaktır. Verilen yanıtlar doğrultusunda `/etc/sysconfig/network` ve `/etc/sysconfig/network-script/ifcfg-eth0` dosyaları yeniden düzenlenecektir. IP adresinin bir DHCP havuzundan alınıyor olması durumunda yapılandırma dosyaları buna uygun biçimde düzenlenecektir.

Ağ ayarları ile birlikte DNS sunucu bilgileri de netconfig tarafından `/etc/resolv.conf` dosyası içerisine eklenecektir.

Ağın çalışır durumda olması için gereksinim duyulan ağ ayarları `ifconfig` programı ile yapılır. `ifconfig`, ağ bağdaştırıcınızın kullanacağı bağlantı kapsülleme modelinden, donanım adresine (MAC), kullandığı protokol adresinden protokol yayın (broadcast) adresi, ağ maskesi ve yayın durumuna değin bir çok değişkeni düzenlemenize imkan veren bir programdır.

Sistemin ağ ayarları yapılandırılmış ise, sistem açıldığında ilgili ağ bağdaştırıcısının ayarları otomatik olarak yapılır ve aksi özellikle istenmedikçe ağ bağdaştırıcısı aktif hale getirilir. Çalışan sistem üzerinde aktif olan ve olmayan bir bağdaştırıcının ayarlarını yapmak üzere `ifconfig` programı kullanılır. `eth0` isimli ethernet ağ bağdaştırıcısına bir IP adresi atamak üzere aşağıdaki gibi bir komut satırı işletilmelidir:

```
# ifconfig eth0 192.168.1.1 up
```

Yukarıdaki komut işletildiğinde `eth0` adı ile kullanılan bağdaştırıcıya `192.168.1.1` IP adresi verilecek ve eğer aktif değil ise `up` parametresi ile bağdaştırıcının aktif hale getirilmesi sağlanacaktır. ifconfig programı, verilmeyen bazı seçenekler için kendi varsayılan değerlerini atayacaktır. Yukarıdaki örnekte IP adresinin `192.168.1.1` olarak seçilmiş olması nedeni ile ağ maskesi `255.255.255.0` olarak kabul edilmiştir. Eğer sistem alt ağlara bölünmüş bir ağ üzerinde yer alıyor ise `255.255.255.0` yerine farklı bir ağ maskesi kullanımı gerekli olabilir; bu durumda IP adresi ile birlikte ağ maskesinin de bildirilmesi gerekecektir. Bu durumda aşağıdaki komut satırı uygun olacaktır:

```
# ifconfig eth0 192.168.1.1 netmask 255.255.255.128 up
```

`ifconfig` programı yalnızca ağ bağdaştırıcısı ayarlarını gerçekleştirmek için değil, mevcut yapılandırmayı incelemek için de kullanılabilir.

```
ifconfig eth0
```

biçimindeki bir komut satırı ile `eth0` isimli bağdaştırıcının ayarlarının görüntülenmesi sağlanacaktır.

```
ifconfig -a
```

komutu ise sistemde aktif tüm bağdaştırıcıların ayarlarını görüntülemek amacıyla kullanılabilir.

`ifconfig` programı bir çok işleve sahip olduğu için çok sayıda parametreye sahiptir. Program hakkında daha detaylı bilgi edinmek üzere programın kılavuz sayfasını incelemek uygun olacaktır. Kılavuz sayfaları yalnızca `ifconfig` için değil, sistem üzerinde var olan hemen her ayar dosyası ve program için de temel başvuru kaynağı niteliği taşımaktadır.

Yerel ağdaki veya İnternet'teki sistemlere isimleri ile ulaşılmaya çalışıldığında sistem bu isimleri IP adreslerine dönüştürecektir. Sistem isimlerini IP adreslerine dönüştürmek için sistemin başvuracağı kaynaklar `/etc/hosts` dosyası, NIS ve ağın DNS sunucusudur.

`/etc/hosts` dosyası, sistem isimlerini IP adresleri ile eşleştirme işini durağan bir biçimde gerçekleştirmek için kullanılabilir. Örnek bir dosya aşağıda verilmiştir:

```
127.0.0.1               localhost
192.168.1.1             laetitia laetitia.homenet
192.168.1.2             trust    trust.homenet
192.168.1.3             pengo    pengo.bddx.net
212.50.50.30            ankara   ankara.linux.org.tr
```

`/etc/hosts` dosyasının yazım kurallarına göre birinci kolon IP adresi ve aynı satırda yer alan boşluk ile ayrılmış isimler ise bu IP adresi ile eşlenen sistem isimleridir. Buna göre örneğin `telnet trust` gibi bir komut verildiğinde sistem /etc/hosts dosyası kayıtlarından bu sistemin IP adresinin `192.168.1.2` olduğunu öğrenecek ve bağlantıyı bu IP adresine gerçekleştirecektir. Bu anlamda `telnet trust` ve `telnet trust.homenet` komutları arasında sonuç açısından herhangi bir fark yoktur.

`/etc/hosts` dosyasının kullanımı ile yalnızca durağan IP adresi ve sistem adı eşlemesi gerçekleştirilebilir. Örneğin IP adresinin `212.50.50.30` olarak bilindiği `ankara.linux.org.tr` isimli sistemin sistem yöneticisinin ani bir kararla sistemin IP adresini değiştirdiğini düşünelim. Böyle bir değişiklik sonrasında elimizdeki IP adresi bilgisi yanlış olacak ve bağlanmak mümkün olamayacaktır. Bu nedenle küçük yerel ağlar için /etc/hosts dosyası ile isim ve IP adresi eşlemeleri uygun olurken büyük ağlar ve İnternet için DNS kullanımı zorunluluk arzetmektedir.

Sistemin isimleri IP adreslerine ve IP adreslerini isimlere eşleme konusundaki çalışması "isim çözme" olarak adlandırılır. İsim çözme süreci sırasında kullanılacak DNS sunuculara ilişkin IP adresleri `/etc/resolv.conf` dosyasında yer alır. Bu dosyadan DNS sunuculara ilişkin satırlar aşağıdaki biçimdedir:

```
nameserver 192.168.1.5
```

Örnek satırda sistemin DNS sorgularının `192.168.1.5` adresli DNS sunucusuna yönlendirilmesi sağlanmaktadır. Bu dosyaya birden fazla `nameserver` satırı ile birden çok DNS sunucu tanımının girilmesi mümkündür; sistem her zaman en üstteki satıra yüksek öncelik atayacak ve ancak bu sunucuya erişilememesi durumunda bir alttaki satırda belirlenen sunucuya bağlanılmaya çalışılacaktır. Aynı alanda (domain) bulunan sistemler birbirlerine sadece isimlerini (alan adı hariç) vererek ulaşabilirler. Bunu sağlamak üzere kullanılan sisteme, hangi alan adı içerisinde yer aldığının bildirilmesi gerekmektedir. Aşağıdaki satır bu konuda bir örnek teşkil eder:

```
domain linux.org.tr
```

Yukarıdaki satır `/etc/hosts` dosyasına eklendikten sonra adresinin son kısmı `.linux.org.tr` olan sistemlere yalnızca sistem adının ilk kısmının verilmesi yolu ile erişilmesi mümkün olacaktır.

18. Temel Ağ Hizmetleri ve inetd

Sunucu kaynaklarının çok kıt olduğu günlerde geliştirilen bir teknoloji olan `inetd` sunucusu, en az kaynak kullanımı ile olabildiğince çok ağ hizmetinin sağlanmasına imkan verecek bir düzenek teşkil eder. Sıkça İnternet Süper Sunucusu olarak da anılan `inetd`, birden çok ağ hizmeti için kukla vazifesi görür. `inetd`, çalışmaya başladığında birden fazla ağ hizmetini kendi başına veriyormuşçasına ağ bağlantı noktalarını dinlemeye başlar. Dinlediği bağlantı noktalarından herhangi birine bir hizmet talebi geldiğinde bu isteği alır ve ilgili hizmet sağlayıcı programı çalıştırarak isteğin karşılanmasını sağlar. Hizmetin karşılanmasından sonra hizmeti sağlayan program sonlanır ve `inetd` çalışmasını sürdürür. Bu çalışma biçimi sayesinde, herhangi bir anda tüm hizmet sağlayıcı yazılımların çalışmasının önüne geçilmiş olur; hizmet sağlayıcı yazılımlar yalnızca gerektiğinde çalıştırılacaktır ve bu çok ciddi miktarda bellek tasarrufu sağlayacaktır.

`inetd` 'nin sunduğu modelin alternatifi, tüm hizmet sağlayıcı yazılımların ayrı birer sunucu olarak çalıştırılmasıdır. Bu modelde, arada `inetd` ya da benzeri bir aracı olmayacağından hizmet daha yüksek performans ile sunulabilecek, ancak bellek tüketimi ciddi biçimde artacaktır. Hangi hizmetlerin `inetd` aracılığı ile, hangilerinin doğrudan ayrık sunucu yazılımları ile sağlanacağı konusundaki kararın belirleyicisi yazılımın kullanım sıklığı olmalıdır. Yoğun biçimde kullanılan hizmetler için ayrık sunucu yazılımları tercih edilmelidir.

`inetd` sunucusu sistemde çalışmaz durumda olması halinde aşağıdaki komut ile başlatılabilir:

```
# /etc/rc.d/init.d/inet start
```

Sunucu, çalıştırılması ile birlikte `/etc/inetd.conf` yolundaki ayar dosyasını okur ve içinde tanımlanan hizmetleri sunmaya başlar. `/etc/inetd.conf` dosyasının her bir satırı sırası ile aşağıdaki alanlardan oluşur:

```
hizmet_adı soket_türü protokol seçimler kullanıcı sunucu_yazılımı parametreler
```

Aşağıda, bu dosyadan üç satır örnek olarak verilmiştir:

```
ftp    stream  tcp   nowait  root        /usr/sbin/in.ftpd     in.ftpd
telnet stream  tcp   nowait  root        /usr/sbin/in.telnetd  in.telnetd
```

Bu dosyadaki ilk kolon hizmet adını belirler. Hizmet adı, `/etc/services` dosyasında bir hizmet bağlantı noktası ile eşleştirilmiş olmalıdır. Dördüncü kolonda sunucu yazılımının hangi kullanıcının yetkileri ile çalıştırılacağına ilişkin bilgiler yer almaktadır. Beşinci kolonda çalıştırılacak programın tam yolu, son kolonda ise programın parametreler listesi yer alır.

`/etc/services` dosyasında, hizmet isimlerinin ağ bağlantı noktası tanımları ile eşlemesi yer alır. Bu dosya içeriğinden örnekler aşağıda yer almaktadır:

```
ftp         21/tcp
telnet      23/tcp
talk        517/udp
```

`/etc/inetd.conf` dosyasında verilen her satır, yeni bir hizmeti tanımlamaktadır. Ağa açılan her farklı bağlantı noktası yeni güvenlik risklerini de beraberinde getirmektedir. Bu nedenle, kullanılmayan tüm ağ hizmetlerinin kapatılması önerilir; `/etc/inetd.conf` dosyası bu türden hizmetlerin kapatılması için başlangıç noktası teşkil eder.

`inetd`'nin ayar dosyasının değişikliğinden haberdar olmasını sağlamak üzere `HUP` sinyalinin her değişiklik sonrasınra `inetd` sürecine gönderilmesi gerekmektedir. Bu amaçla önce `ps` programı vasıtası ile inetd sürecinin süreç numarasının tespit edilmeli ve sonrasında `kill` programı ile bu sürece `HUP` sinyali gönderilmelidir. Bu işlemi basitleştirmek üzere aşağıdaki komut satırı işletilebilir:

```
# /etc/rc.d/init.d/inet restart
```

19. Linux Çekirdeği

İşletim sisteminin çekirdeği kullanıcı yazılımları ile donanım arasında bir aracı olarak çalışır. UNIX çekirdeğinin temel görevleri arasında sistem kaynaklarını kullanıcılar ve kullanıcı süreçleri arasında etkin biçimde paylaştırmak yer alır.

Linux çekirdeği, monolitik bir yapıya sahiptir; tüm çekirdek tek ve büyük bir program biçimindedir. Böyle bir yapıda çekirdeği oluşturan tüm bileşenler birbirleri için tanımlanan alanlara erişebilmekte ve işlem yapabilmektedir; bu nedenle monolitik yapıların karmaşık olduğunu öne sürmek mümkündür.

Alternatif çekirdek mimarisi mikro-çekirdek mimarisidir. Bu türden bir çekirdek mimarisinde, çekirdeğin işlevsel bileşenleri kendi başlarına bir bütün teşkil eden parçalardan oluşmuştur. Çekirdeğin en kritik kesimi bu bileşenler arasında mesajlaşmayı sağlayan kesimidir. Teorik olarak kusursuz ve daha esnek olarak görünen bu mimari çoğu durumda yeni bileşenleri sisteme eklemeyi zorlaştırmaktadır.

Linux, monolitik yapısını daha esnek bir biçime getirmek üzere dinamik olarak yüklenebilir çekirdek modüllerini kullanmaktadır. Modüller gerek duyulduğunda çekirdeğe eklenebilir, kullanılmadıkları zaman ise çıkartılabilir. Bu biçimde çekirdeğin dinamik olarak yapılandırılması sağlanmaktadır. Linux çekirdeğinde modül olarak derlenen parçalar genellikle aygıt sürücüleri (ses kartı, cd-rom, USB aygıtları, vb...) ve temsili aygıtlardır (ağ aygıtları, dosya sistemleri vb).

Normal şartlar altında, Linux dağıtımı ile birlikte gelen çekirdek pek çok aygıt için destek sağlıyor ve kararlı bir biçimde çalışıyor olacaktır. Desteklenmeyen aygıtlar için destek eklemek ya da performansı arttırmak üzere gelişmiş ayarları gerçekleştirmek üzere mevcut çekirdeği yeniden derlemek uygun olacaktır.

Çekirdeğin yeni sürümünün derlenmesi ve kurulması yolu ile de daha yüksek performanslı, daha kararlı ve daha fazla aygıtı destekleyen bir düzeneğin kurulması sağlanabilir. Ancak çekirdeğin yeniden derlenmesi ya da yenilenmesi yalnızca deneyimli sistem yöneticileri için tavsiye edilen bir faaliyettir; çekirdek yenileme sürecinde yapılacak bir hata sistemi çalışmaz hale getirebilir. Deneyim sıkıntısı içerisindeki sistem yöneticilerine tavsiye edilmez.

Sistem yöneticisi ne denli tecrübe sahibi olsa da, çekirdek ayarlarında gözden kaçan bir seçenek yüzünden (ya da verilmeyen bir bileşen desteği nedeniyle) sistem yeni derlenen çekirdeği hiç bir zaman açamayabilir. Bir diğer olasılık, sistemin açılması ancak beklenmedik bir anda işlemeyi durdurması olabilir Bu yüzden görev-kritik makinelerde ve üretimde kullanılan makinelerde, sistem sağlayıcısı tarafından derlenmiş ve paketlenmiş bir çekirdeğin kurulması ya da daha önceden benzer bir makinede derlenip sorunsuz çalıştığından emin olunan bir çekirdeğin esas sunucuya taşınması önerilir.

Bir çekirdek sürümünün desteklediği donanımların tam bir listesine http://www.linuxdoc.org/ adresindeki HARDWARE-HOWTO belgesinden ulaşılabilir. Sistemde çalışan mevcut çekirdeğin sürümüne ilişkin bilgileri edinmek üzere `uname` programı `-r` parametresi ile kullanılabilir:

```
$ uname –r
2.2.16-22
```

Çekirdek sürüm numarası üç bölümden oluşur, her bölüm diğerlerinden nokta (`.`) işareti ile ayrılmıştır; `A.B.C` biçimindedir.

A, çekirdeğin ana sürüm numarasıdır. Çok seyrek değişir; çekirdekte çok radikal değişiklikler yapılırsa ana sürüm kısmı bir arttırılır.

B, çekirdeğin alt sürüm numarasıdır. Alt sürüm numarası, ana sürüm numarasından daha sık bir biçimde değişir. Alt sürüm numarasının çift bir sayı olması o çekirdek sürümünün kararlı durumda olduğunu belirtir. Her türlü platformda, beklenmeyen sorunlar çıkartmayacak şekilde çalışacak çekirdek sürümüdür. Eger bu sayı tek sayı ise, o çekirdek sürümünün geliştirme amaçlı olan kararsız (unstable) çekirdek olduğuna işaret eder. Görev-kritik makinelerde ve üretim makinelerinde bu tip çekirdeklerin kullanılmasından kesinlikle kaçınılmalıdır. Bu çekirdekler, geliştiricilerin ve son gelişmeleri takip etmek isteyenleri makinelerine kurmaları, denemeleri ve doğal olarak hataları bulup, rapor etmeleri içindir. Geliştirme bu çekirdekler üzerinde devam eder. Kararlı duruma geçen özellikler, bir sonraki kararlı çekirdeğe dahil edilir. Ana sürüm numarasının değişmesi durumda alt sürüm numarası yeniden sıfırdan başlatılır.

C, çekirdek revizyon numarasına işaret eder. Çekirdekteki her küçük değişiklikte revizyon numarası bir arttırılır. Alt sürüm numarasının ya da ana sürüm numarasının değişmesi durumunda revizyon numarası yeniden sıfırdan başlatılır.

Linux çekirdeğini kaynak kodu sıkıştırılmış olarak dağıtılmaktadır ve şu anda en güncel kararlı sürüm olan 2.2.18 çekirdeğinin sıkıştırılmıs boyutu 15 MB'dir. Bu dosya açıldığında, sabit diskte yaklaşık olarak 70 MB yer kaplamaktadır.

Çekirdeğin özelliklerini ayarlamakta geçirdiğiniz zaman göz ardı edilerek, "AMD K6 / 466" işlemcili bir makinede tipik bir çekirdek ayarı 2.2.X serisi bir çekirdeği derleme süresi 3 dakika almaktadır. 1995 yılına ait AMIGA 1200 (50 mhz işlemci) sistemde 2.0.X serisi bir çekirdeği derlemek 8,5 saat alırken, Intel 386 işlemcili bir makinede aynı çekirdeği derlemek 1 günü bulabilir.

Çekirdeği kaynak kodundan kendiniz derleyerek nelerin nasıl derleyeceğine (modüler/gömülü) kendiniz karar vermek istiyorsanız ve derleme işleminin, kullandığınız işlemciye göre özel olarak optimize edilmesini istiyorsanız çekirdeğin kaynak kodlarına ihtiyacınız olacaktır. Eğer sadece yeni donanım desteklerinden yararlanmak ve eski sürümdeki hataları giderilmiş bir çekirdeğe ulaşmak istiyorsanız, çekirdek ayarları ve derleme işlemleri ile de uğraşmak istemiyorsanız, RPM olarak dağıtılan daha önceden bilgisayarınızın işlemcisine uygun olarak derlenmiş bir çekirdeği sisteminize hiç zahmetsiz olarak kurabilir, çekirdek derleme sırasında yapabileceğiniz bazı hatalardan dolayı sisteminizin hiç açılmaması veya kilitlenmesi gibi hataları da ortadan kaldırmış olursunuz.

RPM paket sistemi kullanan bir Linux dağıtımına sahipseniz RPM tabanlı bir çekirdeği sisteminize kurmanız için hiç bir engel yoktur. RPM hakkında daha fazla bilgi ve hangi Linux dağıtımlarının RPM paket sistemi ile geldiğini görmek için `http://www.rpm.org` sitesini ziyaret edebilir, RPM paket sistemi kurulu olan bir Linux dağıtımı kullanıyorsanız, `/usr/share/doc` veya `/usr/doc` dizini altında bulunan RPM yardımıcı dokümanlarına ve RPM man sayfalarına göz atabilirsiniz.

RPM olarak paketlenmiş bir çekirdeğe ulaşabileceğiniz en iyi kaynaklar Internet ya da sistem sağlayıcınızın (Linux dağıtımınızı üreten grup/şirket) sağladığı CD'dir. İnternet'ten çekirdek RPM paketlerine ulaşabileceğiniz en güvenilir yer sistem sağlayıcınızın FTP sitesi olacaktır. Örnek olarak Red Hat Linux dağıtımı ele alınırsa, `ftp://ftp.redhat.com/pub/updates` dizini içinden, kurulu olan dağıtımı sürümüne ve kullanılan işlemci tipine göre çekirdek güncellemesi ulaşılabilir. `ftp://ftp.linux.org.tr` adresi altından Linux Kullanıcıları Derneği'nin FTP sunucusuna ulaşabilirsiniz. `ftp://ftp.linux.org.tr/pub/redhat` dizininde `ftp://ftp.redhat.com/` adresinden ulaştığınız Red Hat FTP'nin bir yerel yansısı bulunmaktadır.

Türkiye içinden bağlantılarda ftp://ftp.redhat.com/ adresi yerine ftp://ftp.linux.org.tr/pub/redhat adresinin kullanılması hızlı bir bağlantı ve yurtdışı hatlarının etkin kullanımı açısından tavsiye edilmektedir.

Büyük Linux sistem sağlayıcıları genellikle en son çıkan çekirdekleri, kararlı çekirdekler olmasına rağmen, hemen paketlemezler. Çekirdeğin güvenli olduğuna inanılması için geçen bir süreden sonra bu gruplar/şirketler çekirdek için bir güncelleme paketi çıkartırlar. Tabii bu sistem sağlayıcılarının haricinde başkalarının paketlediği çekirdekler de olacaktır. Bunlara güvenip güvenmemek tamamen güvenlik politikanıza bağlıdır.

RPM haldeki bir çekirdeğin sisteme kuruluşu birkaç adımdan oluşur. Çekirdeğin RPM paketinin `/root` dizininde yer aldığını varsayacağız. Paketi kurabilmek için sistem yöneticisi (`root`) yetkilerine sahip olunması gerekmektedir.

```
# rpm -ivh kernel-2.2.16-3.i386.rpm
kernel ##################################################
```

Komutun işletilmesinden sonra hemen alt satırında kurulan paketin ismini ve kurulum işleminin o anki durumu görüntülenecektir.

Kurulum sonrasında /boot dizininizde sıkıştırılmış Linux çekirdeği ve diğer bazı gerekli dosyalar yer almış olacaktır. Ayrıca `/lib/modules/2.2.16-3` dizini altına da çekirdeğe dinamik olarak yüklenebilen modüller yer alacaktır.

Yeni çekirdeği sisteminize kurduktan sonra Initial RAMDISK'i oluşturmanız gerekecektir. Initial RAMDISK, çekirdeğin yüklenmesinden önce sistemin bazı sürücülere ihtiyacı olması durumunda kullanılır. Örneğin SCSI bir sabit diskiniz varsa varsa, bu diskte yer alan çekirdeğe ulaşılması için o SCSI arabiriminin sürücüsüne ihtiyaç duyulacaktır. RAMDISK sayesinde bir sabit disk sürücü gibi kullanılabilen hafıza alanına

bu SCSI arabiriminin sürücüsü yüklenir ve çekirdek ile ilgili SCSI diske ulaşabilmek için buradakı sürücüyü kullanır. Çekirdek kendini yükledikten sonra bu kullanılmayan alanı boşaltacak ve sistemin kullanımına sunacaktır.

Initial RAMDISK oluşturmak için :

```
# mkinitrd /boot/initrd-2.2.16-3.img 2.2.16-3
```

LILO'yu yeni derlediğiz çekirdeği başlatacak şekilde ayarlamanız gerekmektedir. LILO'nun ayar dosyası genellikle `/etc/lilo.conf` dosyasıdır. Kullandığınız metin editörü yardımı ile `/etc/lilo.conf` doysası içine yeni çekirdeğiniz ile ilgili satırları eklemeniz gerekiyor. Mutlaka eklenmesi gereken satırlar aşağıdaki gibidir:

```
image=/boot/vmlinuz-<buraya çekirdek sürümünü yazınız>
        label=linux-yeni
        root=<buraya kök bölüm (partition) gelecek>
        initrd=/boot/initrd-<buraya çekirdek sürümünü yazınız>.img
        read-only
```

Örneğimize göre bu satirlar aşağıdaki gibi olmalıdır:

```
image=/boot/vmlinuz-2.2.16-3
        label=linux.yeni
        root=/dev/hda1
        initrd=/boot/initrd-2.2.16-3.img
        read-only
```

RPM olarak kuracağınız çekirdekler, `/boot/vmlinuz` adlı sembolik bağlantıyı kurdukları yeni çekirdeği işaret edecek biçimde güncelleyeceklerdir. Eğer `/etc/lilo.conf` içinde yer alan eski çekirdeğiniz ile ilgili satırlardaki `image=` kısmı, `/boot/vmlinuz` 'u gösteriyorsa, bunu da eski çekirdeğinizin asıl dosyasının bulunduğu yer ile değiştirin. Çünkü artık `/boot/vmlinuz` eski çekirdeğinizi değil yeni kurulanı gösteriyor durumdadır.

Şimdi yeni LILO ayarlarını etkin kılmak için komut satırında "lilo" komutunu verin:

```
# lilo

Added linux *
Added linux.yeni
```

Gördüğünüz gibi, LILO yeni ayarları etkin hale getirdi. Yeni derlediğiniz çekirdeği kullaranak istemi başlatmak için, sistem açılırken "LILO: " yazısını gördükten sonra "linux.yeni" yazıp "ENTER" tuşuna basın. Yeni çekirdeğin sorunsuz çalıştığına emin olduktan sonra, sistemin her zaman bu yeni çekirdek ile açılmasını sağlamak için gene `/etc/lilo.conf` dosyasını düzenlemeniz ve değişlikleri etkin kılmak için `lilo` komutunu çalıştırmanız gerekiyor.

Yeni çekirdeği, varsayılan çekirdek yapmak için, `/etc/lilo.conf` dosyasını kullandığınız metin editörü yardımı ile açın. LILO, hangi etiket adına (label) sahip çekirdeği varsayılan çekirdek olarak yükleyeceğini

```
default=linux
```

satırında öğrenmektedir. Görüldüğü gibi LILO, etiket ismi "linux" olan çekirdeği -aksi belirtilmedikçe- başlatacaktır. Bu satırı

```
default=linux.yeni
```

olarak değiştirmeniz işinizi görecektir; ancak tavsiye edilmez. Bunun yerine bu satıra dokunmadan, varsayılan çekirdek olarak başlatılmasını istediğiniz çekirdeğin etiket ismini `linux` yapar ve eski çekirdeğin etiket ismini `linux`'dan `linux.eski` gibi başka bir etiket ismine çevirirseniz yeni çekirdeğiniz -aksi belirtilmedikçe- sistem açıldığında başlatılan çekirdek olacaktır. Ne zaman eski çekirdeği kullanarak sistemi başlatmak isterseniz, sistem açılırken "LILO: " yazısını gördükten sonra `linux.eski` yazıp "ENTER" tuşuna basarsanız sistem eski çekirdek ile açılacaktır.

Yeni bir çekirdek ile çalışmanın ikinci yolu kaynak kod paketi olarak çekirdeği edinmek ve derlemektir. Çekirdek kaynak koduna ulaşmak için en uygun yer `ftp://ftp.kernel.org` adresidir. Linux çekirdeğinin resmi olarak bulunduğu yer olan Kernel.ORG'un FTP sitesinde ilk Linux çekirdeklerinden son Linux çekirdeğine kadar tüm sürümlere ulaşmanız mümkündür. Örnek olarak

ftp://ftp.kernel.org/pub/linux/kernel/vA.B adresinde A.B serisi çekirdek sürümlerinin tümüne ulaşabilirsiniz. Yani 2.2 serisi çekirdeklere ulaşabileceğiniz adres ftp://ftp.kernel.org/pub/linux/kernel/v2.2 adresidir. Bu adrese girdiğinizde `linux-A.B.C.tar.gz` isimli dosyalar göreceksinizdir. A.B.C çekirdeğin sürümünü belirtir. `linux-2.2.16.tar.gz` isimli dosya, 2.2.16 sürümü çekirdeği içeren tar arşividir.

Bu dizin içinde aynen sonu `.tar.gz` ile biten dosyalar gibi sonu `.tar.bz2` ile biten dosyalar da göreceksiniz. Aynı çekirdek sürümünü içeren bu dosyalara arasındaki tek fark, sıkıştırılırken kullanılmış algoritmadır. Eğer sisteminizde bz2 tipi sıkıştırılmış dosyaları açabilecek yazılım varsa (genellikle tüm Linux dağıtımlarında mevcuttur) sonu `.tar.bz2` olan dosyaları tercih edebilirsiniz. Boyutları daha küçük olduğu için bunları İnternet'ten indirme süreniz azalacaktır.

İnternet'ten indirdiğiniz çekirdeği `/usr/src/` dizininine kopyalayın ya da taşıyın. Eğer daha önceden makinenize bir çekirdek kaynak kodu yüklenmiş ise (bir çok Linux dağıtımı kurulum sırasında bunu yapar) bu dizin içinde `linux` diye bir dizin olacaktır veya gene bu isimde bir sembolik bağlantı bulunacaktır. Açacağınız sıkıştırılmış yeni çekirdek de kendini `linux/` dizinine açmak isteyeceğinden dolayı; olası bir karmaşaya zemin hazırlamamak için bu eski dosyaların bulunduğu dizin artık `linux/` olmamalıdır. Eğer `linux` bir sembolik bağlantı ise, çekinmeden silebilirsiniz; ancak bir dizin ise ve eski çekirdek kaynak kodlarınızı saklamak istiyorsanız ismini linux-A.B.C olarak (burada A.B.C çekirdek sürümünü belirtiyor) değiştirebilirsiniz. Eğer eski çekirdek kodlarını saklamak istemiyorsanız bu dizini silebilirsiniz.

/usr/src/ dizininde `linux` adında bir dizin/dosya/bağlantı olmadığında emin olduktan sonra, sıkıştırılmış çekirdek kaynak kodlarını açabilirsiniz. Eğer .tar.gz ile arşivlenip sıkıştırılmış bir çekirdek çektiyseniz `tar zxvpf linux-A.B.C.tar.gz` `.tar.bz2` ile arşivlenip sıkıştırılmış bir çekirdek çektiyseniz `tar Ixvpf linux-A.B.C.tar.bz2` komutu ile, sıkıştırılmış çekirdeği açabilirsiniz.

Açılan dosyaların isimlerini ekranda göreceksiniz. Dosyaların ve dizinlerin tümü /usr/src/linux dizini altına yerleşecektir. Yukarıda da bahsettiğimiz gibi, `/usr/src/linux` geleneksel olarak `/usr/src/linux-A.B.C` (burada A.B.C çekirdek sürümünü belirtiyor) dizinine bir sembolik bağlantıdır. Bunu yapmak için aşağıdaki gibi bir komutu işletmelisiniz:

```
mv linux linux-A.B.C
ln -s linux-A.B.C linux
```

Yeni çekirdek paketinin ayarlarını yapmak üzere aşağıdaki komutları işletmelisiniz:

```
cd /usr/src/linux
make config
```

Çekirdek seçeneklerini yazın. Size çekirdek opsiyonlarının derlenecek çekirdeğe katılıp katılmamasını ve eğer katılacaksa, modül olarak mı katılacağını soracaktır ve sizden bir cevap bekleycektir. Verebileceğiniz cevaplar :

n (HAYIR anlamındadır. Bu özellik/sürücü/vb... çekirdeğe katılmayacaktır)

y (EVET anlamındadır. Bu özellik/sürücü/vb... çekirdeğe katılacaktır)

m (Eğer bir sürücü ise, bu sürücü modül olarak derlenecektir)

? (Bu özellik/sürücü/vb... hakkında detaylı bilgi verir)

Size sorulan sorularında sonunda '[' ']' işaretleri arasında verebileceğiniz cevaplar gösterilmektedir. Bu cevaplar arasında başta yer alan ve büyük harf ile yazılmış olan, varsayılan cevaptır. Bu cevabı vermek için ilgili harfini girmek yerine sadece "ENTER" tuşuna basabilirsiniz. Örneğin :

```
BSD Process Accounting (CONFIG_BSD_PROCESS_ACCT) [N/y/?]
```

sorusuna sadece "ENTER" tuşuna basarak cevap verirseniz cevabınızın 'n' yani HAYIR olduğu kabul edilecektir. `make config` ile çekirdek ayarlarını yapmak pek de kolay değildir. Daha önce verdiğiniz bir cevabı geri dönüp düzeltme şansınız olamadığı gibi, tamamen metinden oluşan arabirim bazen çok karışık bir hal alabilir. Bu yüzden `make config`'e alternatif başka make seçeneklerini kullanabilirsiniz. Bunlar `make menuconfig` ve `make xconfig` komutlarıdır. `make menuconfig`, gene metin tabanlı ama renkli ve yön tuşları ile ayar menüleri arasında gezebileceğiniz ve bu menüler içinde gene yön tuşları ile ayar opsiyonları arasında

gezebileceğiniz bir arabirim sunmaktadır. Bir çok kişi bu `menuconfig'i` tercih etmek isteyecektir. `make xconfig` komutu ise, X Window System altından çekirdek ayarlarını farenizi (mouse) kullanarak yapmanıza imkan verecektir. Bu komutu verebilmek için o anda X Window kullanıyor olmanız gerekmektedir. Ayrıca sisteminizde Tk kütüphanelerinin de yüklü olması gerekmektedir. Çekirdek ayarlarınızı bu üç alternatif yöntemden birini kullanarak yapın ve kaydederek çıkın.

Çekirdek ayarları yapılırken dikkat gösterilmesi gereken ayarlardır. Yanlış ve eksik seçimleriniz sistemin yeni çekirdek ile hiç açılmamasına ya da açıldıktan sonra istenildiği gibi davranış sergilememesine yol açabilir. Bu yüzden ayar yaparken çok emin olmadığınız tüm seçimler hakkında ayar programından yardım isteyiniz. Dikkatli bir şekilde okuduktan sonra karar veriniz. Eğer ne olduğunu hala anlamadıysanız ve karar veremiyorsanız yardım metninde tavsiye edilen cevabı veriniz.

Çekirdek ayarlarınızı istediğiniz gibi yaptıktan sonra sıra çekirdeği derlemeye geldi. Çekirdeği derlemeye başlamadan önce derleme öncesi bir kontrol ve temizlik yapılması gerekecektir.

```
make dep
make clean
```

Yukarıda verilen komutlar sırası ile; derleme için gereken dosyaların yerinde olup olmadığını kontrol eder ve daha önceden derlenmiş dosyaların silinmesini sağlar. Eğer çekirdek kaynaklarını yeni çektiyseniz ve bu kaynaklar üzerinde ilk defa derleme yapıyorsanız `make clean` komutunu kullanmak zorunda değilsiniz; ancak yeniden çekirdek derliyorsanız, mutlaka bu komutu da vermelisiniz.

Kontrol ve temizliğin ardından derleme işlemine geçilebilir. Derleme yapmak için gene diğer işlerde yaptığımız gibi `make` programına bir parametre geçireceği. Çekirdeği derlemeniz için bir kaç parametre seçeneğiniz var. Bunlardan bazıları

`make zImage` (çekirdeği derler ve gzip ile sıkıştırır)

`make bzImage` (çekirdeği derler ve bzip2 ile sıkıştırır)

`make bzdisk`.(çekirdeği derler, bzip2 ile sıkıştırır ve diskete yazar)

`make bzlilo`.(çekirdeği derler, bzip2 ile sıkıştırır, LILO'yu çalıştırır)

`make bzImage`'i kullandığınızda, derleme işlemi eğer hatasız sona ererse, `/usr/src/linux/arch/i386/boot/bzImage` dosyası yeni çekirdeğiniz olacaktır."i386" isimli dizin, makinenizin üzerindeki işlemcinin mimarisine göre değişebilir (sparc, alpha, ppc vs...) Eğer Intel veya Intel uyumlu bir işlemci kullanıyorsanız burası her zaman "i386" olacaktır. İyi bir çekirdek kurulumu için, çıkan `bzImage` dosyasını ve gene aynı dizin altında bulunan `System.map` dosyasını sırası ile /boot dizini altına `vmlinuz-A.B.C` ve `System.map-A.B.C` şeklinde kopyalayabilir ve `/boot/System.map` sembolik bağlantısını yeni `System.map-A.B.C` dosyanızı gösterecek şekilde ayarlayabilirsiniz.

`make bzdisk`'i kullandığınızda, çekirdek bir diskete yazılacak ve bu disket başlatılabilir bir hale (bootable) getirilecektir. Yeni çekirdekleri güvenlice denemek için en ideal yol budur.

`make bzlilo` komutunu kullanırken dikkat etmeniz gereken noktalar, LILO ayarlarınızda varsayılan çekirdek imajının `/vmlinuz` olması ve isim etiketinin "linux" olması gereğidir. Modern Linux dağıtımları çekirdek imajlarını `/boot` adlı ayrı bir disk bölümünde tuttukları için bu seçenek taşınabilir değildir ve kullanmaktan kaçınılmalıdır.

Eğer çekirdek ayarları sırasında, derlenebilir çekirdek modüllerine destek verdiyseniz ve en az bir seçeneği modül olarak derlediyseniz :

```
make modules
make modules_install
```

komutlarını vermeniz gerekecektir. Bu komutlar sırası ile, seçimi gerçekleştirilen seçenekleri modül olarak derleyecek ve sabit diskte ilgili alana yerleştirecektir.

Eğer çalışır durumdaki çekirdeğinizi yeninden derliyorsanız, eski modüllerinizin yedeğini alarak `/lib/modules/A.B.C` (A.B.C çekirdek sürümünüz) dizinini make modules_install işleminden önce boşaltınız.

Bu işlemler sona erdikten sonra, yeni çekirdeğinizi LILO'ya tanıtmanız ve LILO'yu yeniden çalıştırmanız gerekmektedir. Burada yapılması gereken işlemler, LILO ayarlarının yapılması başlığı ile anlatılan bölümdekilerin aynısıdır.

LILO ayarlarınızı yaptıktan sonra ve "lilo" komutunu verdikten sonra sisteminizi yeniden başlatarak yeni çekirdeğinizi LILO ayarınıza bağlı olarak çalıştırabilirsiniz.