Open Source Enterprise Centre GNU/Linux Sertifikasyon Programı

Linux 102 Ders Notları

Son Güncelleme: 9 Şubat 2007

İçindekiler

1 Paylaşılan Kütüphaneler

1.1 Paylaşılan Kütüphanelere Giriş

Linux sistemlerinde çalıştırılabilir iki çeşit program tipi vardır. Birinci çeşit programlar statik olarak bağlanmış çalıştırılabilir programlar olarak adlandırılır. Statik çalıştırılabilirler (static executables) ihtiyaçları olan tüm fonksiyonları barındırırlar. Bir başka deyişle bunlar bütün haldedirler. Bu yüzden statik çalıştırılabilirler kullanılmak için harici bir kütüphaneye ihtiyaç duymazlar. İkinci çeşit ise dinamik çalıştırılabilirler olarak adlandırılır. Bu ikinci tip programlara az sonra değineceğiz.

Statik Çalıştırılabilirler ile Dinamik Çalıştırılabirlerin Kıyaslanması İlgili çalıştırılabilir programın statik olup olmadığını ldd komutunu kullanarak anlayabiliriz.

```
# ldd /sbin/lilo
not a dynamic executable
```

ldd tarafından bildirilen "dinamik çalıştırılabilir değildir" bilgisi bize lilo programının statik bağlantılı olduğunu anlatmaktadır.

Dinamik çalıştırılabilirler çalışmaları için gerekli fonksionları almak için paylaşılmış kütüphanelere ihtiyaç duyan, bir anlamda eksik programlardır. Bu yüzden aynı programa ilişkin statik çalıştırılabilir versiyonun boyutu dinamik çalıştırılabilir versiyonuna göre daha büyük olacaktır.

1.2 Dinamik olarak bağlanmış bağımlılıklar

ln komutunun ihtiyaç duyduğu paylaşılmış kütüphanelerin listesini almak için ldd komutunu kullanırız.

```
# ldd /bin/ln
libc.so.6 => /lib/libc.so.6 (0x40021000)
/lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
```

Gördüğünüz gibi, ln çalışmak için libc.so.6 ve ld-linux.so.2 kütüphanelerine ihtiyaç duyar. Bir kural olarak, az önce de bahsettiğimiz gibi dinamik olarak bağlanmış programlar statik olarak bağlanmışlara göre çok daha az yer tutarlar. Öte yandan bir Linux sistemindeki hemen hemen bütün çalıştırılabilir programlar dinamik bağlantılıdırlar.

1.3 Dinamik Yükleyici

Öyleyse, eğer dinamik çalışabilirler, çalışmaları için ihtiyaç duydukları herşeye sahip değillerse Linux'un hangi parçası gerektiği zaman bu programlarla birlikte gerekli kütüphaneleri yükleme görevini üstlenir ? Bunun cevabı dinamik yükleyici (dynamik loader) adı verilen bir kütüphane; yani ld-linux.so.2 dir (ln komutunun bağımlı olduğu kütüphaneler arasında gördüğümüz gibi)

Şimdi de dinamik yükleyicinin sistemdeki gerekli paylaşılmış kütüphaneleri nasıl bulduğuna bir bakalım.

`ld.so.conf`

Dinamik yükleyici paylaşılan kütüphaneleri /etc/ld.so.conf ve /etc/ld.so.cache dosyaları sayesinde bulmaktadır. Eğer /etc/ld.so.conf dosyasına cat komutu ile bakacak olursanız muhtemelen aşağıdaki gibi bir liste ile karşılaşacaksınız:

```
$ cat /etc/ld.so.conf
/lib
/usr/lib
/usr/X11R6/lib
/usr/i486-linuxlibc1/lib
```

ld.so.conf dosyası dinamik yükleyicinin paylaşılan kütüphaneler için bakacağı bütün dizinlerin listesini içermektedir (otomatik olarak yüklenen /lib ve /usr/lib'in yanında)

`ld.so.cache`

Ancak dinamik yükleyicinin bu bilgiyi görebilmesi için bu bilginin ld.so.cache dosyası içerisine dönüştürülmesi gerekir. Bu işlem ldconfig komutu çalıştırılarak yapılmaktadır.

```
# ldconfig
```

ldconfig tamamlandıktan sonra /etc/ld.so.conf ile yaptığınız tüm değişiklikleri en son haliyle gösteren /etc/ld.so.cache dosyasına sahipsiniz demektir. Bundan sonra dinamik yükleyici paylaşılan kütüphaneleri ararken artık sizin /etc/ld.so.conf dosyasına eklediğiniz yeni dizinlere de bakacaktır.

1.4 ldconfig Hakkında İpuçları

ldconfig komutunun görebildiği tüm paylaşılan kütüphaneleri görüntülemek için şu şekilde yazmak gerekir:

```
# ldconfig -p | less
```

Paylaşılan kütüphane yollarını konfigüre etmek için kullanışlı bir yöntem daha vardır. Bazen dinamik yükleyiciden paylaşılan kütüphaneleri ararken /etc/ld.so.conf içerisinde yazılı yollara bakmadan önce sizin belirlediğiniz özel bir dizin içerisine bakmasını isteyebilirsiniz. Bu yeni yüklenmiş kütüphaneler, çalışmayan eski bir uygulamayı kullanmak için gerekebilir.

1.5 LD_LIBRARY_PATH

Dinamik yükleyiciye önce belirli bir dizine bakmasını söylemek için LD_LIBRARY_PATH değişkenini bu istediğiniz dizinler ile set etmeniz gerekir. Birden fazla sayıda yolu ; ile ayırabilirsiniz. Örneğin:

```
# export LD_LIBRARY_PATH="/usr/lib/old:/opt/lib"
```

LD_LIBRARY_PATH değişkeni export edildikten sonra , o andaki kabukta başlatılan çalıştırılabilirler eğer mümkünse /usr/lib/old ya da /opt/lib altındaki kütüphaneleri kullanacaklar, olmazsa /etc/ld.so.conf içerisinde belirtilen yollarda gerekli kütüphaneleri arayacaklardır.

Linux paylaşılan kütüphaneleri ile ilgili genel bilgilerimizi burada noktalıyoruz. Bu konuda ayrıntılı bilgiyi man ldconfig ya da man ld.so komutları ile öğrenebiliriz.

2 Uygulamaları Kaynaktan Derlemek, Giriş

Diyelim ki elinizde sisteminize yüklemek istediğiniz bir uygulama var. Bu uygulamanın en yeni sürümünü kurmak istiyorsunuz. Ancak bu sürüm henüz rpm paket formatı haline gelmemiş olsun. Uygulamayı sadece kaynak kod halinde bulabildiğinizi düşünelim. Bu durumda yapmanız gereken şey bu uygulamayı kaynağından derlemek olacaktır. Bu bölümde bu işlemin nasıl yapılacağını göreceğiz.

2.1 Dosyaları İndirmek (downloading)

İlk işiniz derlemek için ihtiyacınız olan kaynak dosyaları bulmak ve indirmek olacaktır. Bu dosyaları muhtemelen tar.gz, tar.Z, tar.bz2 ya da tgz uzantısı halinde arşivlenmiş ve sıkıştırılmış olarak bulacaksınız. Arşivi, kullandığınız tarayıcı ya da ftp uygulaması ile indirmelisiniz. Eğer uygulama ile ilgili bir web sayfası varsa, kurulum bilgileri hakkında döküman bulabilmeniz açısından bu siteyi ziyaret etmekte fayda vardır. Yükleyeceğiniz program o anda sisteminizde olan ya da olmayan başka programların varlığını gerektirebilir. Eğer bu başka programların sisteminizde olmadığından eminseniz öncelikle bunları temin etmeli, paket ya da kaynaklarının kullanarak siteminize yüklemelisiniz. Bundan sonra asıl uygulamanızı kurmak için artık hazırsınız demektir.

2.2 Dosya Arşivleme ve Sıkıştırmaya İlişkin Kısa Açıklama

tar komutu yardımıyla dosyalarınızı arşivleyebilir, oluşturulmuş bir arşiv içerisine yeni dosyalar ekleyebilir ve arşivleri açabilirsiniz. Aşağıdaki örnekte önce /home/knoppix altında "temp" adında bir dizin ve bu dizin icerisine file1, file2 adında iki dosya ile dir1 adinda baska bir dizin ve en son olarak da bu dir1 dizini icerisine de file1 adinda bir dosya açalım. Bundan sonra bu temp dizinini arşiv haline getirmek için şunu yazalım:

```
# tar -cf arsiv.tar ./temp
# ls
arsiv.tar  Desktop  temp  tmp
```

Burada -c parametresi arsiv yaratılması icin kullanılır. -f paramtresi de kendisinden sonra gelecek olan ismi yeni arsivin ismi olarak algılar. Şimdi yine /home/knoppix altında temp2 diye bir dizin oluşturalım ve bu dizinin içerisine arşivi açalım:

```
# mkdir temp2
# ls
arsiv.tar  Desktop  temp  temp2  tmp
# cd temp2
# tar -xf ../arsiv.tar
# ls
temp
# cd temp
# ls
dir1  file1  file2
```

Bu örnekte de gördüğümüz gibi arşivimizi aynı şekliyle -x parametresi ile açmış olduk. Şimdi diyelim ki arşivimize yeni bir dosya eklememiz gerekiyor. Bunun için ilk akla gelen şey arşivi açıp yeni dosyamızı ekleyip sonra yeniden arşivlemekse buna hiç gerek yok. Çünkü -r parametresi yardımıyla arşivi açmadan yeni bir dosya eklememiz mümkün:

```
# touch ekle.txt
# tar -rf arsiv.tar ekle.txt
```

Şimdi de arşivimizde hangi dosyaların bulunduğuna -t parametresi yardımıyla bir göz atalım:

```
# tar -tf arsiv.tar
./temp/
./temp/dir1/
./temp/dir1/file1
./temp/file2
./temp/file1
ekle.txt
```

Dosya sıkıştırmaya yarayan araçlardan en yaygın olanlardan birtanesi de gzip2dir. gzip ile bir dosyayı sıkıştırabilir ve gunzip ile sıkıştırılmış dosyayı açabilirsiniz:

```
# gzip ekle.txt
# ls
arsiv.tar  Desktop  ekle.txt.gz  temp  temp2  temp3  tmp
# gunzip ekle.txt.gz
# ls
arsiv.tar  Desktop  ekle.txt  temp  temp2  temp3  tmp
#
```

Sonuç itibariyle bir dizin içerisinde yer alan dosyaları en iyi şekilde paketlemek için bunları önce arşivlemek sonra da sıkıştırmak gerekir. Az önce oluşturduğumuz /temp dizini bu şekilde uygun bir tar.gz paketi haline getirelim:

```
# tar -cf arsiv.tar ./temp
# ls
arsiv.tar  Desktop  temp  temp2  temp3  tmp
# gzip arsiv.tar
# ls
arsiv.tar.gz  Desktop  temp  temp2  temp3  tmp
#
```

2.3 Paketi Açmak

Kaynak arşivi açmak nispeten daha kolaydır. Eğer arşivinizin uzantısı tar.gz, tar.Z ya da .tgz biçiminde ise arşivi açmak için şu komutu kullanabilirsiniz:

```
$tar -xzvf archivename.tar.gz
```

x: açmak için

v: görünür durumda açmak için (açılan dosyalar ekranda yazılır halde)

f: Bundan sonra açılacak arşiv dosya ismi yazılacağı anlamına geliyor

Bundan sonra bütün kaynak dosyaları tek bir dizin içerisinde yeralacak şekilde arşiv dosyası açılacaktır. Bu sayede, bir arşivi açtığınız zaman, o anda çalıştığınız dizin içerisine bu arşivde yer alan dosyalar dolmayacak, onun yerine daha düzenli bir biçimde ayrı bir dizin içerisine yer alacaklardır.

2.4 Arşivleri Listelemek

Çalışmalarınız sırasında, açıldığı zaman çalıştığınız dizin içerisine yüzlerce dosya dolduracak bir arşivi açmanız gerekecektir. Bir çok arşiv dosyası böyle açılmadığı halde bazen bu durumla karşılaşabilirsiniz. Eğer arşiviniz açıldığında içeriğinde yer alan dosyaların hepsinin tek dizin içerisinde toplanacağından emin olmak istiyorsanız, arşivin içeriğini aşağıdaki komut ile görebilirsiniz:

```
$ tar -tzvf archivename.tar.gz | more
```

t: text listeleme için. Herhangi bir açma işlemi yapılmayacağı anlamına geliyor.

Eğer arşiv listesinin sol tarafında ortak bir dizin belirtilmemişse, arşivinizi yeni açacağınız bir dizin içerisine taşımanız ve açma işlemini burada gerçekleştirmeniz gerekmektedir. Aksi takdirde işlem çok karışık bir hal alacaktır.

2.5 bzip2 ile Sıkıştırılmış Arşivleri Açmak

Arşiv dosyanız .tar.bz2 formatında olabilir. Bu uzantıya sahip dosyalar bzip2 ile sıkıştırılmış dosyalardır. Bzip2 genelde gzip den daha iyi bir sıkıştırma yapmaktadır. bzip2 nin tek dezavantajı sıkıştırma ve açma işlemini daha yavaş gerçekleştirmesi ve gzip'in kullandığından daha fazla bellek alanı kullanmasıdır. Ancak yeni nesil bilgisayarlarda bu ayrıntı artık önemsizdir. Zaten bzip2'nin zaman geçtikçe daha pöpüler olarak kullanılması da bu gerçeği ispatlamaktadır.

bzip2'nin artan ününden dolayı birçok linux dağıtımı artık yamalı halde tar versiyonlarını barındırmaktadır. Bu versiyonların kullanımında y veya i seçeneği ile tar uygulamasına dosyanın bzip2 ile sıkıştırıldığını ve açmak için de yine bzip2 kullanacağını belirtebiliriz. Sahip olduğunuz tar uygulamasının yamalı versiyonu olup olmadığını anlamak için şunu yazın:

```
$ tar -tyvf archive.tar.bz2 | more
```

ya da

```
$tar -tivf archive.tar.bz2 | more
```

Bu komutlardan herhangi birisi çalışmazsa (ve tar yanlış parametre kullandığınızı söylüyorsa)bile hala yapabileceğimiz şeyler vardır. Okumaya devam ediniz :)

bzip2 Boruhatları Sizin tar uygulamanız kısayollar ile bzip2 dosyalarını açamıyorsa ne yapmak gerekir? Neyse ki sisteminiz GNU versiyonu tar uygulaması içermese bile bu işi yapabilmek için nerdeyse tüm Unix sistemlerinde kullanılabilen kolay bir yol mevcuttur. bzip2 dosyasının içeriğini görüntülemek için bir boruhattı yaratabiliriz:

```
# ls
# cat ../arsiv.tar.bz2 | bzip2 -d | tar -t | more
./temp/
./temp/dir1/
./temp/dir1/file1
./temp/file2
./temp/file1
# cat ../arsiv.tar.bz2 | bzip2 -d | tar -x | more
# ls
temp
#
```

2.6 bzip2 Boruhatları (devamı)

Bundan önceki iki örnekte, arşiv dosyamızın içeriğini görüntüleyen ve onu açan standart bir Unix boruhattı yarattık. tar uygulaması, kendisine stdin uzerinden gelen dosyayı açarak disk üzerine değil onun yerine more programına gönderdi.

Eğer bu boruhattı yöntemini kullandığınızda sistem bzip2 diye bir uygulamayı bulamadığını bildiriyorsa muhtemelen bzip2 sisteminizde yüklenmemiştir. Uygulama ile ilgili kaynakları redhat.com[1] ve debian.org[2] adreslerinde bulabilirsiniz. bzip2 nin sisteme kurulmasından sonra (bu dökümanı takip ederek) ilk önce kurmak istediğiniz uygulamayı açıp kurma işlemini tamamlayabileceksiniz.

2.7 Kaynakları İncelemek

Kaynaklarınızı açtıktan sonra açma işlemini yaptığınızı dizine girmek ve buradaki dosyaları kontrol etmek isteyeceksiniz. Genelde kurulumla ilgili bir dosyayı aramak akıllıca olacaktır. Genel olarak bu bilgiler ana kaynak dizini içerisinde yer alan README ya da INSTALL dosyaları içerisinde yazılıdır. Pratikte sizin çalıştığınız platforma özel kurulum bilgilerini README.platform ya da INSTALL.platform (burada platform kullandığınız işletim sistemini belirtiyor) dosyalarında bulursunuz.

2.8 Konfigürasyon

Modern kaynakların birçoğu ana kaynak dizini içerisinde bir konfigürasyon script dosyası barındırır. Bu script özel olarak kaynakları kurmak için tasarlanmıştır. Böylece sisteminiz üzerinde bu kaynaklar düzgün bir şekilde derlenir. Çalıştırıldığı zaman, script sisteminizi inceler yeteneklerini hesaplar ve kaynakların kurulumu ve yüklenmesine ilişkin komutları barındıran Makefile dosyaları oluşturur.

Bu script her zaman "configure" olarak adlandırılır. Eğer ana kaynak dizini içerisinde bir configure dosyası bulursanız iyi bir ihtimalle bu dosya sizin kullanmanız için oraya koyulmuştur. Eğer configure skriptini bulamıyorsanız, kaynak dosyalarınız muhtemelen çeşitli sistemler üzerinde çalışacak şekilde dizayn edilmiş standart Makefile dosyası ile gelmiştir. Bu durumda bundan sonra gelecek olan konfigürasyon adımlarını geçip dokümanı "make" hakkında konuştuğumuz yerden itibaren takip edebilirsiniz.

2.9 configure Kullanmak

configure skriptini çalıştırmadan önce bu uygulama hakkında fikir sahibi olmakta yarar var. ./configure –help yazarak programınız için mevcut tüm konfigürasyon seçeneklerini görüntüleyebilirsiniz. Özellikle –help çıktısının en üstünde olanlar ve gördüğünüz diğer seçeneklerin birçoğu neredeyse her configure skripti içerisinde yer alan standart seçeneklerdir. Sonlarda yer alan seçenekler ise sizin derlemek istediğiniz pakete özgü seçeneklerdir. Bu seçenekleri incelemek ve hangilerini kullanıp kullanmayacağımıza karar vermek gerekir.

2.10 –prefix Seçeneği

GNU autoconfige tabanlı birçok configure scripti, programı nereye kuracağınızı kontrol etmenize olanak sağlayan –prefix seçeneğine sahiptir. Öntanımlı olarak kaynakların çoğu /usr/local dizini içerisine yüklenir. Bunun anlamı binary'ler /usr/local/bin, kılavuz sayfaları /usr/local/man vb. şeklinde dosyaların yerleştirileceğidir. Normalde bizim istediğimiz de budur; /usr/local içerisinde derlediğimiz programlar saklanacaktır.

2.11 –prefix Kullanmak

Eğer kaynakları başka bir yere yüklemek isterseniz, (mesela /usr içerisine) –prefix=/usr seçeneğini configure scriptine göndermek gerekir. Aynı şekilde /opt dizini içerisine kurulum yapmak isterseniz de yine configure scriptine –prefix=/opt seçeneğini göndermemiz gerekir.

2.12 Peki ya FHS ne diyor?

Bazen bir program, kurulumda öntanımlı olarak dosyalarını disk üzerinde standart olmayan bir yere koymak isteyebilir. Yani özel olarak bir kaynak arşivi Linux Dosyasistemi Hiyerarşisine uygun olmayan

[1] http://sources.redhat.com/bzip2
[2] http://packages.debian.org/stable/utils/bzip2

kurulum yollarını kullanıyor olabilir. Neyse ki configure scripti ile kullandığımız prefix seçeneği sadece kurulum yolunu değiştirmemize izin vermekle kalmaz aynı zamanda kılavuz sayfaları gibi çeşitli sistem parçalarının da kurulum yolunu değştirmemize olanak sağlar. Birçok kaynak arşivi henüz FHS uyumlu olmadığından, prefix seçeneğinin bu özelliği çok kullanışlı olmaktadır. Muhtemelen her zaman kaynak paketinizi FHS uyumlu yapabilmek için –mandir=/usr/share/man ve –infodir=/usr/share/info seçeneklerini configure scriptine göndermeniz gerekmektedir.

2.13 Configure

Değişik konfigürasyon seçeneklerine bakıp kullanacaklarınızı seçtikten sonra, configure programını çalıştırmanın zamanı gelmiştir. Configure programını çalıştırırken, komut satırından bazı seçenekler girmek durumunda kalabileceğinizi (genellikle öntanımlı seçenekler çalışır fakat bu tam istediğiniz sonucu elde edeceğiniz anlamına gelmez) unutmayınız.

configure komutunu çalıştırmak için

```
$./configure <seçenekler>
```

```
$ ./configure
```

veya

```
$ ./configure --prefix=/usr --mandir=/usr/share/man --infodir=/usr/share/info --with-thr
```

şeklinde yazabilirsiniz. İhtiyaç duyduğunuz seçenekler, konfigüre edeceğiniz belli paketlere bağımlıdır. Configure, çalıştırdığınızda öncelikle bir süre için sisteminizde hangi araçlar ve hangi seçenekler olduğunu algılar ve bunları ekrana basar.

2.14 config.cache

Konfigürasyon işlemi sona erdiğinde configure betiği, bütün konfigürasyon verisini config.cache dosyasında depolar. Bu dosya configure betiği ile aynı yerde durur. Sistemde bir değişiklik yaptıktan sonra eğer ./configure komutunu tekrar çalıştırmanız gerekirse önce rm config.cache dosyasını silmenizde fayda var, aksi takdirde configure öncelikle bu bilgileri kullanacaktır.

2.15 configure ve Makefile Dosyaları

configure betiği çalışmasını tamamladıktan sonra kaynakları derlemenin zamanı gelmiştir. make adında bir program kullanılarak bu adım gerçekleştirilebilir. Eğer yazılım paketinizde configure betiği varsa ve bunu çalıştırdıysanız, bu betik Makefile denilen ve sisteminiz için özelleştirilmiş dosyaları yaratacaktır. Bu dosyalar make programına kaynakları nasıl kuracağını, binary'leri, kılavuz dosyalarını ve destek dosyalarını nasıl yükleyeceğini anlatır.

2.16 Makefile Giriş

Makefile dosyaları genel olarak makefile ya da Makefile olarak adlandırılır. Kaynak dosyaların bulunduğu dizinde normalde bir tane makefile olacaktır. Hazır yaratılmış Makefile dosyaları kullandığınız program gibi hedeflerin tam olarak nasıl kurulacağı gibi kuralları içerir. make programı tüm komutların çalışması gereken sırayı ve çalışma şekillerini ortaya koyar.

2.17 make Programını Çağırmak

make programını çağırmak çok kolay bir işlemdir. Çalıştığınız dizin içerisindeyken komut satırında sadece "make" yazmanız yeterlidir. Bundan sonra make programı çalıştığınız dizinde yer alan makefile ya da Makefile olarak adlandırılan dosyayı bulacak ve yorumlayacaktır. Eğer sadece make yazarsanız, program öntanımlı hedefi kuracaktır. Program geliştiriciler normalde kendi makefile dosyalarını oluştururlar ve böylece öntanımlı hedef tüm kaynağı derleyecektir.

```
$ make
```

Bazı makefile dosyalarında öntanımlı hedef yoktur ve derleme işleminin başlaması için sizin bir tane belirtmeniz gerekir.

```
$ make all
```

Bu komutlardan bir tanesini yazdıktan sonra bilgisayar programınızı object koda çevirirken biraz zaman geçecektir. Hiçbir hata olmadığını varsayarsak, bu aşamadan sonra derlenmiş programınızı artık sisteminize kurmaya hazırsınız demektir.

2.18 Kurulum

Program derlendikten sonra bir önemli adım daha kalmış demektir: kurulum. Program derlenmiş olsa bile kullanıma hazır değildir. Programın tüm bileşenleri kaynak dizinden sistem üzerinde kullanılacağı doğru dizinler içerisine kopyalanmalıdır. Mesela tüm binary dosyaları /usr/local/bin ve tüm kılavuz sayfaları /usr/local/man vb. dizinleri içerisine yüklenmelidir.

Yazılımı kurmak için root kullanıcısı olmanız gerekir. Bunun için sisteme en baştan root olarak girebilir ya da ayrı bir terminalde su yazarak root kullanıcısı olabilirsiniz. Bundan sonra o andaki kabuk oturumundan "exit" yazarak ya da control-D ye basarak çıkana kadar root haklarına sahipsiniz demektir. Eğer zaten root kullanıcısı iseniz, kurulum işlemine devam edebilirsiniz.

2.19 make install

Kaynakları kurmak için ana dizinde sadece aşağıdaki komutu yazmanız yeterlidir:

```
# make install
```

"make install" yazmakla, make programına "install" hedefini gerçekleştirmesini anlatmış oluruz. Bu hedef geleneksel olarak en son yaratılan kaynak dosyaların sisteminiz üzerinde doğru yerlere kopyalanması için kullanılır. Böylece program kullanılabilir hale gelecektir. Eğer herhangi bir –prefix seçeneği belirlemezseniz, büyük bir ihtimalle bir kaç dosya ve dizin /usr/local dizin ağacı içerisine kopyalanacaktır. Programınızın büyüklüğüne bağlı olarak kurulum hedefi tamamlanması için bir kaç saniyeden birkaç dakikaya kadar zaman geçebilir.

Ayrıca dosyaları kolaylıkla kopyalamak için, make install kurulan dosyaların doğru izin ve sahiplik haklarına sahip olduğunu da gözetecektir. make install başarıyla tamamlandıktan sonra program artık yüklenmiştir ve kullanıma hazırdır (ya da neredeyse hazırdır.)

2.20 Program bir defa yüklendikten sonra

Program yüklendikten sonra ne yapmak gerekiyor? Tabi ki onu çalıştırmak gerekmektedir. Henüz yüklediğiniz programın nasıl çalıştırılacağı konusunda bilgi sahibi değilseniz aşağıdaki gibi yazarak programın kılavuz sayfasını inceleyebilirsiniz:

```
$ man [program_adı]
```

Programınızın ancak bazı gerekli ayar adımlarının geçilmesinden sonra çalışması da olasıdır. Örneğin eğer web sunucusu yüklediyseniz, onu sistem açılışı sırasında da otomatik olarak çalışacak şekilde ayarlamanız ya da scriptlerin localhost altında calışması için konfigurasyon dosyası içerisinde birtakım ayarların yapılması gerekir. Ayrıca yine bazı programlar için de öncelikle /etc altında yer alan programla ilgili konfigürasyon dosyasını da ayarlamanız gerekebilir.

2.21 İşte bu kadar!

Artık belirli bir yazılım paketini kendi kaynağından yüklemiş bulunuyorsunuz. Aşağıdaki şekilde yazarak programınızı çalıştırabilirsiniz:

```
$ program_adı
```

Tebrikler!

2.22 Olası Problemler

configure, make hatta make install betiklerinin bir hata ile dönmesi çok karşılaşılan bir durumdur. Aşağıdaki bölümlerde bu bilinen ortak problemlerden bahsedeceğiz.

2.23 Eksik Kütüphaneler

Bazen belirli bir kütüphanenin eksik olması yüzünden configure betiğinin çalışmadığını görebilirsiniz. Kurulum işlemine devam edebilmek için, o andaki kurulum işlemini geçici olarak bir kenara bırakıp programın ihtiyacı olan bu kütüphaneler için bazı kaynak ya da binary paketleri bulmak gerekecektir. Doğru kütüphane yüklendikten sonra configure ve make betikleri işlemlerini tamamlayabileceklerdir.

2.24 Diğer Problemler

Bazen nasıl düzeltileceğini bilmediğiniz problemler ile karşılaşabilirsiniz. Unix/linux konusunda tecrübeniz arttıkça configure ve make işlemleri sırasında karşılaşacağınız anlaşılması daha zor gibi görünen problemlere tanı koyabileceksiniz.

Bazen yüklenen bir kütüphanenin çok eski ya da çok yeni olmasından kaynaklanan hatalar olabilir. Ya da program geliştiricilerinin yazdıkları programı sizin sisteminizde çalışmayacağını öngörememiş olmasından dolayı bu tür hatalarla karşılaşabilirsiniz.

2.25 Diğer Problemler (devamı)

Bu tür problemler ile karşılaştığınızda en iyi nereden yardım alabileceğinizi belirleyiniz. Eğer bu sizin programı kaynağından derlemek konusunda ilk denemeniz ise, önce derlemesi sorunsuz olacak başka bir program ile bu denemeyi yapmakta yarar vardır. Bu diğer basit programı bir defa derlediğinizde asıl programın derlenmesi sırasında meydana gelen problemin hangi noktada olduğu konusunda fikir sahibi olabilirsiniz.

3 Paket Yönetimi Kavramları, Paket Yönetimi Avantajları

Sisteminize bir yazılımı yüklemek için uygulamaları kaynaktan derlemenin yanında başka bir yöntem daha vardır. Yazılım paketlerini yüklemek, güncellemek ya da kaldırmak amacıyla tüm linux dağıtımları belli şekillerde paket yönetimleri sunmaktadır. Paket yönetimi, uygulamaların kaynaktan derlenmesine kıyasla bazı avantajlar sunar:

* Kurulum ve kaldırma kolaylığı * Mevcut kurulmuş paketleri güncelleme kolaylığı * Konfigürasyon dosyalarının korunması * Yüklenen dosyalarının takibinin basitleştirilmesi

3.1 Paket Yönetiminin Dezavantajları

En popüler paket yönetim araçlarını kullanmaya başlamadan önce paket yönetiminden hoşlanmayan linux kullanıcılarının da olduğunu belirtmekte fayda var. Bu kullanıcılara göre:

* Belli bir sistem için oluşturulan Binary'ler daha iyi performans göstermektedir. * Yüklenen bir paketin ihtiyaç duyduğu kütüphanelerden arındırılmış olması ciddi bir sorundur * Paketlerin yaratılması oldukça zordur * Paket veritabanında meydana gelecek bir bozulma sisteme zarar verebilir

Sözkonusu olumsuzluklar olsa da bir çok kullanıcıya göre paket yönetiminin avantajları dezavantajlarına baskın gelmektedir. Ayrıca yukarıda bahsedilen her kusur için de uygun bir karşıt fikir de mevcuttur. Çoklu paketler farklı sistemler için optimize edilebilirler ve paket yönetici araçları da bağımlı kütüphaneleri belirleyecek şekilde geliştirilebilir, veritabanları diğer dosyalara bağlı olarak yeniden oluşturulabilir ve bir paketin yaratılması için başta harcanan efor paketin daha sonra güncellenmesi ve kaldırılabilmesi kolaylığı sayesinde hafifletilebilir.

4 rpm the (R)ed Hat (P)ackage (M)anager, rpm'e Giriş

Red Hat'in 1995 de rpm'i ortaya koyması Linux dağıtımları için çok büyük bir adım oldu. Bu sadece Red Hat linux üzerinde paket yönetimini sağlamakla kalmayıp sahip olduğu GPL lisansı sayesinde açık kaynak paketleme açısından bir standart haline geldi. rpm için çeşitli grafik arayüzleri ve kullanımı kolaylaştıran araçlar olsa da öntanımlı olarak komut satırı üzerinde bir arayüze sahiptir. Bu bölümde örnek olarak Xsnow programını kullanarak rpm için en çok bilinen komut satırı işlemlerini inceleyeceğiz.

Eğer bundan sonrasını takip etmek istiyorsanız birçok rpm tabanlı dağıtım ile çalışabilen aşağıdaki rpm'i indiriniz:

```
xsnow-1.41-1.i386.rpm
```

Not: Bu paketi indirmek isterseniz kullanabileceğiniz kaynaklardan bir tanesi de rpmfind.net[3] olabilir.

Not: Eğer burada kullanılan çeşitli rpm terimleri aklınızı karıştırıyorsa şunu hatırlayınız: "bir rpm" bir rpm paketini temsil ederken "rpm" daima bir programı anlatmaktadır.

4.1 Bir rpm Yüklemek

Şimdi Xsnow rpm'i rpm -i kullanarak yükleyebiliriz. Tabi bu işlem için öncelikle root kullanıcısı olmak gerekir:

```
$ su
Password:
# rpm -i xsnow-1.41-1.i386.rpm
```

Eğer bu komut sonunda hiç bir çıkış üretilmiyorsa komutumuz çalışmış demektir. X masaüstünde eğlenmek istiyorsanız Xsnow uygulamasını çalıştırmanız gerekir. Kişisel olarak genelde bir rpm yüklediğimiz görsel bir geri besleme bekleriz. Bunun için -h (işlemi gösteren işaret) ve -v (görünür mod) seçeneklerini kullanmamız gerekir:

Bunun için önce rpm2i kaldıralım ve sonra -v ve -h seçenekleri ile yeniden yükleyelim:

```
# rpm -iv xsnow-1.41-1.i386.rpm
warning: xsnow-1.41-1.i386.rpm: V3 DSA signature: NOKEY, key ID b1f6e46c
Preparing packages for installation...
xsnow-1.41-1
# rpm -e xsnow
# rpm -ivh xsnow-1.41-1.i386.rpm xsnow
warning: xsnow-1.41-1.i386.rpm: V3 DSA signature: NOKEY, key ID b1f6e46c
Preparing...                ########################################### [100%]
   1:xsnow                  ########################################### [100%]
```

4.2 Bir rpm'i yeniden Yüklemek

Eğer aşağıdaki komutu tekrar çalıştırrsanız, aşağıdaki gibi bir mesaj ile karşılaşırsınız:

```
# rpm -ivh xsnow-1.41-1.i386.rpm
warning: xsnow-1.41-1.i386.rpm: V3 DSA signature: NOKEY, key ID b1f6e46c
Preparing...                ########################################### [100%]
        package xsnow-1.41-1 is already installed
```

Bir rpm tekrar yüklemek isteyeceğiniz durumlar olabilir. Örneğin /usr/X11R6/bin/xsnow binary'sini yanlışlıkla sildiğinizde bu durum ile karşı karşıya kalırsınız. Bu drumda rpm'i rpm -e ile kaldırıp tekrar yüklemelisiniz.

```
# rpm -e xsnow
```

Dikkat ediniz; birazdan göreceğiniz örnekte rpm den gelen bilgi mesajı paketin sistemden silinmesine engel teşkil etmez. Çünkü bu bilgiye göre paket zaten sistemde bulunmamaktadır.

```
# rpm -e xsnow
error: package xsnow is not installed

# rpm -ivh xsnow-1.41-1.i386.rpm
warning: xsnow-1.41-1.i386.rpm: V3 DSA signature: NOKEY, key ID b1f6e46c
Preparing...                ########################################### [100%]
   1:xsnow                  ########################################### [100%]
```

4.3 Bir rpm i Zorlama ile Yüklemek

Bazen sistemde bir programa bağlı olarak çalışan başka programlar varken, o programı sistemden kaldırmak zorluk yaratabilir. Örneğin Xsnow a bağlı olarak çalışan bir x-amusements rpm'i yüklediğinizi varsayalım. Bu durumda rpm -e ile Xsnow'u sistemden kaldıramazsınız.

[3] http://www.rpmfind.net

```
# rpm -e xsnow
error: removing these packages would break dependencies:
    /usr/X11R6/bin/xsnow is needed by x-amusements-1.0-1
```

Bu durumda Xsnow uygulamasını –force seçeneği ile yeniden yükleyebilirsiniz. Şimdi zaten yüklübir paketi -force kullanmadan yüklemeye çalışalım:

```
# rpm -ivh xsnow-1.41-1.i386.rpm
warning: xsnow-1.41-1.i386.rpm: V3 DSA signature: NOKEY, key ID b1f6e46c
Preparing...                ########################################### [100%]
        package xsnow-1.41-1 is already installed
```

Şimdi de -force kullanalım:

```
# rpm -ivh --force xsnow-1.41-1.i386.rpm
warning: xsnow-1.41-1.i386.rpm: V3 DSA signature: NOKEY, key ID b1f6e46c
Preparing...                ########################################### [100%]
   1:xsnow                  ########################################### [100%]
```

4.4 –nodeps Seçeneği İle Yüklemek ve Kaldırmak

–force kullanımına alternatif olarak bağımlılıkları kontrol etmeden de rpm'i –nodeps seçeneği ile sistemden kaldırabilirsiniz. Bu kullanım da nadiren işe yarayabilir.

```
# rpm -e --nodeps xsnow
# rpm -ivh xsnow-1.41-1.i386.rpm xsnow
```

Bir rpm'i yüklerken de –nodeps seçeneğini kullanabilirsiniz. Aslında düzgün kurulum ve kullanım açısından –nodeps seçeneğinin kullanımı pek tavsiye edilmese de bazen gerekebilir.

```
# rpm -ivh --nodeps xsnow-1.41-1.i386.rpm xsnow
```

4.5 Paketleri Güncellemek

Diyelim ki elinizde Xsnow 1.42 versiyonu için yeni bir rpm olsun. Bu durumda mevcut kurulumunuzu güncellemek isteyeceksiniz. Eğer rpm -ivh –force seçeneğini kullanırsanız kurulumu gerçekleştirirsiniz. Ancak bu durumda da rpm in kendi veritabanında her iki versiyonun da kurulu olduğu bilgisini bulacak ve paketi ya iki defa kuracak ya da bununla ilgili bir çakışma hatası verecektir.

```
# rpm -ivh xsnow-1.42-8.i386.rpm
warning: xsnow-1.42-8.i386.rpm: V3 DSA signature: NOKEY, key ID db42a60e
Preparing...                ########################################### [100%]
        file /usr/X11R6/bin/xsnow from install of xsnow-1.42-8 conflicts with
         file from package xsnow-1.41-1
        file /usr/X11R6/man/man1/xsnow.1x.gz from install of xsnow-1.42-8
         conflicts with file from package xsnow-1.41-1
```

Bunun yerine kurulumunuz güncellemek için rpm -U seçeneğini kullanmanız gerekir.

```
# rpm -Uvh xsnow-1.42-8.i386.rpm
warning: xsnow-1.42-8.i386.rpm: V3 DSA signature: NOKEY, key ID db42a60e
Preparing...                ########################################### [100%]
   1:xsnow                  ########################################### [100%]
```

Burada küçük bir ipucu verelim: Genelde nadiren rpm -i seçeneğini kullanırız. Çünkü rpm -U zaten sistemde paket yoksa yine de kurulumu gerçekleştirir. Ama -i seçeneğinden farklı olarak da eğer paket sistemde varsa varolan sürümü günceller. Bu kullanım özellikle komut satırında bazıları kurulmuş bazıları da henüz kurulu olmayan birden fazla paket yazdığınız zaman daha da kullanışlı olacaktır. Diyelim ki sistemimizde sadece xsnow-1.41 versiyonu yüklü olsun. Şimdi hem xsnow-1.42 hem de xfish tank rpm'lerini yüklemek için -U seçeneğini kullanalım.

```
# rpm -Uvh xsnow-1.42-8.i386.rpm xfishtank-2.1tp-1.i386.rpm
warning: xsnow-1.42-8.i386.rpm: V3 DSA signature: NOKEY, key ID db42a60e
Preparing...                ########################################### [100%]
   1:xfishtank              ########################################### [ 50%]
   2:xsnow                  ########################################### [100%]
```

4.6 rpm -q İle Sorgulama Yapmak

Örneklerde de farkedeceğiniz gibi rpm kurulumunda paketler için tam isim ancak rpm kaldırılması sırasında sadece isim yazılmaktadır. Bunun sebebi daha önce de bahsettiğimiz gibi rpm'in kendi içerisinde o anda kurulu olan paketlere ilişkin bir veritabanına sahip olmasıdır. Bu yüzden kurulu paketler için sadece isim kullanmak yeterlidir. Mesela rpm'e hangi Xsnow versiyonunun yüklü olduğunu soralım:

```
# rpm -q xsnow
xsnow-1.42-8
```

Aslında rpm paketlerle ilgili olarak sadece isim ve sürüm bilgisinden daha fazlasına sahiptir. rpm -qi kullanarak Xsnow hakkında daha fazla şey öğrenebiliriz:

```
# rpm -qi xsnow
Name        : xsnow                         Relocations: (not relocateable)
Version     : 1.42                               Vendor: Red Hat, Inc.
Release     : 8                              Build Date: Mon 26 Aug 2002 11:51:09 PM EEST
Install date: Thu 22 May 2003 12:37:02 PM EEST      Build Host: daffy.perf.redhat.com
Group       : Amusements/Graphics            Source RPM: xsnow-1.42-8.src.rpm
Size        : 111829                            License: MIT
Signature   : DSA/SHA1, Wed 04 Sep 2002 12:45:26 AM EEST, Key ID 219180cddb42a60e
Packager    : Red Hat, Inc. <http://bugzilla.redhat.com/bugzilla>
URL         : http://www.euronet.nl/~rja/Xsnow/
Summary     : An X Window System based dose of Christmas cheer.
Description :
The Xsnow toy provides a continual gentle snowfall, trees, and Santa
Claus flying his sleigh around the screen. Xsnow is only for the X
Window System, though; consoles just get coal.
```

4.7 rpm -ql ile Dosyaları Listelemek

rpm tarafından kullanılan veri tabanı çok fazla bilgi içermektedir. Yukarıda da gördüğümüz gibi burada yüklenen paketin sürümünü ve bununla ilişkili diğer bilgileri gözlemleyebilmekteyiz. rpm ayrıca rpm -ql kullanımı ile verilen bir kurulu pakete ait dosyaları da listeleyebilir:

```
# rpm -ql xsnow
/usr/X11R6/bin/xsnow
/usr/X11R6/man/man1/xsnow.1x.gz
/usr/share/doc/xsnow-1.42
/usr/share/doc/xsnow-1.42/README
/usr/share/pixmaps/xsnow.png
```

Bu komutun çıkışını sadece konfigürasyon dosyaları ve dökümantasyon dosyaları ile sınırlandırmak isterseniz -c ve -d seçeneklerini ekleyebilirsiniz. Bu kullanım uzun dosya listelerine sahip olan rpm ler için oldukça kullanışlıdır. Biz bu kullanımı Xsnow uygulamamız için çalıştıralım:

```
# rpm -qld xsnow
/usr/X11R6/man/man1/xsnow.1x.gz
/usr/share/doc/xsnow-1.42/README
```

4.8 rpm -qp İle Paketleri Sorgulamk

Yüklemeden önce eğer rpm hakkında mevcut bilgiyi rpm -qi ile öğrenebilseydiniz bu paketi kurup kurmayacağınıza karar verebilirdiniz. Ancak -qi ile sadece yüklü paketler hakkında ayrıntılı bilgi alabilmekteyiz. Aslında rpm -qp bir rpm için veritabanını değil rpm'in kendisini sorgulamaktadır. Bu ana kadar gördüğümüz tüm sorgulamalar kurulu paketlere uygulanabildiği gibi benzer şekilde -p seçeneği ile rpm dosyalarına da uygulanabilir. Burada az önce yaptığımız tüm örnekleri bu sefer -p seçeneği kullanarak tekrar deneyelim:

```
# rpm -qp xsnow-1.42-8.i386.rpm
warning: xsnow-1.42-8.i386.rpm: V3 DSA signature: NOKEY, key ID db42a60e
xsnow-1.42-8
```

```
# rpm -qpd xsnow-1.42-8.i386.rpm
warning: xsnow-1.42-8.i386.rpm: V3 DSA signature: NOKEY, key ID db42a60e
/usr/X11R6/man/man1/xsnow.1x.gz
/usr/share/doc/xsnow-1.42/README
```

4.9 Tüm Kurulu Paketleri Sorgulamak

-a seçeneği kullanarak sisteminize yüklenmiş tüm paketleri sorgulayabilirsiniz. eğer çıkışı sort üzerinde bir pager uygulamasına boru ile gönderirseniz sisteminizde hangi paketlerin kurulu olduğunu güzel bir şekilde görüntüleyebilirsiniz. Örneğin:

```
# rpm -qa | sort | less
```

[burada çıkışı yazmadık]
Şimdi sistemimizde kaç tane rpm'in yüklü olduğuna bakalım:

```
# rpm -qa| wc -l
    641
```

Şimdi de tüm bu rpm'lerin içerisinde kaç tane dosya olduğuna bakalım:

```
# rpm -qal| wc -l
  97322
```

Bir ipucu: rpm -qa kullanarak çoklu sistemler için kullanım kolaylığı sağlanabilir. Eğer sıralı çıkışı bir makinadaki dosyaya yönlendirip aynı işi başka makina için de yaparsak, diff programını kullanarak hangi paketlerin farklı olduğunu gözlemleyebiliriz.

4.10 Bir Dosyanın Sahibini Bulmak

Bazen verilen bir dosyanın hangi rpm2e ait olduğunu bulmak gerekebilir. Teorik olarak /usr/X11R6/-bin/xsnow dosyasına hangi rpm'in sahip olduğunu aşağıdaki gibi bir komut ile ortaya çıkarabilirsiniz:
rpm -qf ile bir dosyanın hangi rpm'e ait olduğunu sorgulayabilirsiniz.

```
# rpm -qf /usr/X11R6/bin/xsnow
xsnow-1.42-8
```

4.11 Bağımlılıkları Göstermek

–nodeps benzeri seçenekleri kullanmadığınız sürece rpm normalde bağımlılıkları bozacak işlemleri yapmanıza izin vermeyecektir. Örneğin sisteminizde öncelikle x kütüphaneleri olmadan xsnow yükleyemezsiniz. Bir defa Xsnow yüklediğinizde de öncelikle Xsnow'u kaldırmadan x kütüphanelerini de kaldıramazsınız. Bazen sorun olsa da bu bize rpm'in gücünü göstermektedir. Bunun anlamı bir rpm yüklediğinizde onun direk çalışmasını beklersiniz. rpm zaten sisteminizdeki bağımlılıkları belirlediğinden bu bağımlılıklar yüzünden fazladan iş yapmak istemezsiniz. Bazen bağımlılıkları çözmeye çalışırken, bir paketi -R seçeneği ile kullanmak yararlı olabilir. Böylece sistemde ne olması gereltiğini tam olarak belirleyebilirsiniz.

Örneğin xsnow paketi C kütüphanesine, math kütüphanesine, X kütüphanelerine ve belirli rpm versiyonlarına bağımlıdır.

```
# rpm -qpR xsnow-1.42-8.i386.rpm
warning: xsnow-1.42-8.i386.rpm: V3 DSA signature: NOKEY, key ID db42a60e
rpmlib(PayloadFilesHavePrefix) <= 4.0-1
rpmlib(CompressedFileNames) <= 3.0.4-1
libc.so.6
libc.so.6(GLIBC_2.0)
libm.so.6
libX11.so.6
libXext.so.6
libXpm.so.4
```

Ayrıca -p seçeneğini es geçerek aynı bilgi için kurulu veritabanını da sorgulayabilirsiniz.

```
# rpm -qR xsnow
```

4.12 Bir Paketin Bütünlüğünü Sağlamak

Bir web sayfasından ya da ftp sitesinde paket indirdiğimizde güvenlik amacıyla bu paketi kurmadan önce bütünlüğünü kontrol etmek isteyebiliriz. Tüm rpm'ler MD5 sum imzalıdır. Ayrıca bazı yazarlar paketlerinin güvenliğini artırmak amacıyla PGP ve GPG imzaları da kullanırlar. Bir paketin imzasını kontrol etmek için –checksig seçeneğini kullnabilirsiniz:

```
# rpm --checksig xsnow-1.41-1.i386.rpm
xsnow-1.41-1.i386.rpm: md5 GPG  NOT OK
```

Bir dakika! Bu çıkışa göre GPG imzası geçerli değildir. Neyin yanlış olduğunu anlamak için çıkışı açıklamalı durumda inceleyelim:

```
# rpm --checksig -v xsnow-1.41-1.i386.rpm
xsnow-1.41-1.i386.rpm: MD5
sum OK: 8ebe63b1dbe86ccd9eaf736a7aa56fd8 gpg: Signature made Thu 10 May
2001 01:16:27 AM EDT using DSA key ID B1F6E46C gpg: Can't check
signature: public key not found
```

O halde problemimiz yazarımızın public anahtarını alamamamız. Paketin yazarının web sitesinden anahtarı aldıktan sonra (rpm -qi çıkışında web adresi gösterilmektedir) bu key dosyasını import edersiniz. Bir anahtar import edildiğinde sizin sisteminizde keyring denilen ve içerisinde tüm anahtarların tutulduğu bir dosya içerisine yazılır. Böylece kullandığınız, indirdiğiniz döküman ve dosyalar bu anaharlara göre kontrol edilerek güvenlik sağlanmaktadır.

```
# gpg --import dan.asc
# rpm -K xsnow-1.41-1.i386.rpm
xsnow-1.41-1.i386.rpm: md5 gpg OK
```

Not: Bu kullanım sisteminizde mevcut rpm versiyonuna göre değişebilir.

4.13 Kurulmuş Bir Paketin Kontrolünü Sağlamak

Bir paketin bütünlüğünü kontrol etmeye benzer olarak, rpm -V ile yüklenmiş dosyalarınızın da bütünlüğünü kontrol edebilirsiniz.

```
# rpm -V xsnow
```

Normalde hiç bir sorun olmadığını anlatmak amacıyla bu komut hiç bir çıkış üretmez. Şimdi işleri biraz daha karıştırıp tekrar deneyelim:

```
# rm /usr/X11R6/man/man1/xsnow.1x.gz
rm: remove regular file '/usr/X11R6/man/man1/xsnow.1x.gz'? Y
# cp /bin/sh /usr/X11R6/bin/xsnow
cp: overwrite '/usr/X11R6/bin/xsnow'? y
[root@localhost erman]# rpm -V xsnow
S.5....T   /usr/X11R6/bin/xsnow
missing  d /usr/X11R6/man/man1/xsnow.1x.gz
```

Bu çıkış bize Xsnow binary'si için yapılan MD5 sum, dosya boyutu ve mtime testlerinin başarısız olduğunu göstermektedir. Ve bütün kılavuz sayfaları eksik durumdadır! Şimdi bu bozulmuş kurulumu onaralım ve uygulamamızı çalıştıralım:

```
# rpm -e xsnow
[root@localhost erman]# rpm -ivh xsnow-1.42-8.i386.rpm
warning: xsnow-1.42-8.i386.rpm: V3 DSA signature: NOKEY, key ID db42a60e
Preparing...
              ########################################### [100%]
   1:xsnow
              ########################################### [100%]
[root@localhost erman]# xsnow
Xsnow-1.42, December 14th 2001 by Rick Jansen (rja@euronet.nl)
WWW: http://www.euronet.nl/~rja/Xsnow/
```

<—— Şimdi masaüztüne bakınız :) —->

4.14 rpm'i Konfigüre Etmek

rpm nadir olarak konfigüre edilmeye ihtiyaç duyar. Eski versiyonlarda run-time operasyonları etkilemek için /etc/rpmrc içerisinde değişiklikler yapabiliyordunuz. Yeni versiyonlarda bu dosya /usr/lib/-rpm/rpmrc içerisine taşınmıştır ve sistem yöneticileri tarafından değiştirilebilir değildir. Daha çok çeşitli platformlar için uyumluluk bilgilerini ve bayrakları listeler. Eğer rpm'i konfigüre etmek istiyorsanız /etc/rpm/macros dosyasını değiştirerek bunu yapabilirsiniz. Bu nadiren yapılan bir işlem olduğu için burada ayrıntılı olarak bahsetmeyeceğiz. Aşağıdaki komut ile doğru olan dökümantasyon dosyasını bulabilirsiniz:

```
# rpm -qld rpm | grep macros
```

5 Debian Paket Yönetimi, apt-get Giriş

Debian paket yönetimi çeşitli farklı araçlardan meydana gelmektedir. Komut satırı araçlarından olan apt-get ile yeni paketleri en kolay şekilde yükleyebilirsiniz. Örneğin frozen-bubble programını yüklemek için root kullanıcısı olarak aşağıdakini yapınız:

```
apt-get install frozen-bubble
Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:
  frozen-bubble-data
The following NEW packages will be installed:
  frozen-bubble
1 packages upgraded, 1 newly installed, 0 to remove and 214  not upgraded.
Need to get 5225kB of archives. After unpacking 291kB will be used.
Do you want to continue? [Y/n] y
Get:1 http://ftp.de.debian.org testing/main frozen-bubble-data 1.0.0-3 [5155kB]
8% [1 frozen-bubble-data 423935/5155kB 8%]                    13.1kB/s 6m5s
```

Bu çıkışı gözden geçirdiğimizde frozen-bubble programının yüklendiğini, web'den getirldiğini, açıldığını ve kurulduğunu görebilirsiniz.

5.1 Taklit Kurulum

Eğer apt-get kuracağınız paketlerin başka paketlere bağlı olduğunu tespit ederse bu paketleri de otomatik olarak bulup kuracaktır. Son örnekte sadece frozen-bubble kurulmuştur. Çünkü frozen-bubble'ın bağlı olduğu tüm paketler zaten sistemde mevcut idi. Fakat bazen apt-get'in bulması gereken paket sayısı çok fazla olabilir ve kurulumu başlatmadan önce nelerin yüklenmesi gerektiğini görmekte fayda vardır. Bunu tam olarak -s seçeneği ile yapabilirsiniz. Eğer sistemimize bir icq istemcisi olan ickle isimli programı yüklemek istersek, önce hangi paketlerin yüklenmesi gerektiğini öğrenmek için:

```
# apt-get -s install ickle
Reading Package Lists... Done
Building Dependency Tree... Done
The following NEW packages will be installed:
  ickle
0 packages upgraded, 1 newly installed, 0 to remove and 214  not upgraded.
Inst ickle (0.3.2-1 Debian:testing)
Conf ickle (0.3.2-1 Debian:testing)
```

Bundan sonra da eğer isterseniz apt-get, paketlerin yükleneceği ve konfigure edileceği (ya da kurulacağı) liste sırasına göre işleme devam eder.

5.2 Paket Kaynak Listesi: apt-setup

apt-get sizin için paketleri otomatik olarak getirdiğinden, henüz kurmadığı paketleri nereden bulması gerektiğini bilmelidir. Bu bilgi /etc/apt/sources.list içerisinde bulunmaktadır.

```
# cat /etc/apt/sources.list
# See sources.list(5) for more information, especialy
# Remember that you can only use http, ftp or file URIs
# CDROMs are managed through the apt-cdrom tool.

# Security updates for "stable"
deb http://security.debian.org stable/updates main contrib non-free
deb http://security.debian.org testing/updates main contrib non-free

# Stable
deb http://ftp.de.debian.org/pub/debian stable main contrib non-free
deb http://ftp.de.debian.org/pub/debian-non-US stable/non-US main contrib non-free
....
```

<listenin devamını yazmadık>

Bu listeyi eliniz ile değiştirebilseniz bile (bkz: source.list kılavuz sayfası) bu iş için daha interaktif bir araç kullanabilirsiniz.

```
# apt-setup
```

Bu araç sizin debian paketlerini bulabileceğiniz CDROMlar, web siteleri ve ftp siteleri yerler konusunda işlem yapmanıza olanak sağlar. İşiniz bittiğinde yaptığınız değişiklikleri /etc/apt/source.list dosyasına kaydeder ve sorduğunuz zaman apt-get yine bu listedeki kaynaklarda paketleri aramaktadır.

5.3 apt-get'ten dselect'e

apt-get aracı apt-get kılavuz sayfasında da bulabileceğiniz bir çok komut satırı seçeneğine sahiptir. Öntanımlı olan seçenekler genel olarak gayet iyi çalışmaktadır ancak eğer aynı seçeneği her zaman kullandığınızı düşünüyorsanız /etc/apt/apt.conf dosyasına bir ayar eklemek isteyebilirsiniz. Bu dosyanın yazım kuralları apt.conf kılavuz sayfasında ayrıntılı olarak anlatılmaktadır.

```
# man apt.conf
```

Dediğimiz gibi apt-get bizim kullandığımız install komutunun yanında bir çok komuta sahiptir. Bunlardan bir tanesi de Debian sisteminizde yüklü her debian paketi için statü ayarına uyan apt-get dselect-upgr komutudur.

5.4 dselect'e Başlarken

dselect denilen araç bir Debian sisteminde paketleri yönetmek için kullanılan bir arayüzdür. Dselect menüsü kullanıarak sistem yöneticisi aşağıdaki ilemleri gerçekleştirebilir:

- Mevcut paket versiyonlarının listesini güncellemek
- Mevcut ve kurulmuş paketlerin durumlarını görüntülemek
- Paket seçimlerini değiştirmek ve bağımlılıkları yönetmek
- Yeni paketler yüklemek ve paketleri yeni versiyonları ile güncellemek

Her paket için durum (statü) /var/lib/dbkg/status dosyasında yer alsa da bu dosyayı güncellemek için en iyi yol yine daha interaktif bir araç olan dselect'i kullanmaktır.

```
# dselect

Debian ‘dselect’ package handling frontend.

   0. [A]ccess    Choose the access method to use.
   1. [U]pdate    Update list of available packages, if possible.
   2. [S]elect    Request which packages you want on your system.
   3. [I]nstall   Install and upgrade wanted packages.
   4. [C]onfig    Configure any packages that are unconfigured.
 * 5. [R]emove    Remove unwanted software.
   6. [Q]uit      Quit dselect.
```

```
Debian 'dselect' package handling frontend.

   0. [A]ccess    Choose the access method to use.
   1. [U]pdate    Update list of available packages, if possible.
   2. [S]elect    Request which packages you want on your system.
   3. [I]nstall   Install and upgrade wanted packages.
   4. [C]onfig    Configure any packages that are unconfigured.
 * 5. [R]emove    Remove unwanted software.
   6. [Q]uit      Quit dselect.

Move around with ^P and ^N, cursor keys, initial letters, or digits;
Press <enter> to confirm selection.   ^L redraws screen.

Version 1.10.9 (i386).
Copyright (C) 1994-1996 Ian Jackson.
Copyright (C) 2000,2001 Wichert Akkerman.
This is free software; see the GNU General Public Licence version 2
or later for copying conditions.  There is NO warranty.  See
dselect --licence for details.
Move around with ^P and ^N, cursor keys, initial letters, or digits;
Press <enter> to confirm selection.   ^L redraws screen.

Version 1.10.9 (i386).
Copyright (C) 1994-1996 Ian Jackson.
Copyright (C) 2000,2001 Wichert Akkerman.
This is free software; see the GNU General Public Licence version 2
or later for copying conditions.  There is NO warranty.  See
dselect --licence for details.
```

5.5 dselect Select Modunu Kullanmak

Her paketin durumunu görüntülemek ve değiştirmek için Select seçeneğini seçebilirsiniz. Bundan sonra ekran dolusu yardım görüntülenecektir. Bu yardımı okuduktan sonra <enter> tuşuna basınız. Daha sonra aşağıdakine benzer şekilde bir paket listesi göreceksiniz:

```
 EIOM Pri Section  Package      Inst.ver    Avail.ver   Description
    - All packages -
    --- Obsolete and local packages present on system ---
    ----- Obsolete/local Required packages -----
    ------- Obsolete/local Required packages in section base -------
 *** Req base     adduser      3.50        <none>
 *** Req base     base-files   3.0.8       <none>
 *** Req base     base-passwd  3.5.3       <none>
 *** Req base     bash         2.05b-7     <none>
 *** Req base     bsdutils     2.11z-1     <none>
 *** Req base     console-tool 0.2.3dbs-32 <none>
All packages
The line you have highlighted represents many packages; if you ask to
install, remove, hold, &c it you will affect all the packages which
match the criterion shown.

If you move the highlight to a line for a particular package you will
see information about that package displayed here. You can use 'o' and
'O' to change the sort order and give yourself the opportunity to mark
packages in different kinds of groups.
```

5.6 Paket Durumu

Her paketin durumu kapalı bir başlık olan EIOM altında yer almaktadır. Bizi ilgilendiren kolon her paketin aşağıdakilerden bir tanesi ile işaretlendiği M karakteri altındadır:

İşareti değiştirmek için istediğiniz koda göre bir tuşa basınız (eşit, dash ya da altçizgi). Eğer işareti * (asteriks) karakterine değiştirmek istiyorsanız + tuşuna basmalısınız. Bunu yaptıktan sonra büyük harf Q kullanarak yaptığınız değişiklikleri kaydedip Select ekranından çıkabilirsiniz. Eğer dselect içerisinde

yardım almak isterseniz herhangi bir zamanda ? tuşuna basarak yardım alabilirsiniz. Sonra da tekrar space tuşuna basarak yardım ekranından çıkabilirsiniz.

5.7 Kurulum ve Konfigürasyon

Debian paketlerin durum ayarlarına bağlı olarak siz apt-get dselect-upgrade yazana kadar paketleri yüklemez ya da kaldırmaz. Bu komut aslında sizin için bir seferde birkaç tane adımı gerçekleştirir: Kurulum, Kaldırma ve Konfigürasyon. Kurulum ve Kaldırma adımları sırasında soru sormak için program durmaz. Fakat konfigürasyon adımında paketlerin sizin istediğiniz yolla konfigüre edilmesini sağlamak amacıyla bir takım sorular sorulmaktadır. Bu adımları çalıştırmak için başka yollar da vardır. Örneğin her adımı ayrı olarak dselect menüsünden seçebilirsiniz. Bazı paketler kendi konfigürasyon adımları için debconf denilen bir sistem kullanırlar. Böylece kendi kurulumları ile ilgili soruları text terminali, grafik arayüz veya web sayfası gibi çeşitli ortamlarda sorabilirler.

5.8 dpkg İle Kurulmuş bir Paketin Durumunu Ele Almak

Uzun paket listeleri için şu ana kadar görmüş olduğumuz Debian paket yönetim araçları birden fazla adımdan oluşan operasyonlar için kullanılabilecek en iyi araçlardır. Ancak yine de bu araçların paket yönetimi konusunda eksik olduğu yönler vardır. Bu durumlar için dpkg kullanmak isteyeceksiniz. Örneğin bir paketin tüm durum ve açıklamalrını görmek için -s seçeneğini kullanınız:

```
#dpkg -s ickle
Package: ickle
Status: deinstall ok config-files
Priority: optional
Section: net
Installed-Size: 1364
Maintainer: Leo Costela <costelaaa@ig.com.br>
Version: 0.3.2-1
Config-Version: 0.3.2-1
Depends: ickle-common (= 0.3.2-1), libc6 (>= 2.2.4-4),
    libglib1.2 (>= 1.2.0), libgtk1.2 (>= 1.2.10-4),
    libgtkmm1.2, libicq2000 (= 0.3.2-1), libsigc++0,
    libstdc++2.10-glibc2.2 (>= 1:2.95.4-0.010810),
    xlibs (>> 4.1.0)
Suggests: ickle-control (= 0.3.2-1)
Description: An ICQ2000 client for GTK+
 Ickle is an ICQ client which is able to communicate with ICQ servers
 using the new ICQ2000 protocol. This offers much better reliability
 than the older ICQ99 protocol. ickle has a constantly improving
 interface, now supporting history browsing, and much more.
 .
 Ickle currently supports:
    * Sending/Receiving messages, URLs and SMS messages
    * Retrieving away messages
    * Fetching user info from server
    * Fetching/setting away messages
    * Custom away messages
    * Authorisation Requests
    * "User Added You" messages
 .
 This version is compiled without gnome support, for gnome support
 install the package ‘ickle-gnome’.
```

5.9 Bir dosya ile .deb Arasındaki Bağlantı

.deb paketi dosyaları içerdiğinden dolayı, paketin içerisinde yer alan dosyaları listelemenin bir yolu olduğunu düşünebilirsiniz. Evet böyle bir şey yapmanız mümkündür ve bunun için -L seçeneğini kullanmanız gerekmektedir:

```
# dpkg -L frozen-bubble
/.
```

```
/usr
/usr/lib
/usr/lib/menu
/usr/lib/menu/frozen-bubble
/usr/lib/perl5
/usr/lib/perl5/auto
/usr/lib/perl5/auto/fb_c_stuff
/usr/lib/perl5/auto/fb_c_stuff/fb_c_stuff.bs
/usr/lib/perl5/auto/fb_c_stuff/fb_c_stuff.so
/usr/games
/usr/games/frozen-bubble
/usr/games/frozen-bubble-editor
/usr/share
/usr/share/doc
/usr/share/doc/frozen-bubble
/usr/share/doc/frozen-bubble/README
/usr/share/doc/frozen-bubble/AUTHORS
/usr/share/doc/frozen-bubble/copyright
/usr/share/doc/frozen-bubble/changelog.gz
/usr/share/doc/frozen-bubble/changelog.Debian.gz
/usr/share/man
/usr/share/man/man6
/usr/share/man/man6/frozen-bubble.6.gz
/usr/share/man/man6/frozen-bubble-editor.6.gz
/usr/share/gnome
/usr/share/gnome/apps
/usr/share/gnome/apps/Games
/usr/share/gnome/apps/Games/frozen-bubble.desktop
/usr/share/perl5
/usr/share/perl5/fbsyms.pm
/usr/share/perl5/FBLE.pm
/usr/share/perl5/fb_c_stuff.pm
/usr/share/perl5/fb_stuff.pm
/usr/share/pixmaps
/usr/share/pixmaps/frozen-bubble-icon-48x48.png
/usr/share/pixmaps/frozen-bubble-icon-16x16.png
/usr/share/pixmaps/frozen-bubble-icon-32x32.png
/usr/share/pixmaps/frozen-bubble.xpm
```

Bunun tersine belirli bir dosyayının hangi pakete ait olduğunu anlamak için de -S seçeneğini kullanabilirsiniz.

```
# dpkg -S /usr/share/pixmaps/frozen-bubble.xpm
frozen-bubble: /usr/share/pixmaps/frozen-bubble.xpm
```

Gördüğünüz gibi kolonun solunda paketin ismi belirtilmektedir.

5.10 Yüklenecek Paketleri Bulmak

Genellikle apt-get ihtiyacınız olan bir Debian paketini zaten biliyor olacaktır. Eğer bu olmazsa bu debian paketleri listesi içerisnden ya da web den bu paketleri bulabilecek durumda olmalısınız. Eğer .deb dosyasını bulur ve indirirseniz -i seçeneğiyle kurabilirsiniz:

```
# dpkg -i /tmp/dl/xsnow_1.40-6_i386.deb
```

eğer .deb olarak aradığınız dosyayı bulamayıp yerine .rpm ya da başka bir paket formatında bulduysanız alien programını kullanabilirsiniz. alien programı paketleri çeşitli formatlardan .deb formatına çevirmektedir.

5.11 Dpkg İle Bir Paketi Konfigure Etmek

Bu paketlerden bir tanesini konfigüre etmek için dbkg –configure komutu kullanmalısınız. Bu kullanım yüklenmiş bir programın dosyaları ya da programa ait ayar dosyaları zarar gördüğü zaman işe yarayabilir.

5.12 Ek Debian Paket Yönetim Araçları

Burada anlattıklarımıza ilave olarak, daha birçok Debian paket yönetimi araçları bulabilirsiniz. Bunun için Debian anasayfasına, Debian kurulum rehberine, Debian paket listelerine ve Alien anasayfasına bakabilirsiniz.

6 Huzurlarınızda çekirdek... Linux!

Genel olarak Linux ismi, Linux dağıtımı adı altında geçen, programların birarada çalışmasından oluşan dağıtımları tanımlamak için kullanılır. Halbuki teknik olarak Linux sadece çekirdek' (kernel)in ismidir. Herne kadar Linux diye adlandırıldığımız diğer parçalar (kabuk, derleyiciler vs.) sistemin çalışması ve tam bir işletim sistemi oluşturması için aynı derecede gerekli olsalar da teknik olarak çekirdekten ayrıdırlar. Yine de insanlar "Linux" kelimesini "Linux tabanlı dağıtım" olarak kullanmaya devam edeceklerdir. Yine de herkesin hemfikir olacağı nokta Linux çekirdeğinin her Linux İşletim sisteminin kalbini oluşturduğudur.

6.1 Donanım ile arabirim oluşturmak

Linux çekirdeğinin temel görevi sisteminizde donanıma erişim için direkt bir arabirim sağlamaktır. Çekirdek ham donanım ve uygulama programları arasında bir soyutlama katmanı sağlar. Böylelikle programlar ana kartınızın veya disk kontrol ünitesinin detaylarını bilmek durumunda kalmadan çok daha üst seviyede diskten okuyup ona bilgi yazabilirler mesela.

6.2 CPU Soyutlama

Linux çekirdeği aynı zamanda sisteminizdeki işlemciye de bir soyutlama katmanı sağlar - bu sayede birçok program eş zamanlı çalışıyormuş izlenimi verir. Herbir programın gerektiği kadar işlemci zamanı ve diğer kaynaklardan yararlanmasını çekirdek ayarlar.

Şu anda bir Linux sistem üzerinde çalışıyorsanız, kullandığınız çekirdek ya UP(uniprocessor (tek işlemcili)) ya da SMP(symmetric multiprocessor(simetrik çok işlemcili)) bir çekirdektir. Eğer bir SMP ana karta sahipseniz ve UP çekirdek kullanıyorsanız, Linux'unuz ekstra işlemcilerden haberdar olmayacaktır. Bunu düzeltmek için kendi donanımınıza uygun bir SMP çekirdek derlemeniz gerekir. Halihazırda SMP çekirdekler, küçük bir performans kaybıyla da olsa UP sistemlerde de çalışacaklardır.

6.3 IO(giriş/çıkış)'ları Soyutlama

Çekirdek aynı zamanda her türlü dosya girdi çıktılarını soyutlama işini de üstlenir. Her bir programınızın bu işleri kendi başına yaptığını bir düşünün. Disk kontrolörlerinizi değiştirdiğiniz takdirde bütün programlarınız çakılıp kalacaklardı. Bu yüzden Linux çekirdeği de programların kullanabileceği veri depolama ve erişimi için soyutlama için Unix modellerini kullanmaktadır. Bu şekilde kullandığınız veritabanı, veriyi IDE ya da SCSI RAID dizisinde ya da ağ üzerinden erişilen bir dosya sisteminde tutup tutmadığınızla ilgilenmeyecek, sadece o veri ile işini yapacaktır.

6.4 Ağ Merkezi

Linux'un en ünlü ve iddialı olduğu alanlardan biri oldukça güçlü ağ yapısı, özellikle de TCP/IP desteğidir. Eğer şimdiye kadar TCP/IP desteğinin Linux çekirdeğinde yeraldığını tahmin ettiyseniz, haklısınız. Çekirdek, standartlarla uyumlu, yüksek seviyeli bir arabirim ile ağ üzerinden veri gönderilmesini ve alınmasını destekler. Arka plnda ise Linux çekirdeği sizin ethernet kartınız ya da pppd daemon ile düşük seviyeli bağlantıları halleder ve Internet iletişimini sağlar. Bu konunun detayları bir sonraki dersin konusudur.

6.5 Ağın Faydaları

Linux'un en iyi özelliklerinden biri çekirdekte yeralan birçok yararlı özelliktir. Mesela bir Linux çekirdeğini, bütün evinizdeki ağın modem üstünden Internet'e çıkış kapısı olması için konfigüre edebilirsiniz –buna IP Masquerading veya IP NAT denir. Bunun dışında Linux çekirdeği ağdaki diğer Unix makinalarla dosya paylaşabilecek şekilde de konfigüre edilebilir. Çekirdekte buna benzer bir çok faydalı özellikl vardır, bunları çekirdek özelliklerini incelerken daha detaylı olarak da göreceksiniz.

6.6 Açılış işlemleri

Sanırız, şu an Linux'un açılşta hangi aşamalardan geçtiğini gözden geçirmek için tam da zamanı. Linux tabanlı sisteminizi açtığınızda çekirdek imajı (tek bir ikilik dosyadır) diskten hafızaya Lilo ya da Grub gibi bir yükleyici tarafından yüklenir ve ardından çekirdek sistemin kontrolünü alır. Yaptığı ilk şeylerden biri mevcut donanımları gözden geçirmek ve desteklemek üzere konfigüre edildiği donanımları tanımaktır. Donanımlar doğru olarak tanınıp işler hale getirildiklerinde, çekirdek normal kullanıcı alanında çalışan programları (process(işlemler) diye de geçer bunlar) çalıştırır. İlk çalıştırılan program /sbin/init'dir. Init ise /etc/inittab altında tanımlanan diğer programları çalıştırır. Saniyeler içinde sisteminiz ayağa kalkmış ve çalışır halde, kullanımınıza hazırdır. Herne kadar hemen hiçbir zaman çekirdeğe doğrudan erişim sağlamasanız da, Linux çekirdeği bütün normal programların "üstünde" aralıksız çalışmakta, bu programların ve kütüphanelerin ihtiyaç duydukları sanallaştırma ve soyutlama işlemlerini çeşitli katmanlarda gerçekleştirmektedir.

6.7 Ve huzurlarınızda... modüller

Yakın tarihli bütün Linux çekirdekleri, çekirdek modüllerini destekler. Çekirdek modülleri oldukça hoş şeylerdir -basitçe çekirdeğin bir parçası olan, göreceli olarak çok daha küçük dosyalardan oluşurlar. Çekirdek belli bir fonksiyonu gerçekleştirmek için bir modüle ihtiyaç duyduğunda o modülü diskten otomatik olarak yükler ve kendisine entegre eder, böylelikle dinamik olarak yetkinliğini arttırabilir. Eğer bir çekirdek modülünün özellikleri bir süre kullanılmazsa, çekirdek onu hafızdan çıkartabilir ve böylelikle autocleaning denen bir işlemi gerçekleştirerek gereksiz kaynak kullanımını azaltır. Çekirdek modülleri olmasa, çalışmakta olan çekirdeğimize (ki kendisi tek bir dosyadan oluşur) bütün gereken özellikleri yüklememiz gerekecekti. Normalde ise temel ihtiyaçları sağlayacak bir çekirdek derleyip, ilerde ihtiyaç duyabileceğimiz modülleri ekleriz. Eğer gerekirse uygun modül çekirdeğe daha sonra dinamik olarak eklenebilir. Bu aynı zamanda RAM ve diğer kaynakların da korunması anlamına gelir, çünkü bir modül ancak gerek duyulduğunda yüklenir ve atıl hale geldiğinde hafızadan çıkartılır.

6.8 Modüller nerede durur ?

Çekirdek modülleri genelde /lib/modules/x.y.z (x.y.z modüllerin uyumlu olduğu çekirdek sürümünü belirtir) altında dururlar; her modülün isminin sonunda ".o" uzantısı vardır, bu sayede bu dosyalar makina dili komutları içeren dosyalar olarak tanımlanırlar. Tahmin edebileceğiniz gibi her modül, belli bir çekirdek fonksionalitesini tanımlar. Bir modül FAT dosyasistemi desteği sağlayabilirken bir başkası ISA ethernet desteği sağlayabilir.

6.9 Modüller – her işlem için kullanılamaz

Herşeyi bir modül haline getiremeyeceğinizi belirtmekte fayda var. Modüller disk üzerinde durdukları için sizin başlatılabilir linux çekirdeğinizin kendi içinde mevcut disk kontrolörünüzün desteğine sahip olması gerekir. Eğer bu destek yoksa (veya bu desteği modüler olarak derlediyseniz) çekirdeğiniz bu modülleri yüklemeyi beceremeyecektir (çünkü modüllerin yüklü olduğu diske erişim yeteneğinden yoksundur) ve bu yüzden de sisteminiz başlatılamayacaktır.

7 Çekirdek kaynaklarını bulup bilgisayarınıza indirmek

Bu döküman yazıldığı sıralarda en yeni çekirdek 2.4.18'di. 2.4.18 çekirdek 2.4 tutarlı çekirdek serisinin bir parçasıdır. Bu seri gerçek dünyada üretim amaçlı kullanılacak sistemler için hazırlanmaktadır. Bu seri ile eş zamanlı olarak geliştirilen 2.5 serisi de kullanıma açıktır, fakat bunların üretim sistemlerinde kullanılması tavsiye edilmemektedir. 2.5'deki 5 sayısı tek bir sayıdır, ve çekirdek terminolojisinde bu serinin çekirdek geliştiricilerine ve deneme kullanıcılarına yönelik deneysel bir sürüm olduğunu belirtir. 2.5 sürüm çekirdekler, yeteri kadar güvenilir hale gelip, üretim sistemlerinde kullanılabilir hale ulaştıklarında 2.6 sürüm numarasını alacaklardır (çift sayıya dikkat).

7.1 Hangi çekirdek kaynaklarını kullanmalı ?

Eğer mevcut çekirdeğinizin yeni bir sürümünü derlemek istiyorsanız (mesela SMP desteği eklemek isteyebilirsiniz) en iyi yöntem dağıtımınızın çekirdek kaynak kodu paketini kurmaktır. Bunu kurduktan sonra /usr/src/linux dizini altında bazı dosyaların hazır olduğunu göreceksiniz. Bunun dışında tamamen yeni

bir çekirdek kurmayı da isteyebilirsiniz. Genelde en doğru yaklaşım dağıtımınızın yeni ya da güncellenmiş kaynak kodu paketinden kurulum yapmaktır. Bu paket sizin Linux sisteminizde optimal olarak çalışmak üzere yamanmış ve konfigüre edilmiş çekirdek kaynak kodlarını içerecektir.

7.2 Çekirdeği kaynağından edinmek

Eğer macera meraklısıysanız, çekirdek kaynak kodunu http://www.kernel.org/pbu/linux/kernel adresinden "genel bir" kaynak kodu indirerek edinebilirsiniz. Bu dizinde Linus ya da Marcello tarafından yayınlanmış resmi kaynak kodlarını bulabilirsiniz. Burada önemli nokta dağıtımınızda bulunan bütün özellikleri bu çekirdeğin desteklemeyebileceği. Bu sebeple tam olarak ne yaptığınızdan emin olmadan böyle bir "genel" çekirdek derlememenizde fayda var.

Kernel.org'da değişik dizinler altında düzenlenmiş çekirdek kaynak kodlarını bulacaksınız. Kernel (v2.2, v2.4 vs.) Her dizinin içinde "linux-x.y.z.tar.gz" ve "linux-x.y.z.tar.bz2" isimli dosyalar bulunacaktır. Bunlar temel kaynak kodu dosyalarıdır. Bunların dışında "patch-x.y.z.gz" ve "patch-x.y.z.bz2" dosyaları da vardır. Bunlar ise önceki sürümleri güncellemeye yarayan yama dosyalarıdır. Eğer yeni bir çekirdek derleyecekseniz "linux" diye başlayan dosyalardan birini almanız gerekir.

7.3 Çekirdek kodu paketini açmak

kernel.org'dan yeni bir çekirdek kodu paketi aldıysanız önce bunu açmak gerekecektir. Bunun için önce /usr/src dizinine gidi. Eğer burada bir "linux" dizini varsa onu "linux.old" dizini haline getirin (mv linux linux.old tabi bunu root olarak yapıyorsunuz) Şimdi yeni çekirdeği açmanın zamanı. /usr/src altındayken tar zxvf /kernelin/bulundugu/yer/linux-x.y.z.tar.gz ya da cat /kernelin/bulundugu/yer/-linux-x.y.z.tar.bz2 — bzip2 -d — tar xvf -. Buradaki fark dosyaların bzip2 ya da gzip ile sıkıştırılmasından kaynaklanmaktadır. Yeni linux dağıtımlarında tar jxvf /kernelin/bulundugu/yer/linux-x.y.z.tar.bz2 -C /usr/src/linux/ komutu da tar.bz2 dosyasının açılmasını sağlayacaktır.

Bu komutu verdikten sonra çekirdek kaynak kodları sözkonusu dizine açılacaktır. Yalnız unutmamanız gereken nokta bu dosyaların açık halinin muhtemelen 50MB'dan fazla tutacağıdır. Buna göre yer açmanızda fayda var harddiskinizde.

8 Çekirdeği Düzenlemek

Çekirdeği derlemeden önce bazı düzenlemeler yapmamız gerekecek. Bu düzenlemeler, yeni çekirdeğimizde hangi özelliklerin olacağını ve hangilerinin olmayacağını belirleyecek. Bu sırada hangi özelliklerin ana çekirdek dosyasında yeralacağı (ve boot işlemi sırasında yükleneceği) hangilerinin ise daha sonradan eklenebilir modüller olarak derleneceğine de karar vereceğiz.

Eski zamanlarda çekirdeği düzenlemek ve derlemek ciddi bir problemdi. Zira config betiğini çaıştırmayı gerektiriyordu. Herne kadar hala bu imkan mümkün olsa da biz pek tavsiye etmiyoruz, tabi yüzlerce soruya teker teker cevap vermek isterseniz siz bilirsiniz.

8.1 Yeni yöntem

/usr/scr/linux dizinine girdikten sonra "make config" yerine "make menuconfig" ya da "make xconfig" komutunu çalıştırabilirsiniz. make menuconfig güzel bir konsol tabanlı ve renkli menu sistemi sunmakta, make xconfig ise benzer bir sistemi x tabanlı bir arabirim ile kullanmanıza izin vermekte.

make menuconfig kullandığınızda dikkat etmeniz gereken noktalardan biri, solunda < > bulunan seçeneklerin modül olarak derlenebileceği.

Üzerinde durduğunuz bir seçeneği space tuşuna basarak çekirdeğe dahil edebilir "<*>", çekirdekten çıkartabilir "< >" ya da modül olarak derlemek üzere seçebilirsiniz "<M>". Bunun dışında y tuşu ile ekleme, n tuşu ile çıkartma ve m tuşu ile modül yapmayı direkt seçmeniz de mümkün. Birçok çekirdek seçeneğinin detaylı açıklamalara sahip olması da bir başka şanstır, bu bilgilere erişmek için de h tuşuna basmanız gerekir.

8.2 Konfigürasyon ipuçları

Malesef hepsini burada açıklayamayacağımız kadar çok konfigürasyon seçenekleri mevcuttur. Bunun yerine size önemli kategorilerin açıklamalarını yapmayı uygun bulduk.

Code maturity level options (Kod olgunluğu seviye seçenekleri)

Şimdi make menuconfig ya da make xconfig seçeneği ile çekirdek düzenleme ortamına girelim.

Kod olgunluğu seviye seçenekleri: Bu kategoride tek bir seçenek vardır. "Prompt for development and/or incomplete code/drivers" (geliştirme ve/veya tamamlanmamış seçenekleri seçebilmek ister misiniz ?) Eğer bu seçeneği seçerseniz, deneysel kabul edilen bir çok özellik (ReiserFS?, devfs vs.) kendileri ile ilgili kategoriler altında erişilebilir olacaklardır. Eğer bu seçenek seçilmediyse göreceğiniz bütün özellikler "kararlı" yapı altında kullanılmakta olan özellikler olacaktır. Bunu seçmeniz, seçim şanslarınızı arttıracağından iyi olur.

Modüller ve İşlemci ile alakalı seçenekler: Loadable module support (Yüklenebilir modül desteği)

Bu konfigürasyon kategorisinin altında çekirdeğin modül desteği ile alakalı üç seçenek vardır. Genelde üçünün de seçili olması gerekir.

Processor type and features (işlemci cinsi ve özellikleri): Bu bölüm çeşitli işlemci konfigürasyon seçeneklerini içermektedir. Burada "Symmetric multiprocessor support" özellikle önemlidir ve birden fazla işlemciye sahip makinalar için elzemdir. Aksi takdirde sisteminizdeki ilk CPU algılanacaktır. "MTTR Support" seçeneği genellikle seçilir , çünkü modern sistemlerde X konusunda performans artışı sağlamaktadır.

General and parallel port options: (Genel ve paralel port seçenekleri)

General setup (Genel düzen): Bu bölümde Networking ve PCI Support seçenekleri genel olarak seçili olmalıdır. "Kernel support for ELF binaries" (Elf çalışabilir programları için çekirdek desteği) de. a.out ve MISC binary seçenekleri de tavsiye edileir fakat bunların modül olarak derlenmesi daha iyi olacaktır. "System V IPC" ve "Sysctl support" seçeneklerini de seçmeye özen gösterin. Bu seçenekler hakkında daha fazla bilgi edinmek isterseniz help seçeneği ile detaylı bilgiler alabileceğiniz daha önce belirtmiştik.

"Paralel port" seçeneği özellikle yazıcı kullananlar için gerekecektir. Tam bir yazıcı desteği için "Character Devices" (karakter aygıtları) bölümünden "Paralel Printer Support" (Paralel yazıcı desteği) kısmını da seçmek gerekir.

RAID ve LVM Multi-device support (RAID and LVM)(Çoklu aygıt seçimi): Bu bölümde Linux'un yazılımsal RAID ve mantıksal bölüm yöneticisi özellikleri seçilir. Yazılımsal RAID, disklerinizin birbirleri ile yedekli olarak çalışmasını ve bu sayede performans ve güven artışını sağlamayı amaçlar. Yazılımsal RAID konusunda detaylı bilgilere sondaki kaynaklar bölümünden ulaşabilirsiniz.

Networking and related devices(Ağ ve ilgili aygıtlar) Networking options (Ağ seçenekleri): Bu bölümde her türlü ağ seçenekleri mevcuttur. Linux sisteminizi tipik bir ağa bağlayacaksanız, "Packet socket" "Unix domain sockets" ve "TCP/IP Networking" seçeneklerini seçmeniz gerekmektedir.

Bu temel seçenekler dışında bir çok seçenek daha, mesela "Network packet filtering" (Ağ paket filtreleme, ip tables kullanmak, firewall kurmak için) bulunmaktadır.

Çekirdeğinizin desteklemesini istediğini ethernet kartlarını da belirtmeniz gerekmektedir. Aradığınız kart muhtemelen "Ethernet (10 or 100MBit) alt kategorisinin altında bulunmaktadır.

IDE support(IDE desteği) ATA/IDE/MFM/RLL support: Bu bölümde IDE sürücüler, CD ve DVD Rom'lar ve diğer benzer çevrebirimler kullananlara özgü seçenekler yeralmaktadır. Eğer sisteminizde IDE diskleri varsa "Enhanced IDE/MFM/RLL disk/cdrom/tape/floppy support" seçeneğini seçin.

Bunun dışında "Include IDE/ATA-2 DISK support"(IDE/ATA2 disk desteğini kat) ve mevcut ana kartınıza uygun chipset seçeneklerini seçin ve bunu çekirdeğe ekleyin, modül olarak derlemeyin ki sisteminiz boot edebilsin). Eğer IDE CD Rom'a sahipseniz, "Include IDE/ATAPI CD-ROM support" seçeneğini de eklemeyi unutmayın. Anakartınızın chipset desteğini seçmeden de IDE disklere muhtemelen ulaşabileceksiniz fakat bu durumda ana kartın performans avantajlarından yararlanmamış olduğunuzdan daha yavaş bir erişimle karşı karşıya kalabilirsiniz.

"Enable PCI DMA by default if available" (eğer varsa PCI DMA desteğini aç) seçeneği de heme hemen bütün sistemler için tavsiye edilen bir özelliktir. DMA olmadan IDE çevrebirimleri PIO modunda ve duruma göre potansiyellerinin on onbeş katı daha yavaş çalışabilirler.

Belli bir disk için DMA'nın etkin hale geldiğini hdparm -d 1 /dev/hdx ile root olarak kontrol edebilirsiniz. Burada /dev/hdx DMA'i etkin hale getirmek istediğiniz diskin adıdır.

SCSI support(SCSI desteği) Bu kategori SCSI diskler ve çevrebirimlerle ilgili bütün seçenekleri barındırır. Eğer SCSI tabanlı bir sisteminiz varsa "SCSI support", "SCSI disk support", "SCSI CD-ROM support ve "SCSI tape support" seçeneklerinizi ihtiyaçlarınıza göre seçebilirsiniz.

IDE disklerde olduğu gibi eğer sistemin yüklü olduğu disk SCSI bir disk ise bu destekleri yüklenebilir modüller olarak değil ana çekirdek dosyasının içine koymanızda fayda var.

Bunlardan başka SCSI'nin gerektiği gibi çalışabilmesi için "SCSI low-level drivers" bölümüne gidip, SCSI kartınızın seçili olduğundan emin olun, yine bu seçenek de çekirdeğin içinde direkt yer almalıdır, modül olarak değil.

Miscellaneous character devices (Çeşitli karakter aygıtları): Character devices: Bu bölüm çeşitli çekirdek aygıt sürücülerini içerir. "Virtual terminal" ve "Support for console on virtual terminal" seçeneklerinin

seçili olduğundan emin olun. Bunlar sistem açıldıktan sonra standart text tabanlı terminallerin sizi karşılaması için gereklidirler.

Büyük ihtimalle "Unix98? PTY support"u da seçmenizde fayda var. Bunlar dışındaki her şeyi seçimi size kalmıştır. Eğer paralel yazıcı kullanmak istiyorsanız "Paralel printer support" seçeneğini seçmeniz gerektiğini tekrar hatıarlatalım.

"Enhanced real-time clock support" tavsiye edilir, "/dev/agpgart (AGP desteği)" ve "Direct Rendering Manger" da X altında performans sağlayan Linux 3D hızlandırıcılı kartlar için gereklidir. X'i bu hızlı modda çalıştırmak için çekirdek derleme dışında da bazı ayarlar yapmak gerekecektir.

File systems and console drivers(Dosya sistemleri ve konsol sürücüleri): File systems: Bu kategorideki seçenekler tahmin edeceğiniz gibi dosya sistemlerini içermektedir. Root (/) dosya sisteminiz için gerekli olan dosya sistemi desteğini (ext3, reiserfs, jfs, xfs vs.) çekirdeğe dahil etmeniz gerekmektedir. Bundan başka "/proc file system support" seçeneği de bir çok dağıtım için elzemdir.

Genel olarak "/dev/pts file system support for Unix98? PTYs" seçeneğini seçmeniz tavsiye edilir, tabi "/dev file system support" seçeneğini seçmiyorsanız. O takdirde "/dev/pts" seçeneğini seçilmemiş bırakmanız gerekir.

Console drivers: Tipik olarka herkez "VGA text console" (x86 sistemler için elzemdir) seçecektir. İsterseniz "Video mode selection support" de seçebilirsiniz. "Frame buffer support" seçtiğiniz takdirde ise text tabanlı ekran yerine grafik ekranda yazıları görebilirsiniz. Fakat bu seçeneklerini bazıları X ile karışıklıklara sebep olmaktadır bu yüzden düz VGA konsola bağlı kalmakta, en azından başlangıçta fayda vardır.

9 Çekirdeği derlemek ve kurmak

Çekirdeği bir kez konfigüre ettikten sonra artık derlemeniz gerekir. Fakat bunu yapmak için önce bağımlılıkları öğrenmelisiniz. Bunu basitçe /usr/src/linux dizini içinde make dep komutunu çalıştırarak yapabilirsiniz.

9.1 make bzImage

Artık asıl çekirdek imajını derlemenin zamanı geldi. Yine aynı dizinde make bzImage yazıp entera basın. Makinanızın özelliklerine ve seçtiğiniz seçeneklere göre yarım saat kadar bir süre sonra (veya çok daha uzun) /usr/src/linux/arch/i386/boot (x86 PC çekirdeği için) altında bzImage dosyasını oluşturulmuş bulacaksınız.

Bu imajı nasıl sisteme kuracağınıza da değineceğiz ama şimdilik modüllere bir bakalım.

9.2 Modülleri derlemek

bzImage'ı oluşturduğumuza göre, modülleri derlemenin zamanı geldi demektir. Eğer çekirdek seçeneklerinden hiçbirinde modül seçeneğini işaretlemediyseniz bile bu basamağı atlamayın, modülleri bzImage'ın hemen ardından derlemek en doğru harekettir. Zaten eğer modülleri etkin hale getirmediyseniz bu aşama oldukça hızlı geçecektir.

```
make modules && make modules_install
```

yazın ve entera basın.

Bu komutla birlikte modüller derlenecek ve /usr/lib/kernelsürümü dizinine kurulacaktır.

Tebrikler sıfırdan bir çekirdek derlediniz.

10 LILO'da başlangıç konfigürasyonu

Artık LILO'yu yeni çekirdeği yükleyecek şekilde konfigüre etmenin zamanı geldi. LILO en çok bilinen ve kullanılan linux başlangıç yükleyicisidir, bütün popüler Linux dağıtımlarında da bulunur.

İsterseniz öncelikle /etc/lilo.conf dosyasına bakarak başlayalım. Bu dosyada bir yerde image=/vmlinuz gibi bir satır olacaktır. Bu satır LILO'ya çekirdek için nereye bakması gerektiğini belirtir.

10.1 LILO'yu konfigüre etmek

LILO'yu yeni çekirdeği başlatacak şekilde konfigüre etmek için iki seçeneğiniz var. İlki mevcut çekirdek dosyasının üstüne yenisini yazmak –ki çok tavsiye edilmez, herşeyin yolunda gideceğinden bu kadar emin misiniz ? Daha güvenilir seçenek ise LILO'yu istenirse eski çekirdekle sistemi açabilecek şekilde konfigüre etmek. Şimdi bu yöntemi nasıl uygulayacağımızı göreceğiz.

10.2 LILO kodu

/etc/lilo.conf dosyanız aşağıdaki gibi olabilir

```
boot=/dev/hda
delay=20
vga=normal
root=/dev/hda1
read-only
image=/vmlinuz
label=linux
```

lilo.conf'a yeni bir başlangıç seçeneği eklemek için şunları yapın. Önce /usr/src/linux/arch/i386/-boot/bzImage dosyasını /vmlinuz2 gibi bir isimle root'a (/) kopyalayın.

Ardından lilo.conf dosyanızın son üç satırını kopyalayıp tekrar yapıştırın. Nerdeyse bitti :)

Şimdi lilo.conf dosyanızın aşağıdaki gibi gözükmesi lazım

```
boot=/dev/hda
delay=20
vga=normal
root=/dev/hda1
read-only
image=/vmlinuz
label=linux
image=/vmlinuz
label=linux
```

Önce ilk "image=" satırını "image=/vmlinuz2" şeklinde değiştirin ve ikinci "label=" satırını da "label=oldlinux" şeklinde değiştirmeyi unutmayın. Tabi eğer dosyanın yukarlarında bir yerlerde "delay=20" gibi bir satır yoksa böyle bir satır da eklemeniz gerekecek. Eğer böyle bir satır varsa bekleme süresini minimum 20 yapın.

Sonuç olarak nihai lilo.conf dosyanız aşağıdaki gibi gözükecek;

```
boot=/dev/hda
delay=20
vga=normal
root=/dev/hda1
read-only
image=/vmlinuz2
label=linux
image=/vmlinuz
label=oldlinux
```

Bütün bu değişiklileri yapıp text editörünüzden çıktıktan sonra lilo komutunu verip diskinizdeki konfigürasyonun güncellenmesini sağlayın. Eğer bunu yapmazsanız yaptığınız değişiklikler bir işe yaramayacaktır. Ayrıca bunu yapınca boot map dosyası da güncellenecektir.

10.3 LILO konfigürasyonunun nedenleri ve niçinleri

Gelelim yaptığımız değişikliklerin açıklanmasına. Bu lilo.conf dosyası iki ayrı çekirdekle açılış yapmak üzere düznelendi. Orjinal çekirdeğiniz /vmlinuz ile açılış yapabileceğiniz gibi yeni derlediğiniz /vmlinuz2 çekirdeği ile de açılış yapabilirsiniz. Otomatik olarak yeni çekirdek seçilecektir, çünkü onun image/label satırı dha yukarda tanımlanmıştır. Eğer eski çekirdeği boot etmek isterseniz açılış sırasında shift tuşuna basın. Bu size istediğini etiketi girip kernel seçme şansını verecektir. Burada oldlinux yazıp boot edebilir, ya da tab'a basıp seçeneklerinizi gözden geçirebilirsiniz.

11 PCI aygıtları

Bu bölüm PCI aygıtları ile çalışmanın ince detaylarından bahsedecektir. Linux altında PCI aygıt desteğini sağlamak basitçe "General Setup" bölümü altında "PCI Support" seçeneğini seçmekle olur. Bunun yanında "PCI device name database" seçeneğini seçmeniz de tavsiye edilir. Bu sayede PCI aygıtlarının aygıt numarası ve sayısal bilgileri yerine gerçek ingilizce isimlerini görebilirsiniz.

11.1 Mevcut PCI aygıtlarını incelemek

Bilgisayarınıza takılı olan PCI aygıtlarını incelemek için cat /proc/pci yazabilirsiniz. Çıkan bilginin çok işinize yarayacağı, hatta anlaşılabilir olacağı bile kesin değil :) Bunun yerine lspci -v yazarsanız çok daha anlaşılabilir bir çıktı ile karşılaşırsınız. lspci komutu pciutils paketinin bir parçasıdır bu programlara http://atrey.karlin.mff.cuni.cz/ mj/pciutils.html adresinden ulaşabilirsiniz. Genellikle de hemen her dağıtımda bulunurlar.

lspci -v komutu ile bilgisayarınızda bulunduğunu bile bilmediğiniz donanımlarla karşılaşabilirsiniz. Genellikle bunlar bilgisayarınızın ana kartının üstünde bulunan pci tabanlı çevrebirimleridir. Bunlar genellikle BIOS vasıtası ile aktif ya da pasif hale getirilebilirler. Pciutils paketi aynı zamanda setpci komutu ile çeşitli PCI donanımlarının zamanlamasına müdahale edebilecek bir program sağlar. PCI donanım zamanlamaları ve etkileri konusunda IBM Developerworks'ün "Linux Hardware Stability Guide" makalesinin ikinci bölümünü okuyabilirsiniz.

11.2 PCI donanım kaynakları

Görevlerini yerine getirebilmek için sisteminizdeki PCI aygıtları değişik donanım kaynaklarınızdan faydalanırlar, kesmeler bunlara bir örnektir. Birçok PCI aygıtı donanımsal kesmeleri (interruptlar) işlenecek verileri olduğunu sisteme haber vermek için kullanırlar. Çeşitli donanım aygıtlarınız için hangi kesmelerin kullanıldığını görmek için /proc/interrupts dosyasına bakmanız yeterlidir. Bunun için cat /proc/-interrupts komutunu vermeniz gerekir. Çıktısı şu şekilde olacaktır.

```
> cat /proc/interrupts
 CPU0
  0:    1848603 XT-PIC timer
  1:      97489 XT-PIC keyboard
  2:          0 XT-PIC cascade
  5:      12148 XT-PIC eth0
  9:    1397985 XT-PIC nvidia
 10:          1 XT-PIC soundblaster
 11:       9921 XT-PIC bttv
 12:     540231 XT-PIC PS/2 Mouse
 14:      25550 XT-PIC ide0
 15:          7 XT-PIC ide1
NMI:          0
LOC:    1848535
ERR:          0
MIS:          0
>
```

İlk kolon bir IRQ numarası verir, ikinci kolon ise kernel tarafından o kesmeden kaç tane üretildiğini gösterir. Son kolon ise sözkonusu donanımın kısa ismini içerir.

Donanımlarınızın kullanmakta olduğu IO portlarının listesini de şöyle görebilirsiniz

```
> cat /proc/ioports
0000-001f : dma1
0020-003f : pic1
0040-005f : timer
0060-006f : keyboard
0080-008f : dma page reg
00a0-00bf : pic2
00c0-00df : dma2
00f0-00ff : fpu
0170-0177 : ide1
01f0-01f7 : ide0
```

```
0213-0213 : isapnp read
0220-022f : soundblaster
02f8-02ff : serial(set)
0300-0303 : MPU-401 UART
0376-0376 : ide1
03c0-03df : vga+
03e8-03ef : serial(set)
03f6-03f6 : ide0
03f8-03ff : serial(auto)
0a79-0a79 : isapnp write
0cf8-0cff : PCI conf1
4000-403f : Intel Corp. 82371AB PIIX4 ACPI
5000-501f : Intel Corp. 82371AB PIIX4 ACPI
e000-e01f : Intel Corp. 82371AB PIIX4 USB
e400-e43f : 3Com Corporation 3c905 100BaseTX [Boomerang]
e400-e43f : 00:0b.0
f000-f00f : Intel Corp. 82371AB PIIX4 IDE
f000-f007 : ide0
f008-f00f : ide1
```

12 Linux USB

Çekirdeği konfigüre ederken büyük ihtiamlle "USB support" bölümü dikkatinizi çekmiştir. Burada Universal Serial Bus (Evrensel Seri Veriyolu) denilen USB'nin özellikleri bulunmaktadır. Nispeten yeni bir veriyolu olan USB özellikle çevrebirim alanında gittikçe yaygınlaşmaktadır, şimdiden USB klavye, fare, modem, printer, scanner hatta depolama cihazları çıkmıştır ve gün geçtikçe de daha yaygın kullanılmaktadırlar.

Linux üzerinde USB desteği de sadece bir kaç yıllık olduğundan, birçok Linux kullanıcısı şimdiye kadar pek USB cihaz kullanmamış ya da Linux altında USB desteğinin tam nasıl çalıştığını bilmiyor olabilir.

12.1 USB'yi etkin hale getirmek

Linux USB desteğini sağlamak için önce "USB Support" bölümüne gidip "Support for USB" seçeneğini seçin. Bundan sonra yapmanız gereken uygun USB sisteminizi seçmek. Seçenekleriniz "EHCI", "UHCI", "UHCI(alternate driver)" ve "OHCI". Linux kullananların problem yaşadığı temel nokta burasıdır.

UHCI, OHCI, EHCI eee, hani bunun kolay olması gerekiyordu ;)

EHCI ve arkadaşlarının ne olduğunu anlamak için öncelikle USB kullanacak her anakartın belli bir USB sistemi olduğunu anlamak gerekir. İşte bu sistemlerin isimleri EHCI, UHCI ve OHCI'dir.

EHCI sürücüsü, yeni model hızlı USB 2.0 sistemlerde yeralır.

OHCI sürücüsü ise, PC olmayan sistemleri de desteklemek amacıyla geliştirilmiştir, SIS ve ALI chipsetlerini de destekler.

UHCI sürücüsü ise basitçe geri kalan cihazlar içindir

Tedbirli davranmak için EHCI ve UHCI'den birini seçip (çok da farketmez) OHCI desteğini de çekirdeğe eklemekte fayda vardır.

12.2 Son birkaç adım

USB desteğini etkin hale getirip gerekli HCI sürücüsünü sağladıktan sonra yapılması gereken bir iki küçük ayar daha var. "Preliminary USB device filesystem"i etkin hale getirmeniz ve Linux ile kullanacağınız herhangibir USB sürücüsünü sağlamanız ger ekir. Mesela USB oyun kontrolerini etkin hale getirmek için "USB Human Interface Device (full HID) support"u seçmeniz gerekebilir.

12.3 usbdevfs'yi bağlamak (mount)

USB desteği olan yeni çekirdeğinizi boot ettikten sonra USB aygıt dosya sistemini /proc/bus/usb altında bağlamanız gerekir

```
mount -t usbdevfs none /proc/bus/usb
```

Sistem boot ettiğinde otomatik olarak USB sistemi bağlayabilmek içinse fstabın içine /proc satırından sonra

```
none /proc/bus/usb usbdevfs defaults 0 0
```

satırını eklemek gerekir.

13 Diff, Patch

Unix'de herşeyin bir dosya olarak ele alındığını ve bu şekilde işlem gördüğünü daha önce anlatmıştık. Bu sebeple özellikle text işleme komutları oldukça gelişmiş durumdadır. Bir dosya içinden istenen kelimeleri ayıklama, onları sıralama vs. gibi işlemleri zaten daha önceki derselerde yapmıştık. Bir başka önemli işlem de bir dosyaya yapılan değişiklikleri bir değişiklikl dosyasında toplayıp, önceki dosya ve değişiklileri kulanarak dosyanın son haline ulaşmak.

Bu iş için kullandığımız komutlar diff ve patch komutları.Şimdi bunları teker teker ele alıyoruz.

13.1 Diff

Önce bir dosya yarÖrnek olarak 2.4.18 kernelimize preemptive patchini uygulayacağız.atalım. Bu dosya bizim evimizde vereceğimiz bir partinin isim listesi olsun.

```
birinci.txt
```

ve bunun içinde şunları yazalım

```
Ali
Berktan
Ebru
Tanya
Osman
Mehmet
Kemal
```

Ve kaydedelim. Şimdi ise aynı dosyayı değiştirip, partiye katılamayacaklarını belirtmiş olan Osman, Mehmet ve ve Ebru'yu çıkartalım. Yerine Mine ve Ela'yı koyalılım, partimizde daha çok kız olsun ;) Dosyanın son hali

```
Ali
Berktan
Tanya
Kemal
Ela
Mine
```

olacak. Bu halini de son.txt diye kaydedelim ve şu komutu çalıştıralım.

```
diff birinci.txt son.txt
```

```
3d2
< Ebru
5,6d3
< Osman
< Mehmet
7a5,6
> Ela
> Mine
```

Belki ilk bakışta çok anlamlı gelmeyebilir bu çıtı ama aslında çok da anlamlı ed komutlarıdır bunlar. Açıklamak gerekirse şu anlamlara gelmektedirler

birinci.txt dosyasının üçüncü satırı ikinci dosyada yok onu sil birinci.txt dosyasının beşinci ve altıncı satırları olan Osman ve Mehmet ikinci dosyada yok onları sil ikinci dosyanın beşinci ve altıncı satırları ilk dosyada yok, onları yedinci satırdan itibaren ekle (Kemal'in ardına)

şimdi çok popüler birisi olduğunuzu ve bu listenin binlerce kişi uzunluğunda olduğunu düşünün. Ya ikinci bir listeyi de herkese göndereceksiniz, yada bu birkaç satırlık diff dosyası ile bu işi kolayca halledeceksiniz.

Tabi bunun için sözkonusu değişiklikleri bir dosya haline getirmemiz gerekmekte, bu da ilk derste öğrendiğimiz yönlendirme komutları ile gerçekleşmekte:

```
$ diff birinci.txt son.txt > degisiklik.txt
```

Artık elimizde iki dosya var birinci.txt ve degisiklik.txt dosyası.
şimdi yapmamız gereken

```
$ patch birinci.txt degisiklik.txt
patching file birinci.txt
```

Herşey yolunda gittiyse şu an birinci.txt dosyası ve son.txt dosyalarının aynı olması lazım. Bakalım aynı mıymışlar ?

```
$ diff birinci.txt son.txt
```

Herhalde diff gibi bir komutumuz varken, binlerce satırlık iki dosyanın aynı olup olmadığını gözümüzle kontrol edecek değildik :) Gördüğünüz gibi diff komutu hiçbir çıktı vermemiş, dolayısıyla bu iki dosyanın aynı olduğunu onaylamıştır.

Basitçe iki dosya için bu şekilde kullanılan patch ve diff komutları yüzlerce dosya için de kullanılabilir.

Örnek olarak 2.4.18 kernelimize preemptive patchini uygulayacağız.

14 Çekirdeği yamamak (patch)

Linux çekirdeği, belli bir ekip tarafından geliştirilmekte olan bir yazılım projesidir. Bu ekibin dahilinde ve haricinde birçok kişi Linux çekirdek kodu ile ilgilenmektedir. Bu doğrultuda insanlar zaman zaman kendileri için geliştirdikleri özelliklere çekirdeğe eklemekte ve bundan diğer kişileri haberdar etmektedirler. Eğer bu değişiklikler, yeteri kadar faydalı iseler bir süre sonra zaten çekirdeğin bir parçası halini alırlar.

Diyelim ki bu türden bir çekirdek yaması biliyorsunuz, (örneğimiz de Linux'un daha etkin çalışmasını sağlayan pre-emptive yaması olsun) ve bunun dahil olduğu bir çekirdek derlemek istiyorsunuz. Normalde make xconfig çalıştırdığımızda Processor Type and Features altında bulunması gereken Preemptible Kernel seçeneği göreceğiniz gibi standart çekridek kodunda bulunmamakta.

Bunun için yapmanız gereken sözkonusu yama dosyasını temin etmek ve bunu mevcut çekirdek kodunuzun bulunduğu /usr/src/linux altına kopyalamak. Bunun ardından ise o dizinin altında şu komutu veriyorsunuz

```
patch -p1 < preempt-kernel-rml-2.4.18-5.patch
patching file CREDITS
patching file Documentation/Configure.help
patching file Documentation/preempt-locking.txt
patching file MAINTAINERS
patching file arch/arm/config.in
patching file arch/arm/kernel/entry-armv.S
patching file arch/arm/tools/getconstants.c
patching file arch/i386/config.in
patching file arch/i386/kernel/entry.S
patching file arch/i386/kernel/i387.c
patching file arch/i386/kernel/smp.c
patching file arch/i386/kernel/traps.c
patching file arch/i386/lib/dec_and_lock.c
patching file arch/sh/config.in
patching file arch/sh/kernel/entry.S
patching file arch/sh/kernel/irq.c
patching file drivers/ieee1394/csr.c
patching file drivers/sound/sound_core.c
patching file fs/adfs/map.c
patching file fs/exec.c
patching file fs/fat/cache.c
patching file fs/nls/nls_base.c
patching file include/asm-arm/dma.h
patching file include/asm-arm/hardirq.h
patching file include/asm-arm/pgalloc.h
patching file include/asm-arm/smplock.h
patching file include/asm-arm/softirq.h
patching file include/asm-i386/hardirq.h
```

```
patching file include/asm-i386/highmem.h
patching file include/asm-i386/hw_irq.h
patching file include/asm-i386/i387.h
patching file include/asm-i386/pgalloc.h
patching file include/asm-i386/smplock.h
patching file include/asm-i386/softirq.h
patching file include/asm-i386/spinlock.h
patching file include/asm-sh/hardirq.h
patching file include/asm-sh/smplock.h
patching file include/asm-sh/softirq.h
patching file include/linux/brlock.h
patching file include/linux/dcache.h
patching file include/linux/fs_struct.h
patching file include/linux/sched.h
patching file include/linux/smp.h
patching file include/linux/smp_lock.h
patching file include/linux/spinlock.h
patching file include/linux/tqueue.h
patching file kernel/exit.c
patching file kernel/fork.c
patching file kernel/ksyms.c
patching file kernel/sched.c
patching file lib/dec_and_lock.c
patching file mm/slab.c
patching file net/core/dev.c
patching file net/core/skbuff.c
patching file net/socket.c
patching file net/sunrpc/pmap_clnt.c
```

Gördüğünüz gibi patch komutu, sözkonusu dosyayı alıp, o dosya doğrultusunda gerekli yerlerde gerekli değişiklikleri yaptı.

Şimdi tekrar make xconfig yapıyoruz ve görüyoruz ki biraz önce olmayan Preemptible Kernel seçeneği artık var.

Bu şekilde bir yada daha fazla yama ile çekirdeğimizi yamayabilir istediğimiz gibi yeni özellikler ekleyebiliriz. Burada unutulmaması gereken yamanın sürümü ile çekirdek kodunun sürümünün aynı olması gerektiği, aksi takdirde problemlerle karşılaşma ihtimalimiz yüksek.

15 Kaynaklar

Linux Kernel How-to kernel kurulum bilgilerini edinmek için iyi bir kaynaktır. LILO, Linux Crash Rescue How-TO dosyaları da sizlere yardımcı olabilir.

Linux Kernel'leri www.kernel.org adresinde bulunmaktadır

www.linuxdoc.org adresinden Linux konulu hemen her türlü bilgiye ulaşabilirsiniz. Burada "Guides" başlığı altındaki belgeler özellikle oldukça faydalı olacaktır. Eric S. Raymond'un Unix ve Internet Fundamentals How-To belgesi de bilgilendirici olabilir.

Türkçe içerik olarak da yukarda sayılan bir çok belgenin türkçelerini belgeler.org adresinde bulabileceğiniz gibi, ileriseviye.org ve fazlamesai.net gibi kaynaklardan da faydalanabilirsiniz.

16 TCP/IP Ağ İşlemlerine Giriş

Birkaç GNU/Linux yüklü makinadan oluşan Ethernet tabanlı basit bir YAA (Yerel Alan Ağı - LAN [Local Area Network]) kurmak çok sık karşılaşılan göreli olarak basit bir iştir. Genellikle bilmeniz gereken tek şey Linux yüklü bilgisayarlarınızda bir tür Ethernet ağ kartı bulunup bulunmadığıdır. Daha sonra merkezi bir Ethernet hub ya da switch cihazı ve uygun kablolama ile makinalar birbirine fiziksel olarak bağlanır. Eğer tüm makinaların çekirdeğinde ilgili Ethernet kartları ve TCP/IP desteği aktif halde ise artık makinalarınız yerel ağ üzerinden haberleşmek için her şeye sahip demektir.

Ancak tek başına Ethernet konusunun pek eğlenceli olduğu söylenemez. Bu aşamada tüm donanımınız ve çekirdeğiniz yerel ağ üzerinden iletişim kurmaya hazır olmakla birlikte bu durum tek başına yeterli değildir. Linux uygulamalarının büyük çoğunluğu ağ üzerinde ham Ethernet paketlerini (ya da çerçevelerini) kullanarak veri değiştokuşu yapmaz. Bunun yerine, uygulamaların büyük bir kısmı TCP/IP

protokolü olarak isimlendirilen ve daha yüksek ve soyut seviyeli bir protokol kullanır. Mutlaka TCP/IP sözünü daha önce duymuşsunuzdur. Bu bir protokol süitidir ve bünyesindeki protokoller bugün Internet diye bildiğimiz devasa uluslararası ağ üzerindeki iletişimin temel kurallarını belirler (TCP/IP'nin açılımı: Trasmission Control Protocol / Internet Protocol).

16.1 Çözüm: Ethernet üzerinden TCP/IP

Yapılması gereken Ethernet uyumlu ağınızdaki bilgisayarları TCP/IP protokolü ile iletişim kurabilecekleri şekilde ayarlamaktır. Bunun nasıl çalıştığını anlamak için önce Ethernet'i biraz anlamalıyız. Ethernet ile birbirlerine bağlı bilgisayarlardan oluşan bir ağda, her makinanın üzerindeki Ethernet kartının kendine özgü ve diğerlerinden farklı bir adresi vardır. Kartın fiziksel olarak içine üretilirken gömülen bu adres şuna benzer:

```
00:01:02:CB:57:3C
```

Ethernet sistemi CSMA/CD (Carrier Sense and Multiple Access with Collision Detection - Taşıyıcı Algılama ve Çarpışma Tespitli Çoklu Erişim) isimli bir yöntem kullanır. CSMA/CD sistemi kısaca şu demektir: tüm cihazlar tek bir fiziksel ortam üzerinden haberleşirler, belirli bir anda sadece tek bir cihaz yayın yapabilir ve tüm cihazlar eşzamanlı (simultane) olarak yapılan yayını dinleyebilirler. Eğer 2 farklı cihaz aynı anda yayın yapmaya kalkarsa iletişim çarpışması oluşur, bu tespit edilir ve her iki cihaz da çok kısa bir süre beklerler (bu süre belli bir aralıkta gelişigüzel tayin edilir) ve tekrar yayın yapmaya çalışırlar. Böylece tek bir seferde tek bir cihazın belli bir uzunluktaki mesajını ağ ortamındaki bilgisayarlara iletmesi pratik olarak sağlanır, bir cihazın ürettiği ve ağ ortamında yayınladığı mesajın kime ait olduğu da mesaja eklenen ve yukarıda bahsi geçen Ethernet adresi sayesinde belirlenmiş olur.

16.2 IP adreslerine giriş

Bu donanım adresleri yerel ağdaki makinaları birbirlerinden ayırt etmeek için kullanılır. Donanım adresini kullanan bir makina bir başka makinaya bir Ethernet paketi yollayabilir. Ancak TCP/IP tabanlı iletişim yöntemleri farklı bir adresleme şekli kullanır ve bunlara IP adresleri denir. IP adreslerinin görüntüsü şuna benzer:

```
192.168.1.1
```

Her bir çevresel ağ cihazı için, çekirdekte, ilgili bir arabirimin varolması gerekmektedir. Örneğin Linux'da, ethernet arabirimleri, eth0 ve eth1 şeklinde isimlendirilmişlerdir. PPP arabirimleri ppp0 ve ppp1 olarak ve FDDI arabirimleri ise fddi0 ve fddi1 şeklindeisimlendirilmişlerdir. Bu arabirim isimleri, bir ayar komutunda fiziksel bir cihazı belirtmek istediğiniz zaman kullanılmakta ve bunun dışında bir kullanım alanları bulunmamaktadır.

TCP/IP ağı tarafından kullanılmadan önce, her bir arabirime dünyanın geri kalanıyla haberleşildiğinde kimlik numarasıymış gibi kabul edilen, bir IP adresi atanmalıdır. Bu adres, arabirim isminden farklıdır, eğer bir arabirimi kapı olarak düşünürseniz, adresi de o kapının üzerine yapıştırılmış isim plakası (ya da kapı numarası) olarak düşünebilirsiniz.

Donanımın bir kısmı tarafından ki bu donanım parçası En Büyük Aktarım Birimi (MTU - Maximum Transfer Unit) olarak bilinir, işlenen datagramların boyutlarını kısıtlamak gibi bir ayarın yapılması da mümkündür.

IP ağ protokolu, adresleri, 32 bitlik numaralar olarak görür. Her bir makineye, bulunduğu ağ ortamında tek olan bir numara atanmak zorundadır.Internet Protokolünün genelde 4. sürümü kullanılmaktadır. Yerini almak üzere tasarlanmış olan bir de 6. sürümü vardır. IPV6 değişik bir adresleme şeması kullanır ve daha büyük bir adresleme alanına sahiptir. Linux'da IPv6 desteği bulunmaktadır, ancak henüz bu kitabın kapsamına alınmamıştır. Linux çekirdeğindeki IPv6 desteği iyidir, ancak ağ ortamında kullanılmasını sağlayacak uygulamaların IPv6'ya göre değiştirilmesi gerekmektedir.

Eğer sadece, başka bir dış ağ ile ilişkisi bulunmayan, yerel bir ağda çalısıyorsanız, bu numaraları kendi istediğiniz şekilde atayabilirsiniz. Bazı IP adres aralıkları, belirli tipte özel ağlara ayrılmıştır. Ancak Internet üzerindeki siteler için, bu numaralar, NIC (Network Information Center - Ağ Bilgi Merkezi) olarak bilinen, merkezi bir otorite tarafından atanmaktadır. Çoğunlukla, IP adresleriniz sizin yerinize IP adresinizi alan bir sağlayıcı tarafından atanır. Bunun yerine, ağınız için kullanacağınız IP adresini doğrudan NIC'den hostmaster@internic.net adresine yazarak bir eposta ile ya da http://www.internic.net/ adresindeki formu doldurarak alabilirsiniz.

IP adresleri, okunulabilirliği arttırmak amacıyla, öktet denilen, 4 adet 8 bitlik (1 baytlık - 0 dan 255 e kadar) numaraya ayrılmıştır. Örneğin quark.physics.groucho.edu adresindeki makinenin (Grouche Üniversitesinde, fizik bölümündeki quark isimli makinenin) IP adresi 0x954C0C04 (onaltılık sistemde) olsun.

Bu adres 149.76.12.4 olarak farklı bir gösterimde de yazılabilir. Bu ikinci gösterim, genelde noktalı dörtlü gösterim olarak bilinir.

Bu tip gösterimin başka bir nedeni ise, IP adreslerinin ilk öktetlerinin ağ numarasını, geri kalanların ise makine numarasını göstermeleridir. NIC'e IP adresi almak için başvurduğunuzda kullanmayı düşündüğünüz her makine için size bir IP verilmez, onun yerine size bir ağ numarası atanır ve bu aralıkdaki tüm geçerli IP'leri tercihinize göre ağınızdaki makinelere atamanıza izin verilir.

Makine kısmının sayısı, ağın büyüklüğüne bağlıdır. Farklı ihtiyaçları karşılamak üzere, IP adreslerinde farklı noktalardan ayrılan, farklı ağ sınıfları tanımlanmıştır. Bu sınıflar şu şekilde tanımlanmışlardır:

A Sınıfı: A sınıfı ağlar 1.0.0.0'dan 127.0.0.0'a kadar olan ağları kapsar. Sadece ilk oktet, ağ numarasını belirler, geri kalanlar ise her bir ağ içinde kullanılabilecek (atanabilecek) makine sayısını belirler ki bu sınıfta, dolayısıyla, 24 bit makine kısmına ayrılmıştır. Bu da kabaca, ağ başına 1.6 milyon makine anlamına gelmektedir.

B Sınıfı: B sınıfı ağlar, 128.0.0.0'dan 191.255.0.0'a kadar olan ağları içermektedir. Ağ numarası, ilk iki oktet ile belirlenir. Bu sınıf ile 65,024 adet makineden oluşan 16,320 adet ağ tanımlanabilir.

C Sınıfı: C sınıfı ağlar ise 192.0.0.0 ile 223.255.255.0 arasını kapsar. İlk 3 oktet ağ numarasını belirler ve geri kalanlar da her bir ağ içinde bulunabilecek makine sayısını belirler. Dolayısıyla her biri 254 makineli yaklaşık 2 milyon ağ tanımlanabilir.

D, E, ve F Sınfları: 224.0.0.0 ile 254.0.0.0 arasında kalan adresler, ya deneysel amaçlıdırlar ya da belirli amaçlar için ayrılmışlardır ve herhangi bir ağ tanımlamazlar. Bir internet üzerinde, paketlerin, bir seferde bir çok noktaya iletilmesi hizmetini veren IP çokluyayını (IP Multicast) için bu aralıktaki adresler atanır.

Yukarıdaki 149.76.12.4 IP numaralı quark'ın adresi, B sınıfı olan 149.76.0.0 numaralı ağdaki 12.4 numaralı makineyi belirtir.

Bir önceki listede, tüm olası numaraların kapsanmadığını fark edebilirsiniz. Bunun nedeni 0 ve 255 numaralı öktetlerin özel amaçlar için ayrılmış olmasındandır. Bir adreste, host (konak) bilgisayar tarafındaki bitlerin tamamı 0 ise o adres ağı belirler, eğer bu bitlerin tamamı 1 ise bu adres yayın (broadcast, ağın tümüne yayın) adresi olarak bilinir. Buna göre 149.76.255.255 geçerli bir makine adresi belirtmez onun yerine 149.76.0.0 ağındaki tüm makineleri belirtir.

0.0.0.0 ve 127.0.0.0 ağı ise sizin kendi makinenizdeki IP trafiğini rahatlatmak amaçlı, sizin kendi makinenizi ifade eder. Birincisine öntanımlı rota (default route) diğerine de geridönüş (loopback) adresi denir.

Yani bilgisayarınız içinde kalan, yerel ağa çıkmanızı gerektirmeyen durumlarda 127.0.0.1 IP adresini kullanarak yine kendi bilgiyarınız içinde, programların haberleşmelerinisağlayabilirsiniz. Genelde, 127.0.0.1 adresi, kısa devreymiş gibi davranan, geridönüş arabirimi (loopback interface) olarak adlandırılan ve sizin kendi makinenizde bulunan özel bir arabirime ayrılmıştır.

16.3 IP adresleri ile Ethernet arayüzünü eşleştirmek

Ethernet tabanlı yerel ağın (LAN) TCP/IP ile çalışabilmesi için her makinanın Ethernet kart adresine (donanım adresine) bir IP adresi karşılık getirmeniz, başka bir deyişle bunları birbirleri ile eşleştirmeniz gerekir. Bunu Linux ortamında yapmanın kolay bir yolu vardır.

Aslında eğer halihazırda Linux ortamında Ethernet kullanıyorsanız mevcut Linux dağıtımınızın başlangıç betiklerinde şuna benzer bir komut vardır:

```
ifconfig eth0 192.168.1.1 broadcast 192.168.1.255 netmask 255.255.255.0
```

Yukarıdaki ifconfig komutu eth0'ı (dolayısı ile eth0'ın donanım adresini) 192.168.1.1 numaralı IP adresi ile eşleştirir. Bu işlevin yanı sıra aynı komut satırı IP ile ilgili diğer bilgileri de sisteme söyler: broadcast adresi (192.168.1.255) ve ağ maskesi (netmask) (255.255.255.0). Bu komutu çalıştırıldığında eth0 arayüzünüz aktive olmuştur ve bir IP adresine sahiptir.

16.4 ifconfig -a kullanımı

Halihazırda çalışmakta olan tüm ağ cihazlarınızı ifconfig -a komutunun çıktısında görebilirsiniz:

```
eth0
```

```
Link encap:Ethernet inet addr:192.168.1.1

HWaddr 00:01:02:CB:57:3C

Bcast:192.168.1.255

UP BROADCAST RUNNING MULTICAST MTU:1500


Mask:255.255.255.0 Metric:1

RX packets:0 errors:0 dropped:0 overruns:0 frame:0 TX packets:0 errors:0
dropped:0 overruns:0 carrier:0 collisions:0 txqueuelen:100 RX
bytes:0 (0.0 b) TX bytes:0 (0.0 b) Interrupt:5 Base address:0xc400
Link encap:Local Loopb
ack inet addr:127.0.0.1 Mask:255.0.0.0 UP LOOPBACK RUNNING MTU:16436 Metric:1
RX packets:1065 errors:0 dropped:0 overruns:0 frame:0 TX packets:1065 errors:0
dropped:0 overruns:0 carrier:0 collisions:0 txqueuelen:0 RX bytes:2
62542 (256.3 Kb) TX bytes:262542 (256.3 Kb)

lo
```

Yukarıda ayarları yapılmış bir eth0 arayüzünün yanısıra bir lo (localhost) arayüzü görmektesiniz. lo isimli arayüz özel bir sanal ağ arayüzüdür ve herhangi bir ağ bağlantısı olmaması halinde de TCP/IP uygulamalarının bilgisayarda yerel olarak çalışmalarını sağlamak için geliştirilmiştir.

16.5 TCP/IP çalışıyor!

Bir kez tüm ağ arayüzleriniz aktive edilip IP adresleri ile eşleştirildi mi Ethernet tabanlı ağınız artık TCP/IP trafiğini taşımak için kullanılabilir. Sisteminizdeki bilgisayarlar birbirlerine IP adreslerini kullanarak erişebilirler ve bilgisayarlarınıza ping, telnet, ssh gibi komutlarla erişebilirsiniz.

16.6 İsim çözümleme sınırları

Şu aşamada her ne kadar bilgisayarlarınıza ping 192.168.1.1 gibi komutlarla erişseniz de onlara tanıdık isimlerle hitap edemezsiniz. Söz gelimi ping mybox yazarsanız çalışmaz. Bunun için yapmanız gereken her Linux yüklü bilgisayarda /etc/hosts isimli bir dosya oluşturmak ve bu dosyanın içinde IP adreslerine karşılık istediğiniz isimleri yazmaktır.

Yani mesela üç bilgisayardan oluşan bir ağ var ise /etc/hosts dosyası şunun gibi olacaktır:

```
127.0.0.1     localhost
192.168.1.1   mybox mybox.gentoo.org
192.168.1.2   testbox testbox.gentoo.org
192.168.1.3   mailbox mailbox.gentoo.org
```

Söz konusu /etc/hosts dosyasında zorunlu olarak "localhost" 127.0.0.1 eşleştirmesi olduğuna dikkat edin. Ayrıca yerel ağdaki tüm bilgisayarların isimleri (hostname) kısa ve uzun formatta tanımlanmıştır. Bu /etc/hosts dosyasını diğer bilgisayarlara da kopyaladıktan sonra artık bilgisayara IP adresleri yerine tanımladığımız isimlerle hitap edebiliriz, mesela artık ping mybox gibi bir komutu çalıştırabiliriz!

16.7 DNS kullanımı

Küçük çaplı yerel ağlarda yukarıdaki yöntem pratik gibi görünebilir ama bilgisayar sayısı arttıkça isim-adres eşleştirmesinin bu şekilde yapılmas çok baş ağrıtır. Bu tür durumlarda "IP-makina ismi" eşleştirmesini tek bir bilgisayarda depolayıp bir "DNS sunucusu" çalıştırmak çok daha pratik olacaktır (DNS - domain name service server - alan adı servisi sunucusu). Böylece her makinayı öyle bir ayarlayabiliriz ki bir isme karşılık hangi IP adresinin karşılık geldiğini tek bir merkezi makinaya sorup öğrenebilir. Bunu yapmanın yolu her makinada /etc/resolv.conf isimli bir dosya oluşturmaktır. Bu dosyanın içeriği şunun gibidir:

```
domain gentoo.org nameserver 192.168.1.1 nameserver 192.168.1.2
```

Yukarıdaki /etc/resolv.conf ayar dosyasına göre uzun formatta verilmeyen her isim (testbox.gentoo.org yerine testbox demek gibi) yerel ağdaki bir makinanın ismi olarak kabul edilmelidir. Yine yukarıdaki dosyadaki bilgiler 192.168.1.1 adresli bir makinada bir DNS sunucunun çalıştığını ve 192.168.1.2 adresli makinada da yedek DNS sunucusunun beklemekte olduğunu söylemektedir. Aslında ağa bağlı hemen hemen tüm Linux PC'ler resolv.conf dosyasında öntanımlı bir isim sunucu içerirler (ağa aktif olarak bağlı olmasalar bile). Bunun sebebi ISS (Internet Servis Sağlayıcı) şirketinin sunduğu DNS sunucuyu kullanmak durumunda olmalarıdır.

Böylece kullanıcılar Web tarayıcılarını açıp, bir IP adresi yerine ibm.com gibi akılda kolay kalan bir isim yazıp web sitesine bağlanabilirler.

16.8 Dışarıya bağlanmak

Madem az önce Internet'e bağlanmaktan bahsettik 3 sistemden oluşan basit ağımızı 'dışarıya nasıl bağlayabiliriz' konusuna gelelim. Tipik olarak bir tür 'router' satın alabilir ve bunun bir ucunu bizim Ethernet tabanlı yerel ağımıza diğer ucunu da DSL, kablo modem, T1 ya da normal telefon hattına bağlayabiliriz. Bu router'a belli bir IP adresi verilir ve böylece bu cihaz ağımızdaki sistemlerle iletişim kurabilir. Daha sonra ağımızdaki her bilgisayarı öntanımlı olarak bu router'ı Internete çıkış kapısı (gateway) kabul edecek şekilde düzenleyebiliriz. Yani sistemdeki makinalardan birinin adresinin içermeyen her veri paketi router'a yönlendirilir ve sonra ağın dışına çıkması gereken bu paketin yönlendirilmesi ile artık router ilgilenir. Genellikle kullandığınız dağıtımın başlangıç betikleri (initialization scripts) sizin için öntanımlı olarak router ayarlarını yaparlar. Bunu gerçekleştirmek için sisteminiz çalıştırılırken aşağıdakine benzer bir komut kullanılır:

```
route add -net default gw 192.168.1.80 netmask 0.0.0.0 metric 1
```

Yukarıdaki komuta göre öntanımlı router adresi 192.168.1.80'dir. Sisteminizde ayarları yapılmış tüm routerlarla ilgili bilgileri edinmek için route -n komutunu verebilirsiniz. Hedef adresi olan "0.0.0.0" yol öntanımlı yoldur.

16.9 Ev ödevi

Şu ana dek Linux ağ işlemlerine dair en temel kavramlardan bahsettik. Maalesef uygun IP adresini seçmek, ağ maskeleri, broadcast adresleri, vs. gibi her konuya değinecek yerimiz ve vaktimiz yok. Bununla birlikte Linux ve ağ işlemleri hakkında en çok ve en detaylı dokümantasyon üretilmiş konudur. Buradaki temel bilgileri kavradıktan Linuxdoc.org sitesindeki Guides kısmında yer alan Linux Network Administrator Guide'ın özellikle 2 ile 6 arasındaki bölümlerini okumanız sizin için çok faydalı olacaktır!

17 Internet servisleri ve inetd'ye giriş

Tek bir Linux sistemi düzinelerce hatta yüzlerce ağ servisi sunabilir. Örneğin telnet programını kullandığınızda uzaktaki bir sistemin telnet hizmetine erişirsiniz. Ya da mesela ftp programını kullandığınızda uzaktaki bir sistemin ftp hizmetine erişirsiniz. Bu servisleri sunmak isteyen uzaktaki sunucu sistem ya tek tek her bir servisle ilgili yazılımı çalıştırır ve bağlantı kurulmasını bekler (örn. /usr/sbin/in.telnetd ve /usr/sbin/in.ftpd programları) ya da inetd isimli bir yazılım çalıştırır. inetd yazılımı gelen bağlantı kurma taleplerini dinler ve uygun servisle ilgili yazılımı çalıştırır. Bu yüzden de inetd aynı zamanda "Internet süpersunucusu" olarak da bilinir. Tipik bir Linux kurulumunda inetd gelen bağlantıların çoğu ile başa çıkmak üzere ayarlanmıştır. Sadece birkaç program (sshd ve lpd gibi) inetd'den bağımsız olarak ağdan gelen ve kendilerini ilgilendiren veri paketleri ile ilgilenirler.

17.1 inetd'yi ayarlamak: /etc/services

Az önce inetd'nin gelen bağlantı taleplerini türlerine göre sınıflandırdığını söylemiştik. Her bağlantı talebi temelde bir TCP/IP paketidir ve başlık kısmında bazı tanımlayıcı bilgiler barındırır. Bizi en çok ilgilendiren kısımlar kaynak adresi, hedef adresi, protokol ve port numarası bilgilerini içeren kısımlardır. Bilgisayar gelen bağlantı talepleri inetd tarafından port numarası ve protokole göre (genellikle TCP ya da UDP, inetd tarafından sunulabilen tüm servislerin bir listesi için bkz. /etc/protocols) sınıflandırılır. Her satırın formatı aşağıdaki gibidir:

```
service-ismi port-numarası/protocol-ismi aliases # yorumlar
```

Örnek bir /etc/services dosyasında bizi ilgilendiren satırlara bir göz atalım:

```
# grep ^[^#] /etc/services | head -5

tcpmux echo echo discard discard 1/tcp 7/tcp 7/udp 9/tcp 9/udp # TCP port service multiplexer

sink null sink null
```

Genellikle /etc/services halihazırda faydalı tüm servis isimlerini ve port numaraları bilgilerini içerir.

Eğer kendi servislerinizi ve port numaralarınızı ekleyecekseniz öncelikle atamaları yapılmış port numaralarını bilmeniz gerekir.

17.2 inetd'yi ayarlamak: /etc/inetd.conf

inetd yazılımın asıl konfigürasyon dosyası /etc/inetd.conf'dir ve şu formata sahiptir:

```
service-name socket-type protocol wait-flag user server-program
```

Servisler inetd.conf içinde port numarası ile değil servis ismi ile anıldıkları için bunların önceden /etc/services dosyasında tanımlanmış olması gerekir.

Örnek bir /etc/inetd.conf dosyasında bizi ilgilendiren satırlara bir göz atalım (örn. telnet ve ftp):

```
# grep ^telnet /etc/inetd.conf

telnet stream tcp nowait root /usr/sbin/in.telnetd

# grep ^ftp /etc/inetd.conf

ftp stream tcp nowait root /usr/sbin/in.ftpd -l -a
```

Her iki servis için de kullanılan TCP protokolüdür, servislerle ilgili yazılımlar (in.telnetd veya in.ftpd) root kullanıcısı olarak çalıştırılacaktır. /etc/inetd.conf dosyasındaki bilgi giriş alanlarının detaylı açıklaması için inetd(8) man sayfasına bakabilirsiniz.

17.3 Servisleri iptal etmek

inetd ayar dosyasında bulunan bir servisi iptal etmek çok basittir. Sadece ilgili satırın başına bir diyez (#) sembolü koymak sureti ile o satırı yorum satırı haline getirin. Satırı silmemeniz menfaatiniz icabıdır çünkü daha sonra o servisi yine etkin hale getirmek isteyebilirsiniz. Örneğin bazı sistem yöneticileri güvenlik sebeplerinden ötürü (iletişim şifreli olarak gerçekleştirilmediği için) telnet servisini iptal ederler:

```
# vi /etc/inetd.conf [istenmeyen satırın başına # karakterini koyun ve kaydedip çıkın]
# grep ^.telnet /etc/inetd.conf
# telnet stream tcp nowait root /usr/sbin/in.telnetd
```

17.4 inetd'yi bir başlangıç betiği ile başlatmak/durdurmak

Yukarıda bahsettiğimiz türden değişiklikleri /etc/inetd.conf dosyası üzerinde yaptıktan sonra bunların geçerli hale gelebilmesi için inetd servisini yeniden başlatmanız gerekmektedir. Pek çok Linux dağıtımı bünyesindeki /etc/init.d ya da /etc/rc.d/init.d dizininde ilgili betik dosyasını barındırır:

```
# /etc/rc.d/init.d/inet stop
Stopping INET services: # /etc/rc.d/init.d/inet start Starting INET services:

In fact, you can usually use "restart" as a shortcut:

[ OK ]

[ OK ]

# /etc/rc.d/init.d/inet restart Stopping INET services: Starting INET services:

[ [ OK OK ] ]
```

17.5 inetd'yi manuel olarak durdurmak/başlatmak

Eğer yukarıda bahsi geçen betik sizin için işe yaramazsa babadan kalma yöntemi kullanabilirsiniz, bu daha da kolaydır. inetd yazılımını durdurmak için killall komutu işinizi görür:

```
# killall inetd
```

Başlatmak için komut satırından komutu yazmanız yeterli olacaktır. Otomatik olarak arkaplanda çalışmaya başlar:

```
# /usr/sbin/inetd
```

Bunların dışında bir de inetd'yi durdurmadan ayar dosyasını yeniden okumasını sağlamanın kestirme bir yolu vardır: prosese HUP sinyalini göndermek:

```
# killall -HUP inetd
```

Bu aşamada sisteminize telnet ya da ftp programları ile bağlanamamanız gerekiyor çünkü telnet ve ftp servisleri iptal edilmiş durumda. Emin olmak için telnet localhost komutunu deneyin. (Eğer bu sisteme telnet ya da ftp ile erişmeniz gerekiyorsa yapmanız gereken az önce # ile iptal ettiğiniz satırların başındaki # karakterini kaldırmak ve inetd'yi yeniden aktive etmektir):

```
# telnet localhost
telnet: Unable to connect to remote host: Connection refused
```

17.6 TCP wrappers'a giriş

tcp_wrappers paketi tcpd isimli küçük bir daemon olup inetd tarafından ilgili servis yerine çağrılır. tcpd programı her gelen bağlantı talebinin kaynak adresini bir log dosyasına kaydeder ve sadece güvenilir sistemlerden gelen bağlantılara izin verecek şekilde filtreleme yapabilir. tcpd, kullanmak için yapmanız gereken inetd ayar dosyasına aşağıdaki gibi bir satır yazmaktır:

```
ftp      stream  tcp nowait root /usr/sbin/tcpd in.ftpd -l -a
telnet  stream  tcp  nowait  root /usr/sbin/tcpd  in.telnetd
```

17.7 TCP wrappers ile kayıt tutma

Öntanımlı olarak bağlantılar sınırlandırılmamıştır ancak bunların kaydı tutulur. Örneğin yukarıdaki değişikliği yaptıktan sonra etkin olması için inetd'yi yeniden başlatalım ve kısa bir inceleme kaydı tutulan bağlantıları görelim:

```
# telnet localhost login: (press <ctrl-d> to abort)
# tail -1 /var/log/secure
Feb 12 23:33:05 firewall in.telnetd[440]: connect from 127.0.0.1
```

telnet ile bağlanma talebi gördüğünüz gibi tcpd tarafından log dosyasına kaydedilmiştir ve buradan da tcpd yazılımının çalışmakta olduğunu anlayabiliriz. tcpd sürekli bir kayıt hizmeti sunduğundan tek tek servislerin log tutma işi ile uğraşmasına gerek kalmamış olur. Bu bakımdan tcpd'nin yaptığı iş, bağlantıları yakalayıp ilgili servisleri başlatan inetd'nin yaptığı işe benzer. Linux'un (UNIX'in) basitliği olağanüstü değil mi?

17.8 TCP wrappers kullanarak erişimi yerel kullanıcılar ile kısıtlamak

tcpd programı ile ilgili olarak iki ayar dosyası vardır: /etc/hosts.allow ve /etc/hosts.deny. Bu dosyalar aşağıdaki şekle benzerler:

```
daemon_list : client_list [ : shell_command ]
```

Erişim izni şu sıraya göre verilir ya da reddedilir.

- Arama ilk bulunan eşleşmede durdurulur.
- /etc/hosts.allow dosyasında eşleşen bir kullanıcı bulunduğunda erişim izni verilir.

- /etc/hosts.deny dosyasında eşleşen bir kullanıcı bulunduğunda erişim reddedilir. Eğer bir eşleşme bulunmaz ise izin verilir.

Örneğin sadece kendi makinamızdaki kullanıcılara izin vermek istersek /etc/hosts.deny dosyasında bir erişim politikası oluştururuz (kaynağı localhost olanlar haricindeki tüm bağlantıları reddet) :

```
in.telnetd: ALL EXCEPT LOCAL
```

17.9 TCP wrappers ile sadece bilinen kullanıcılara izin vermek

Bu aşamada inetd'yi yeniden yüklemeye gerek yoktur çünkü her telnet talebinde tcpd yeniden çağrılır. Hemen bir deneme yapabiliriz:

```
# telnet box.yourdomain.com
Trying 10.0.0.1... Connected to box.yourdomain.com. Escape character is '^]'.
Connection closed by foreign host.
```

Al sana! Reddedildin! (Reddedilmenin başarı anlamına geldiği ender anlardan biridir, tadını çıkarın ;-) Ağ içindeki kullanıcılara izin vermek için istisnai durumu /etc/hosts.allow dosyası içine yazmamız yeterlidir:

```
in.telnetd: .yourdomain.com
```

Bu noktada artık sisteme telnet ile yeniden girebilir hale geldik. Şimdiyi kadar anlattıklarımız tcp_-wrappers yazılımının sadece birkaç yüzeysel özelliğine işaret ediyordu. tcp_wrappers ile ilgili çok daha detaylı bilgi mevcuttur ve bunları tcpd(8) ile hosts_access(5) man sayfalarında bulabilirsiniz.

17.10 xinetd: geliştirilmiş (extended) inetd

inetd klasik ve geleneksel Internet süpersunucusu olmakla birlikte son dönemlerde bu sistem yeniden yazılmış ve güvenlikle de ilgili olmak üzere daha özellikli hale getirilmiştir. Red Hat ya da Debian gibi pek çok modern Linux dağıtımında xinetd programı inetd'nin yerini alır. Bu gelişmiş yazılımın özellikleri arasında şunlar da vardır:

- Erişim kontrolü (TCP wrappers hazır olarak gelir)
- Gelişmiş kayıt tutma yeteneği (bağlantı süreleri, başarısız bağlantılar, vs.)
- Başka bir makinadan servislerin yönlendirilmesi
- IPv6 desteği
- Tek bir ayar dosyası yerine parçalı ayar dosya yapısı

17.11 xinetd ayarlaması

xinetd ayar dosyası ismi /etc/xinetd.conf'dur. Genellikle bu dosya sadece birkaç satır barındırır ve birkaç ayar içerir, detaylı ayarlamalar başka dosyalara havale edilir:

```
# cat /etc/xinetd.conf

defaults {

instances log_type = 60 = SYSLOG authpriv

log_on_success = HOST PID log_on_failure = HOST RECORD includedir /etc/xinetd.d }
```

Yukarıda gördüğünüz son satır xinetd'ye detaylı ayarları /etc/xinetd.d dizinindeki dosya parçacıklarından okumasını söyler. rectory. telnet ile ilgili bölüme bakalım:

```
# cat /etc/xinetd.d/telnet

service telnet {

flags socket_type wait user server = REUSE = stream = no = root = /usr/sbin/in.telnetd

log_on_failure += USERID }
```

Gördüğünüz gibi xinetd'yi ayarlamak zor değildir ve inetd'ye kıyasla daha kolaydır. xinetd ile ilgili detaylı bilgilere xinetd(8), xinetd.conf(5) ve xinetd.log(5) man sayfalarından erişebilirsiniz. Web üzerinde de inetd, tcp_wrappers ve xinetd ile ilgili bir hayli detaylı bilgi bulabilirsiniz. Lütfen bu dokümanın Kaynaklar kısmında verilen kaynakları incelemeyi ihmal etmeyin.

18 Güvenliğe Genel Bakış - Giriş

Tamamen güvenli bir sistem kurmak imkansızdır. Ancak sabırlı ve dikkatli şekilde çalışarak kuracağınız Linux sunucularınızı cracker'ların, script-kiddie'lerin ve diğer kötü adamların çoğundan koruyabilirsiniz. Bazı standart ve önemli noktalara dikkat ederek sisteminiz üzerinde denemeler yapmaya çalışan şüpheli (ve genellikle amatör) şahısları caydırabilirsiniz. Ancak unutmayın ki sadece burada anlatılan adımları takip ederek %100 güvenli bir sistem kurmanız mümkün değildir. Bunun yerine konu ile ilgili ana başlıklara değinecek ve güvenlik konusunun temellerine giriş yapmanız için basit örnekler vereceğiz. Linux sistem güvenliği kabaca ikiye ayrılabilir: dahili güvenlik ve harici güvenlik. Dahili güvenlik ile kast edilen sistemdeki kullanıcıların istemeden ya da bilinçli olarak sisteme zarar vermelerini engellemektir. Harici güvenlik ile kast edilen ise açık olarak izin verilip yetkinlendirilmiş kullanıcılar haricindeki kullanıcıların sisteme izinsiz girişlerini engellemektir. Bu bölümde önce dahili sonra da harici güvenlik konularına kısaca değineceğiz ve genel yöntemler ile ipuçları ile bölümü tamamlayacağız.

18.1 Dosya izinleri ve log dosyaları

Dahili güvenlik ağdaki kullanıcılarınıza ne kadar güvenebileceğinize bağlı olarak bir hayli karmaşık bir görev halini alabilir. Buradaki yordamlar kısaca bir kullanıcının hassas verilere erişmesi ve sistem kaynaklarını sömürmesini engellemeye yönelik olacaktır. Dosya izinleri ile ilgili olarak, bunları üç durumda değiştirmek isteyebilirsiniz: Öncelikle /var/log dizinindeki dosyaları herkesin okuyabilmesi gerekmez. Sistemde root kullanıcısı dışında kimsenin log dosyalarına burnunu sokması için geçerli bir sebebi yoktur. syslog ile ilgili olarak LPI 101 Bölüm 4'e ve logrotate(8) man sayfasına bakabilirsiniz.

18.2 root kullanıcısının diğer dosyaları ile ilgili izinler

root kullanıcısının nokta ile başlayan dosyalarının da normal kullanıcılar tarafından okunmasının engellenmesi gerekir. ls -la komutu ile root kullanıcısının ev (home) dizinindeki dosyaların izinlerini ve düzgün şekilde korunup korunmadıklarını mutlaka kontrol edin. İsterseniz tüm dizini sadece root tarafından okunur hale de getirebilirsiniz:

```
# cd
# pwd
/root
# chmod 700 .
```

18.3 Kullanıcı dosyalarının dosya izinleri

Ve son olarak da kullanıcı dosyalarına bakalım. Genellikle bu dosyalar ilk yaratıldıklarında öntanımlı olarak herkes tarafından okunabilir şekilde yaratılırlar. Kullanıcıların çoğu böyle bir şeyi bilmez, beklemez ve zaten bunun iyi bir uygulama olduğu da söylenemez. Bunu engellemek için /etc/profile dosyasındaki umask aşağıdaki gibi bir şey yapabilirsiniz:

```
if [ "$UID" = 0 ]; then # root user; set world-readable by default so that
                              # installed files can be read by normal users.
umask 022 else # make user files secure unless they explicitly open them
                    # for reading by other users
umask 077
fi
```

umask'la ve ayarlanması ile ilgili detaylı bilgi için umask(2) ve bash(1) man sayfalarına başvurabilirsiniz. umask(2) man sayfasının aynı isimli C fonksiyonundan bahsettiğine dikkat edin, bu sizi şaşırtmasın çünkü anlatılan bilgiler doğrudan bash içinde kullanılan umask komutu için de geçerlidir. Ayrıca LPI 101 Bölüm 3'te da bununla ilgili detaylar vardır.

18.4 SUID/SGID programlarını bulmak

İzinsiz olarak ve gizlice root erişimi sağlamaya çalışan kötü niyetli bir kullanıcının ilk yapacağı şey sistemde SUID ya da SGID biti ayarlanmış programlarını bulmaya çalışmak olacaktır. LPI 101 Bölüm 3'te bahsettiğimiz gibi bu bitler ilgili programların onlara sahip olan kullanıcı ya da grubun izinleri kapsamında çalışmasına yol açarlar. Elbette bu durum bazı programların düzgün olarak çalışması için gereklidir. Ancak sorun şu ki programdaki bir hata onu çalıştıran kullanıcıya gereğinden fazla hak tanınmasına yol açabilir. SUID ya da SGID biti ayarlanmış programları dikkatli şekilde incelemeli ve buna gerçekten ihtiyaçları olup olmadığına karar vermelisiniz. Sisteminizde hiç ihtiyaç duymadığınız SUID/SGID programları bile bulunuyor olabilir. Bu tür programları aramak için find komutundan faydalanabiliriz. İsterseniz gelin /usr dizinini bir tarayalım:

```
# cd /usr
# find . -type f -perm +6000 -xdev -exec ls {} \;

-rwsr-sr-x 1 root root 593972 11-09 12:47 ./bin/gpg
-r-xr-sr-x  1 root man 38460 01-27 22:13 ./bin/man
-rwsr-xr-x  1 root root 15576 09-29 22:51 ./bin/rcp
-rwsr-xr-x  1 root root 8256 09-29 22:51 ./bin/rsh
-rwsr-xr-x  1 root root 29520 01-17 19:42 ./bin/chfn
-rwsr-xr-x  1 root root 27500 01-17 19:42 ./bin/chsh
-rwsr-xr-x  1 lp    root 8812 01-15 23:21 ./bin/lppasswd
-rwsr-x--   1 root cron 10476 01-15 22:16 ./bin/crontab
```

Karşımıza çıkan listede dikkatle incelenmesi gereken bir aday var: lppasswd isimli dosya CUPS yazdırma sisteminin bir parçasıdır. Kendi sisteminizde yazıcı servisi sunmuyorsanız bu programa da ihtiyacınız yoktur. lpppasswd programında herhangi bir hata ya da açık olmayabilir fakat kullanmadığımız bir programı sistemde bulundurarak neden risk alalım ki? Benzer şekilde kullanmadığınız tüm servisleri iptal etmekte fayda vardır. İhtiyacınız olduğu anda nasıl olsa kolayca bunları yeniden etkin hale getirebilirsiniz.

18.5 ulimit ile kullanıcıların limitlerini ayarlamak

bash kabuğundaki ulimit komutu belli bir kullanıcının hangi kaynağa ne ölçüde erişebileceğini belirlemenizi ve kısıtlamanızı sağlar. Bir kez bir limitin değeri azaltıldı mı o proses çalışırken bunu bir daha artırmanın yolu yoktur. Ayrıca o proses ile ilgili limitlere o prosesin tüm çocuk prosesleri de maruz kalır.

Bu kısaca şu demektir: /etc/profile dosyasında yapacağınız ulimit ayarları tüm kullanıcıları etkileyecektir (kullanıcıların bash ya da /etc/profile dosyasını okuyan bir kabuk yazılımını çalıştırdıklarını var sayıyoruz). Mevcut limit durumunu öğrenmek için ulimit -a komutunu kullanabiliriz:

```
# ulimit -a
core file size  (blocks, -c) 0
data seg size (kbytes, -d) unlimited
file size         (blocks, -f) unlimited
max locked memory  (kbytes, -l) unlimited
max memory size     (kbytes, -m) unlimited
open files               (-n) 1024
pipe size                (512 bytes, -p) 8
stack size              (kbytes, -s) unlimited
cpu time                (seconds, -t) unlimited
max user processes  (-u) 3071
virtual memory         (kbytes, -v) unlimited
```

Bu limitleri sistemin güvenliğini artıracak ve kullanıcıları rahatsız etmeyecek şekilde ayarlamak çok zor olabilir bu yüzden ulimit ile ayarlama yaparken çok dikkatli olmalısınız.

18.6 ulimit ile CPU zamanını sınırlamak

Bir ulimit örnek uygulaması olarak bir prosesin CPU zamanını 1 saniye ile kısıtlamayı ve sonra prosesi döngüye sokarak zaman aşımına uğratmayı deneyelim. Yeni bir bash prosesi başlatmayı unutmayın aksi takdirde bu denemenin sonucunda sistemden çıkarsınız!

```
# time bash
# ulimit -t 1
# while true; do true; done
Killed

real  0m28.941s
user 0m1.990s
sys  0m0.017s
```

Yukarıdaki örnekte "user" zamanı artı "sys" zamanı proses tarafından kullanılan toplam CPU zamanını verir. bash prosesi 2 saniye işaretin gelince Linux çekirdeği 1 saniye limitinin geçildiğini tespit eder ve proses sonlandırılır. Harika değil mi?

Not: Bir saniyelik limit sadece bir örnekti. Sakın kullanıcılarınıza böyle bir şey yapmayın! Birkaç saatlik limit bile sorun yaratabilir çünkü X grafik sunucusu gerçekten CPU zamanını çok kullanır. Gerçek bir uygulama için CPU zamanı haricinde bir kaynağı kısıtlamayı düşünün.

18.7 Kullanıcı limitleri, devam

Eşzamanlı (simultane) loginler ya da disk kullanım miktarı gibi kaynakları da kısıtlamak isteyebilirsiniz ancak bunlar ulimit ile ilgili konular değildir. Bu tür kısıtlamalar için aşağıdaki yazılım paketlerini inceleyebilirsiniz:

- Clobberd yazılımı kullanıcı aktivitesini gözetler ve zaman, ağ kullanımı gibi değerleri ölçer.
- Idled yazılımı çok uzun süre iş yapmayan, boş duran kullanıcıları ya da uzun süredir sisteme bağlı olan kullanıcıları sistemden atabilir.
- Aynı zamanda kullanıcıların belli bir miktardan fazla login olmasını da engelleyebilir.
- LPI 101 Bölüm 4'te dosya sistemi kotaları ile kısıtlama anlatılmıştır.

18.8 Gizli girişlerin engellenmesi (intrusion prevention)

Harici güvenlik iki kategoride ele alınabilir: Gizli girişlerin engellenmesi ve gizli girişlerin tespit edilmesi. Gizli giriş engelleme tedbirleri yetkisiz kullanıcıların sisteme girişlerini engellemek içindir. Eğer bu tedbirler işe yaramazsa gizli giriş tespit sistemleri ile izinsiz girişin ne zaman gerçekleştirildiği ve hasar miktarı tespit edilebilir. Tam bir Linux kurulu sistem oldukça karmaşık bir sistemdir, kurulmuş olan her şeyin kaydını tutmak hele de her paketin güvenliği ile uğraşmak kolay bir iş değildir. Paket sayısı az ise bunları kontrol etmek de kolay olacaktır. Gizli giriş önlemenin ilk kuralı ihtiyaç duymayacağınız paketleri kurmamak ya da kaldırmaktır. Paket yönetim sistemlerine genel bakış için LPI 101'in 4. bölümüne bakabilirsiniz.

18.9 Kullanılmayan ağ servislerini kapatmak (süpersunucu)

Kullanılmayan ağ servislerini kapatmak gizli girişleri engellemek için atılacak ilk adımlardan biridir. Örneğin inetd ya xinetd gibi bir Internet süpersunucusu çalıştırıyorsanız genellikle in.rshd, in.rlogind ve in.telnetd gibi servisler öntanımlı olarak açıktır. Bu ağ servislerinin hemen hemen tamamı ssh gibi daha güvenli alternatiflere sahiptirler. inetd servislerini iptal etmek için /etc/inetd.conf dosyasındaki ilgili servis satırlarının başına "#" sembolü yazmak sureti ile bunları yorum satırı haline getirmek ve inetd'yi yeniden başlatmak yeterlidir.

xinetd kullanıyorsanız servisleri iptal etmek için benzer şeyleri /etc/xinetd.d içindeki ilgili dosya üzerinde yapmanız yeterli olacaktır. Örneğin telnet servisini kapatmak için ya /etc/xinetd.d/telnet dosyasının içeriğinin tamamını yorum satırı haline getirin veya basitçe dosyayı silin. Sonra xinetd'yi yeniden başlatın. Eğer inetd ile birlikte tcpd kullanıyorsanız ya da süpersunucu olarak xinetd kullanıyorsanız gelen bağlantı taleplerini sadece güvenilir bilgisayarlarla sınırlama imkanınız vardır. tcpd ile ilgili olarak bu kılavuzun önceki bölümlerine bakabilirsiniz.

xinetd için ise xinetd.conf(5) man sayfasında "only_from" ifadesini aratın.

18.10 Kullanılmayan ağ servislerini kapatmak (tek başına çalışan sunucular)

Bazı ağ servis sunucuları inetd ya da xinetd tarafından başlatılmaz kendi başlarına çalışırlar. Birkaç örnek vermek gerekirse: atd, lpd, sshd, nfsd ve benzerleri. Aslında inetd ve xinetd de kendi başlarına çalışan sunuculardır ve ilgili ayar dosyalarındaki satırların hepsini yorum satırı haline getirdiyseniz bunları tamamen kapatmayı da düşünebilirsiniz. Kendi başına çalışan sunucular genellikle init sistemi tarafından bilgisayar açılırken ya da runlevel değişirken devreye sokulurlar.

init sisteminin belli bir sunucuyu başlatmasını engellemek için ilgili runlevel dizinindeki ilgili dizinin başlatma betiğine yönlendirilmiş sembolik bağlantıyı bulun ve bunu silin. runlevel dizinleri /etc/rc3.d veya /etc/rc.d/rc3.d gibi isimlere sahiptir (runlevel 3 için). Tabii bu arada diğer runlevel'larla ilgili dizinleri de kontrol etmek isteyebilirsiniz. Durdurmak istediğiniz servislerle ilgili sembolik bağlantıları sildikten sonra halihazırda aktif olan sunucuyu da durdurmanız gerekmektedir. Bunu yapmanın en iyi yolu ilgili servisin init betiğini kullanmaktır ki bu da genellikle /etc/init.d veya /etc/rc.d/init.d dizininde bulunur. Örneğin sshd yazılımı durdurmak için şuna benzer bir komut vermeniz yeterli olacaktır:

```
# /etc/init.d/sshd stop
* Stopping sshd... [ ok ]
```

18.11 Değişiklikleri test etmek

inetd ya da xinetd ayarlarınızı düzenledikten ve gerekli kısıtlamaları yaptıktan sonra veya bir sunucuyu init betiği ile durdurduktan sonra işlerin yolunda gidip gitmediğini anlamak için sisteminizi test etmelisiniz. TCP portlarını telnet istemcisi ile ve servis ismini ya da port numarasını vererek test edebilirsiniz. Örneğin rlogin servisinin iptal edildiğinden emin olmak için:

```
# grep ^login /etc/services
login 513/tcp

# telnet localhost 513
Trying 127.0.0.1...
telnet: Unable to connect to remote host: Connection refused
```

Standart telnet istemcisine ek olarak sisteminizin "açıklığı"nı test etmek için faydalanabileceğiniz birkaç becerikli yazılım daha vardır. netcat ve nmap isimli yazılımlar bunlar arasında en meşhur olanlarıdır. ncat bilgisayar ağlarının İsviçre Çakısı olarak bilinir, temel olarak yaptığı içi ağ bağlantıları kurup TCP ya da UDP protokollerini kullanarak veri okuyup yazmaktır. nmap ise ağı keşfetmek veya güvenliği denetlemek için kullanılan bir araçtır. Esas olarak nmap belli bir adresteki (ya da adreslerdeki) bilgisayarların üzerindeki açık portları tarayıp bunları tespit etmeye yarar. Bu faydalı programlarla ilgili web adreslerini en sondaki Kaynaklar bölümünde bulabilirsiniz.

18.12 Bakım yaparken kullanıcıların girişini engelleme

Yukarıdaki yöntemlere ek olarak loginleri yani sisteme girişleri engellemenin çok genel bir yöntemi vardır: /etc/nologin isimli bir dosya oluşturmak. Bu yöntem sadece kısa süreli sistem bakım işlemlerinde kullanıcıların sisteme girişini engellemek için kullanılır. root kullanıcısı sisteme girebilir fakat diğer (normal) kullanıcılar giremez. Örneğin:

```
# cat > /etc/nologin ===== System is currently undergoing maintenance until
2:00. Please come back later. ===
# telnet localhost login: agriffis

Password:
Login incorrect
==== System is currently undergoing maintenance until 2:00.
Please come back later. ===
```

Bakım işlemleriniz bitince söz konusu dosyayı silmeyi sakın unutmayın, aksi takdirde hiçkimse sisteme giremez. (Yok hayır, ben böyle bir dalgınlık sonucu insanlara saç baş yoldurtmadım, kesinlikle bunu yapan ben değildim ;-)

18.13 iptables (ipchains) konusuna giriş

iptables ve ipchains komutları çalışmakta olan bir Linux çekirdeğinin ağ paket filtreleme kurallarını incelemek ve ayarlamak için kullanlır. ipchains komutu 2.2.x sürümü çekirdekler için kullanılıyordu ve her ne kadar 2.4.x serisi çekirdekler üzerinde kullanılabilse debu komutun yerini iptables komutu almıştır. Paket filtreleme kuralları hem firewall hem de router aktivitelerini ayarlamak için kullanılabilir. Mevcut kuralları incelemek için iptables komutunu -L parametresi ile kullanabilirsiniz:

```
# iptables -L
Chain INPUT (policy ACCEPT) target prot opt source
    Chain FORWARD (policy ACCEPT) target prot opt source
Chain OUTPUT (policy ACCEPT) target prot opt source

destination

destination

destination
```

Bu örnekte gördüğümüz sonuna kadar açık bir sistedir, herhangi bir routing ve firewall işlevi yoktur.

18.14 iptables ve Linux paket filtresi

Linux paket filtresini etkin olarak kullanmak sağlam bir TCP/IP bilgisini ve bunun Linux çekirdeğinde nasıl uygulandığını bilmeyi gerektirir. netfilter ana sayfası (kılavuzun sonundaki kaynaklar kısmına bakın) daha çok bilgi edinmek için uygun bir başlangıç noktasıdır. Kendi kural setinizi sağlıklı oluşturabilecek seviyeye gelene dek size yardımcı olabilecek birçok hazır betik vardır. Bunlardan en kapsamlı olanının ismi gShield isimli bir betik dosyasıdır (kaynaklar bölümüne bakın). Bünyesinde birçok açıklama satırı barındıran bu ayar dosyasını kullanarak en çok kullanılan paket filtreleme kurallarını sisteminizde aktive edebilirsiniz.

18.15 Gizli girişlerin tespiti - syslog dosyaları

Gizli girişlerin tespit konusu, sistemlerini özel gizli giriş tespit sistemlerine (IDS - Intrusion Detection System) emanet etmiş sistem yöneticileri tarafından genellikle ihmal edilir. Maalesef böyle bir durum, kötü niyetli biri söz konusu özel sistemde bir açık bulup bunu geçebilirse karşısına bir hayli kolay girilebilir bir sistem çıkacak demektir ve sistem yöneticisi uzunca bir süre bu durumda haberdar olmayabilir.

Gizli giriş tespiti yapmanın temel yolu sistem kayıtlarına dikkat etmekten geçer yani syslog dosyalarına. Bu dosyalar genellikle /var/log dizininde dururlar ancak isimleri kullandığınız dağıtıma göre bazı değişiklikler gösterebilir.

```
# less /var/log/messages

Feb 17 21:21:38 [kernel] Vendor: SONY Model: CD-RW CRX140E Rev: 1.0n
Feb 17 21:21:39 [kernel] eth0: generic NE2100 found at 0xe800,
    Version 0x031243, DMA 3 (autodetected), IRQ 11 (autodetected).
Feb 17 21:21:39 [kernel] ne.c:v1.10 9/23/94 Donald Becker (becker@scyld.com)
Feb 17 21:22:11 [kernel] NVRM: AGPGART: VIA MVP3 chipset Feb 17 21:22:11
[kernel] NVRM: AGPGART: allocated 16 pages Feb 17 22:20:05 [PAM_pwdb]
    authentication failure; (uid=1000) -> root for su service
Feb 17 22:20:06 [su] pam_authenticate: Authentication failure
Feb 17 22:20:06 [su] - pts/3 chouser-root
```

Bu mesajların tamamının anlamını kavrayacak hale gelmeniz biraz vakit alabilir ancak en önemlileri kendilerini zaten belli eder. Örneğin yukarıdaki log dosyasının sonuna doğru "chouser" isimli bir kullanıcının su komutu ile root haklarına erişmeye çalıştığını ve başarısız olduğunu görüyoruz.

18.16 Gizli giriş tespiti - tripwire

Bazı özel yazılımlar mevcut dosya sisteminin bir tür fotoğrafını çekip bunu daha sonraki dosya sistemi yapısı ile hızlıca kıyaslayabilirler ve değişiklik olup olmadığına bakabilirler. Eğer sistemin normal işleyişi sırasında hangi dosyaların ya da dizinlerin değişmesinin kabul edilebilir hangilerinin kabul edilemez

olduğunu belirtirseniz bu yazılımlar sizi değişiklikler konusunda hızlıca uyarırlar: bazı hassas dosyalardaki küçük ve önemsiz gibi görünen değişiklikler sisteme gizlice girilip dosyalara müdahale edildiğine işaret edebilir. Tripwire bu tür yazılımlardan en meşhur olanıdır (bağlantı için kılavuzun sonundaki kaynaklar kısmına bakın). Bir kere kurduktan sonra tripwire ayar dosyasını kendi sisteminize ve ihtiyaçlarınıza göre ayarlamanız ve yazılımı düzenli olarak çalıştırıp (genellikle günde bir kez) sistemi olası şüpheli dosya değişimlerine karşı test etmeniz gerekir.

18.17 Gizli giriş tespiti - portsentry

Psionic Technologies tarafından geliştirilmiş olan PortSentry paketi gizli giriş önleme ile gizli giriş tespit etme kategorilerine giren melez bir uygulamadır. Çalıştırıldığı zaman bilgisayarınızın ağ trafiğini gözetler ve "şüpheli" kabul ettiği herhangi bir ağ trafiği ile karşılaşırsa bu olayın kaydını tutar ve aynı kaynaktan aynı tür olayın gerçekleşmesini engeller. Bu yazılımla ilgili bağlantılar da Kaynaklar bölümünde bulunabilir. Bir kez yazılımı kurup çalıştırdığınızda PortSentry'nin şüpheli eylemlere karşı neler yaptığını syslog dosyanızda görebilirsiniz:

```
# tail /var/log/messages
Oct 15 00:21:24 mycroft portsentry[603]: attackalert: SYN/Normal scan from
    host: 302.174.40.34/302.174.40.34 to TCP port: 111
Oct 15 00:21:24 mycroft portsentry[603]: attackalert: Host 302.174.40.34
    "ALL: 302.174.40.34" has been blocked via wrappers with string:
Oct 15 00:21:24 mycroft portsentry[603]: attackalert: Host 302.174.40.34 has
    been blocked via dropped route using command: "/sbin/route add -host
    302.174.40.34 reject"
Oct 15 00:21:24 mycroft portsentry[603]: attackalert: SYN/Normal scan from
    host: 302.174.40.34/302.174.40.34 to TCP port: 111
Oct 15 00:21:24 mycroft portsentry[603]: attackalert:
    Host: 302.174.40.34/302.174.40.34 is already blocked Ignoring
Oct 15 00:33:59 mycroft portsentry[603]: attackalert: SYN/Normal scan from
    host: 302.106.103.19/302.106.103.19 to TCP port: 111
Oct 15 00:33:59 mycroft portsentry[603]: attackalert: Host 302.106.103.19 has
    been blocked via wrappers with string: "ALL: 302.106.103.19"
Oct 15 00:33:59 mycroft portsentry[603]: attackalert: Host 302.106.103.19 has
    been blocked via dropped route using command: "/sbin/route add -host
    302.106.103.19 reject"
```

18.18 Genel tavsiyeler: Yazılımı Güncel Tutmak

Ne kadar test edilmiş olursa olsun hemen her yazılımın süreç içinde güvenlikle ilgili hataları tespit edilebilir. Bunlar tespit edildikten kısa bir süre sonra söz konusu yazılımların tamir edilmiş halleri de yayınlanır ve sistem yöneticisinin bunları güncellemesi gerekir. Az önce okumuş olduğunuz tavsiye güvenlik uzmanları tarafından en çok tekrarlanan ve amatör sistem yöneticileri tarafından da en çok görmezden gelinen tavsiyedir. Bir musibet bin nasihate bedeldir ama bu şekilde bir şeyler öğrenmek canınızı yakabilir: Bilgisayarlarınız bir yıldır bilinen ve güvenlik yamaları çoktan yayınlanmış (ve sizin tarafınızdan maalesef uygulanmamış) bir açık yüzünden zarar görürse üzülürsünüz.

"Açık kaynaklı yazılımlar mı yoksa kapalı kaynak kodlu yazılımlar mı daha güvenlidir?" tartışması ise uzun zamandı süren çok sıcak bir tartışmadır. Bu konuda söylenebilecek en iyi şey belki de şudur: düzgün olarak yönetildikleri sürece ve güvenlik yamaları zamanında uygulandığı sürece her iki sistem de güvenli kabul edilebilir. Pek çok web sitesi düzenli olarak tespit edilmiş açıkları ve bunları gideren yamaları yayınlar. Bunlardan en önemlileri ve güvenilir olanları CERT ve SecurityFocus' BugTraq listesidir. Ayrıca freshmeat.net gibi yazılım güncelleme haberlerini insanlara sunan siteler de yazılımlarla ilgili açıkları ve çözümleri duyururlar. Bir başka önemli kaynak da kullandığınız yazılımların resmi web siteleridir. Bahsi geçen önemli sitelerin adresleri kaynaklar bölümünde verilmiştir. Bu arada CERT ve Bugtraq sitelerini ziyaret edip aşina olursanız sizin için iyi olur.

18.19 Genel tavsiyeler: yüksek kaliteli parolalar

Sıradan ve sıkıcı bir tavsiye gibi görünse de kendiniz ve kullanıcılarınız için yüksek kaliteli parolalar seçmeniz, bu konuda kullanıcılarınızı eğitmeniz güvenli bir sistem kurmanın en temel aşamalarından biridir. Asla ve kat'a kolay tahmin edilebilen sözcükler, yakınlarınızın isimleri, telefon numaralarınız, doğumgünü tarihiniz, evde beslediğiniz hayvanın ismi, sevgilinizin adı, vb.gibi kolay tahmin edilebilir

bilgileri kullanmayın. Parolanızda mümkünse aynı zamanda hem harfler hem rakamlar hem de noktalama işaretleri bulunsun.

18.20 Genel tavsiyeler : Güvenliğinizi Test Etmek

Sisteminizin güvenliğini test etmek önemlidir ancak başarılı geçen bir testin ardından rahatlamak gibi bir lüksünüz yoktur. Belli bir teste göre sisteminizin güvenli görünmesi hiçbir açık barındırmadığı anlamına gelmez. Yeterli zamanı, azmi ve araştırma yeteneği olan kötü niyetli bir insanın sisteminize giremeyeceği garanti edilemez.

Yukarıdaki bölümlerde ağ güvenliği için nmap ve netcat yazılımlarından bahsetmiştik. Bunların yanısıra zayıf parolaları tespit etmek de önemlidir. Bu konularla ilgili faydalı araçlar kılavuzun sonundaki Kaynaklar bölümünde yer almaktadır.

19 Yazdırma İşlemlerine Giriş

Bu bölümde Linux üzerinde de kullanılan ve bazen Berkeley LPD olarak da adlandırılan klasik UNIX yazdırma sisteminin kurulumundan ve kullanılmasından bahsedeceğiz. Linux üzerindeki yazdırma servislerine dair başka sistemler de mevcuttur ve bunları Kaynaklar kısmında görebilirsiniz. Bir yazıcının fiziksel olarak kurulması bu kılavuzun kapsamı dışındadır. Yazıcıyı bilgisayara bir kez fiziksel olarak bağladıktan sonra bir yazıcı "spooler" dameon'ı kurmak isteyeceksiniz ki ağdaki bilgisayarlar ve yazıcının bağlı olduğu bilgisayar yazıcıya yazdırılabilecek belgeleri gönderebilsinler.

19.1 Bir yazıcı spooler daemon kurulumu (lpd)

En iyi LPD yazıcı "spooler" yazılımlarından biri LPRng'dir. Kurulum yöntemi dağıtımınıza göre biraz değişiklik gösterebilir. Red Hat ya da Debian yazılım paketlerinin kurulumu için LPI 102 1. bölüme bakabilirsiniz. Bir kez kurulduktan sonra yazıcı spooler daemon'ı (resmi ismi ile Line Printer Daemon) komut satırından çalıştırılabilir. Normal bir kullanıcı olarak sisteme girin ve şunu deneyin:

```
$ /usr/sbin/lpd -help
--X option form illegal
usage: lpd [-FV] [-D dbg] [-L log]
 Options
 -D dbg - set debug level and flags
Example: -D10,remote=5
set debug level to 10, remote flag = 5
 -F  - run in foreground, log to STDERR
Example: -D10,remote=5
 -L logfile  - append log information to logfile
 -V  - show version info
```

Daemon kurulduğuna göre şimdi her seferinde otomatik olarak başlatılacağından emin olmalısınız. Dağıtımınızın LPRng paketi bunun sizin için çoktan ayarlamış olabilir ancak eğer yapmadı ise LPI 101 4. bölüm size runlevel'lar ve lpd gibi daemon yazılımlarının otomatik olarak başlatılması konusunda size bilgi verecektir.

19.2 Temel yazıcı ayarları (/etc/printcap)

Yazdırma spooler daemon'ı bir tür boruhattı gibi çalışır. Pek çok farklı noktadan kendisine gelen yazdırma işlerini kabul eder ve bunları uygun yazıcılara yönlendirir. Yazıcı meşgul iken bu işler "spool"a yani kuyruğa girerler ve yazdırılmak için sıralarını beklerler. Makinaya doğrudan bağlı bir yazıcıya yazdırırken bu boru hattının her iki ucu da /etc/printcap (ya da /etc/lprng/printcap) dosyasındaki ayarlarla belirlenir. printcap dosyasındaki her girdi (bu dosyanın ismi "printer capabilities" sözcüklerinin kısaltmasından gelir) bir yazdırma kuyruğunu gösterir:

```
$ more /etc/printcap
lp|Generic dot-matrix printer entry:\
:lp=/dev/lp0:\
:sd=/var/spool/lpd/lp:\
:pl#66:\
```

```
:pw#80:\
:pc#150:\
:mx#0:\
:sh:
```

Son girdinin en sonunda bir tersbölü (
) bulunmadığına dikkat edin. Sizin sisteminizde başka girdiler de olabilir ve bunlar daha karmaşık olabilir ancak genel olarak bu dosyanın formatı yukarıda gördüğünüz gibidir. Önce girdinin ismi gelir, lp ardından da bu kuyruğun detaylı bir tanımlaması. lp=/dev/lp0 şeklindeki anahtar sözcük, değer ikilisi bu kuyruktaki yazdırma işlerinin hangi cihaza yönlendirileceğini gösterirken sd anahtar sözcüğü de yazdırılacak işlerin hangi dizinde barındırılacağını belirtir. Anahtar sözcük, değer çiftlerinin geriye kalanları /dev/lp0 dosyasına fiziksel olarak nasıl bir yazıcı bağlı olduğunu belirtir.

19.3 Spool dizinlerinin yaratılması

Eğer bir girdi yaratırsanız buna paralel olarak ilgili spool dizinin doğru izinler verilmiş halde mevcut olduğundan emin olmanız gerekir. Genellikle lp kullanıcısı olarak çalışan yazıcı daemon'ının bu spool dizinine erişmesini isteyeceksiniz. root kullanıcısı olarak şu komutları vermeniz gerekir:

```
# mkdir -p /var/spool/lpd/lp
# chown lp /var/spool/lpd/lp
# chmod 700 /var/spool/lpd/lp
# checkpc -f
# /etc/init.d/lprng restart
```

LPRng printcap'i kontrol edebilmeniz için faydalı bir araç içerir. Aynı zamanda, eğer siz unutursanız spool dizinini de ayarlar:

```
# checkpc -f
```

Son olarak lpd'yi tekrar başlatın. Ne zaman printcap'i değiştirseniz değişikliklerin geçerli olması için bunu yapmanız gerekir. lpd yerine lprng kullanmanız gerekebilir:

```
# /etc/init.d/lprng restart
```

Eski Berkeley yazdırma sistemi checkpc aracını içermez bu yüzden birden fazla yazıcınız var ise bunlara bastırılacak belgeler yollayarak printcap ve spool dizinlerinin doğru olup olmadığını kontrol etmeniz gerekir.

19.4 Yazdırma spooler istemcilerini kullanmak

Yazdırma spooler istemcisi sunucu daemon ile iletişim kurabilecek pek çok istemci ile birlikte gelir. En çok kullanılanı lpr programıdır ve yaptığı şey bir yazdırma kuyruğuna ilgili belgeyi yollayıp yazdırılmasını sağlamaktır. Bunu denemek için önce küçük bir metin dosyası oluşturun ve sonra da:

```
$ lpr sample.txt
```

Eğer çalıştı ise ekranda herhangi bir mesaj görmezsiniz fakat yazıcınız çalışmaya başlar ve dosyanızın kağıda basıldığını görebilirsiniz. Eğer çok düzgün görünmüyorsa üzülmeyin biraz sonra gerekli filtreleri nasıl kuracağımızı ve her türlü formatı nasıl yazıcıya gönderebileceğimizi inceleyeceğiz. Yazıcı kuyruğundaki yazdırma "iş"lerinin listesini lpq komutu ile ile öğrenebilirsiniz. -P seçeneği hangi kuyruğun gösterileceğini belirler eğer bunu es geçerseniz öntanımlı kuyruk listelenecektir:

```
$ lpq -Plp
Printer: lp@localhost 'Generic dot-matrix printer entry'
 Queue: 1 printable job
 Server: pid 1671 active
 Unspooler: pid 1672 active
 Rank Owner/ID Class Job Files Size Time
active chouser@localhost+670  A  670  sample.txt 8  21:57:30
```

Eğer bir yazdırma işini iptal etmek istiyorsanız lprm komutunu kullanmalısınız. Bu komut bir yazdırma işi çok uzun sürüyorsa ya da bir kullanıcı yanlışlıkla aynı belgeyi bir kereden daha çok göndermişse işe yarar. Tek yapmanız gereken yukarıdaki listeden job id'sini tespit etmek:

```
$ lprm chouser@localhost+670
Printer lp@localhost:
checking perms 'chouser@localhost+670'
dequeued 'chouser@localhost+670'
```

lpc ile yazıcı kuyruğunda pek çok şey yapabilirsiniz. Detaylar için lpc man sayfasına bakınız.

19.5 Uzaktaki bir LPD sunucusuna yazdırmak

Doğrudan kendi bilgisayarınıza bir yazıcı bağlı olmasa bile yine de lpd'yi kullanarak ağ üzerindeki bir bilgisayara bağlı bulunan yazıcıya belgelerinizi gönderebilirsiniz. İstemci makinada /etc/printcap dosyasına yerel bir yazıcı gibi görünen ama aslında yazdırma işlerini uzaktaki makinaya yönlendiren bir girdi oluşturabilirsiniz:

```
farawaylp|Remote printer entry:\
:rm=faraway:\
:rp=lp:\
:sd=/var/spool/lpd/farawaylp:\
:mx#0:\
:sh:
```

Burada ağ üzerinde bulunan ve kendisine bir yazıcı bağlı olan makinanın ismi faraway'dir ve bu makinadaki yazıcının ismi de lp'dir. spool dizini /var/spool/lpd/farawaylp'dir ve uzaktaki makinanın spool dizinine gönderilmek üzere sırada bekleyen yazdırma işleri bu dizinde barındırılacaklardır. Tabii bu spool dizinini uygun izinleri de vererek yaratmanız gerektiğini unutmayın:

```
# mkdir -p /var/spool/lpd/farawaylp
# chown lp /var/spool/lpd/farawaylp
# chmod 700 /var/spool/lpd/farawaylp
# checkpc -f
# /etc/init.d/lprng restart
```

Yerel olarak uzaktaki yazıcıya farawaylp ismini verdiğimize göre ona yönlendirilmiş işleri de şu şekilde görüntüleyebiliriz:

```
$ lpr -Pfarawaylp sample.txt
```

19.6 Uzaktaki bir MS Windows ya da Samba sunucusuna yazdırmak

Samba sayesinde uzaktaki bir Microsoft Windows yazıcı sunucusuna belge yollamak zor bir iş olmaktan çıkmıştır. Önce yerel printcap girdimizi oluşturalım:

```
smb|Remote windows printer:\
:if=/usr/bin/smbprint:\
:lp=/dev/null:\
:sd=/var/spool/lpd/smb:\
:mx#0:
```

Buradaki yeni anahtar if'dir yani girdi filtresi (input filter). Bunu smbprint betiğine yönlendirmek yazma işinin lp cihazı yerine uzaktaki bir Windows sunucusuna yönlenmesine yol açacaktır. Bu durumda yine de yazdırma daemon'ın kilit işlemleri için kullanacağı bir cihaz listelemeliyiz: /dev/null. Ancak tabii ki hiçbir yazdırma işi bu cihaza gönderilmeyecektir. Spool dizinini yaratmayı unutmayın!

```
# mkdir -p /var/spool/lpd/smb
# chown lp /var/spool/lpd/smb
# chmod 700 /var/spool/lpd/smb
# checkpc -f
# /etc/init.d/lprng restart
```

Favori metin editörünüz ile ismi yukarıdaki gibi belirlenmiş spool dizininde bir .config dosyası oluşturun (örneğimizde: /var/spool/lpd/smb/.config):

```
server="WindowsServerName"
service="PrinterName"
password= " "
user=" "
```

Bu değerleri Windows çalıştıran bilgisayarı ve oradaki yazıcıyı gösterecek şekilde ayarlayın, ve işlem tamam:

```
$ lpr -Psmb sample.txt
```

smbprint betiği genellikle Samba ile birlikte gelir ancak her dağıtımda bulunmayabilir.Eğer bunu sisteminizde bulamazsanız Samba HOWTO belgesinde bulabilirsiniz.

19.7 Magicfilter

Şimdiye dek sadece metin dosyaları ile ilgilendik ve bu da çok heyecan verisi sayılmaz. Genellikle elinizin altındaki herhangi bir yazıcı tek bir grafik formatını basabilecek kapasitededir ancak basılmasını arzuladığımız pek çok farklı format mevcuttur: PostScript, gif, jpeg ve diğerleri. Magicfilter isimli program bir tür girdi filtresi olarak çalışır tıpkı smbprint'in yaptığı gibi. Dosya formatlarını dönüştürmez sadece basılacak belgenin tipinin belirlenmesini sağlayacak bir altyapı sunar ve sonra gerekli dönüştürme yazılımının çalıştırılmasını sağlar. Dönüştürme yazılımları ayrıca kurulmalıdır. Bunlardan en önemlisi ise Ghostscript yazılımıdır; bu yazılım Postscript formatındaki dosyaları yazıcının yerel formatına dönüştürebilir.

19.8 printcap'i Magicfilter'i gösterecek şekilde ayarlamak

Bu araçlar kurulduktan sonra yapılması gereken printcap dosyasını bir kez daha ayarlamaktır. if anahtarı Magicfilter'i gösterecek şekilde yazıcınızı ayarlayabilirsiniz:

```
lp|The EPSON Stylus Color 777 sitting under my desk:\
:if=/usr/share/magicfilter/StylusColor-777@720dpi-filter:\
:gqfilter:\
:lp=/dev/usb/lp0:\
:sd=/var/spool/lpd/lp:\
:pl#66:\
:pw#80:\
:pc#150:\
:mx#0:\
:sh:
```

/usr/share/magicfilter içinde düzinelerce farklı yazıcı için ayarlar mevcuttur, yazıcınız için doğru olanı seçtiğinizden emin olun. Bunlardan her biri bir metin dosyasıdır ve yazıcının tam ismi en tepededir, bu şekilde hangi dosyaya ihtiyacınız olduğunu tespit edebilirsiniz. Ayrıca eklenmiş olan bir gqfilter seçeneği de uzaktaki bir istemciden gelen belgelerin yine filtreye yönlendirilmesini sağlar. Bu sadece LPRng ile çalışır. Daha önce /var/spool/lpd/lp yazıcı spool dizini daha önce ayarlanmış olduğu için tek yapılması gereken printcap sözdizimini kontrol etmek ve sunucuyu yeniden başlatmak:

```
# checkpc -f
# /etc/init.d/lprng restart
```

Artık Postscript dosyaları da dahil olmak üzere çok çeşitli belgeleri yazdırabilirsiniz. Başka bir deyişle favori web tarayıcınızın menüsündeki "Print" seçeneği artık düzgün şekilde çalışıyor olmalı.

19.9 Magicfilter alternatifi olarak Apsfilter

Apsfilter, Magicfilter'in sunduğu özelliklerin çoğunu sunar fakat aynı zamanda spool dizinler, printcap girdileri gibi şeyleri de kendisi ayarlar.

Ghostscript'in yine de kurulu olması gereklidir, ardından Apsfilter elkitabındaki detaylı açıklamaları okuyarak gerekli ayarlamaları yapabilirsiniz.

19.10 Kaynaklar

xinetd anasayfasından ayarlamalarla ilgil detaylı bilgi edinebilirsiniz. Kendi servis isimleriniz ve port numaralarınızı /etc/services dosyasına eklerken önceden ataması yapılmış olan portlar ve servislerle çakışmamalarına dikkat edin. netfilter anasayfası iptables ile ilgili detaylı bilgi edinmek için birincil kaynaktır. Kendi kural setinizi kendi sisteminize göre hassas olarak kurabilecek kadar deneyim sahibi olana dek gShield gibi hazır bir iptables betiğini denemenizi tavsiye olunur.

İlk akla gelen güvenlik araçları arasında şunlar sayılabilir: Tripwire en popüler gizli giriş tespit araçlarındandır.

Psionic Technologies'den gelen PortSentry ise hem tespit etme hem de önlemeye yöneliktir. (LinuxWorld'deki "How to stop crackers with PortSentry" başlıklı makale bu yazılımla ilgili detaylı bilgi verir)

Bunlara ek olarak Wietse Venema'nın TCP Wrappers yazılımına aşina olmanız menfaatiniz icabıdır, bu yazılım sisteminizdeki bağlantıları gözetlemenizi ve kontrol etmenizi sağlar.

Ağınız dibine dek açık mı? Bu tip analizler için şu iki temel yazılımdan faydalanabilirsiniz: netcat basit bir UNIX programıdır ve her türlü ağ bağlantısını istediğiniz TCP ya da UDP protokolü üzerinden kurmanızı sağlar.

nmap ise ağı keşfetmek ve güvenlik denetimi yapmak için kullanılan bir araçtır. nmap açık olan portları tarar.

Parola test etme yazılımları sisteminizdeki parolaların kalitesi konusunda size bilgi verir. Bu amaçla geliştirilmiş en meşhur yazılımlardan biri John the Ripper'dır. Daha detaylı bir test edici olarak SAINT yazılımına göz atabilirsiniz.

19.11 Her sistem yöneticisinin düzenli olarak takip etmesi gereken en önemli siteler:

CERT devlet tarafından finanse edilen ve Carnegie Mellon University tarafından işletilen bir merkezdir. İlgi odağı Internet güvenlik problemleridir. SecurityFocus tarafından barındırılan BugTraq da güvenlikle ilgili problemlerin düzenli olarak duyurulduğu en önemli listelerden biridir. Özel olarak güvenlik yönetimi ile ilgilenmiyorsanız bile bulisteye üye olmanız menfaatiniz icabıdır. Yapılan duyurular arasında sizin sisteminiz ile bağlantılı olanlar ilginizi çekebilir.

Linux güvenliği ile ilgili tavsiye edebilecek diğer sitelere gelince: Linux Security HOWTO, O'Reilly's Security Pageve tabii ki developerWorks'dan Security Zone.

Yazdırma konusu ile ilgili olarak LPRng anasayfası pekçok bilgi içerir. Tabii Printing HOWTO da değerli bir belgedir. Bu konu ile ilgili bir başka önemli kaynak da LinuxPrinting.org sitesidir. Spesifik yazıcılarla ilgili olarak Serial HOWTO belgesine danışabilirsiniz. USB kılavuzu da USB yazıcılarla ilgili bilgi verir.

Heterojen ağlarda Samba büyük bir yardımcıdır. Bu tür bir ortamda yazıcı servisleri ile uğraşacaksanız Samba ana sayfasını ve Samba HOWTO belgesini de okumakta fayda vardır. Burada iki filtreleme programı ele alınmıştır:

Magicfilter ve Apsfilter. Unutmayın ki bu programların bir dönüştürücüye ihtiyacı vardır (tercihen Ghostscript).

19.12 Etkileşimli, güvenli kabuk oturumu

Eskiden, ağ üzerinden etkileşimli bir oturum açmak istediğinizde telnet veya rsh kullanırdınız. Ancak ağ işlemlerinin popülarite kazanmasıyla bu araçların eskisi kadar uygun olmadığı ortaya çıktı. Çünkü telnet ve rsh kesinlikle güvensizdi. Sunucu ve istemci arasındaki bilgileri şifrelenmeden gönderiliyordu ve böylece ağa girebilen herhangi birisi tarafından kolaylıkla okunabiliyordu. Sadece bu kadarla kalmıyor, kimlik doğrulama işlemi için sunucuya yollana parola düz metin halinde gönderiliyordu, bu da ağ üzerinde data yakalayabilen herkes için sizin hesabınıza girmenin çok kolay olduğu anlamına geliyordu. Ayrıca bir ağ yoklayıcısı ile, tüm telnet oturumunuz tekrar oluşturulabilmesi ve sizin ekranda gördüğünüz herşeyin görüntülenebilmesi mümkündü. Sonuç olarak, ağların güvenli ve yoklanamaz olacağı varsayımıyla oluşturulmuş olan bu araçların, bugünkü çok dağılmış ve halka açık ağlar üzerinde kullanılması çok uygun olmadığı açık.

19.13 Güvenli kabuk

Daha iyi ve güvenli bir çözüme ihtiyaç doğmuştu. Bu çözüm güvenli kabuk (secure shell), bir diğer adıyla ssh oldu. Günümüzde çok popüler olan bu araç tüm Linux dağıtımlarında openssh paketi olarak bulunu-

yor. Ssh'ın, güvenli olmayan diğer araçlardan farkı, istemci ile sunucu arasındaki tüm iletişimi güçlü bir şifreleme mekanizmasıyla şifrelemesidir. Böylece istemci ve sunucu arasındaki iletişimi görüntülemek çok zor (hatta imkansız) hale geliyor. Bu da, güvenli kabuk ismini haketmiş olmasını sağlıyor. Ssh çok özellikli bir güvenlik yapısına sahiptir, bir şekilde ağa girilse ve sunucu-istemci iletişimi görüntülense bile parola doğrulama için şifrelemenin yanında bazı anahtar değişimi stratejileriyle, parolanın çalınmasını imkansız kılar. Internetin bu kadar popüler olduğu günümüzde, Linux sistemlerin kullanıldığı ağlarda, güvenliği üstü düzeye çıkarmak için ssh uygun bir araçtır. Güvenliğin önemini kavramış sistem yöneticileri, ağlarında telnet ve rsh kullanılmasını uygun bulmaz, hatta bazen yasaklarlar. Çünkü ssh aynı işi yapabilen ve çok daha güvenli bir alternatiftir.

19.14 ssh kullanımı

Çoğunlukla, tüm dağıtımlardaki openssh paketleri herhangi bir ayarlamaya ihtiyaç duyulmaksızın kullanılabilirler. openssh kurulunca birkaç ikilik dosyanız olacak. Bir tanesi sshd çalıştıran bir güvenli kabuk sunucusuna bağlanmanızı sağlayacak olan güvenli kabuk istemcisi: ssh. Ssh kullanmak ve bir oturum açmak için:

```
$ ssh knoppix@remotebox
```

yazarsınız. Burada remotebox makinesine knoppix kullanıcısıyla bağlanma komutu vermiş olduk. Telnet'teki gibi bir parola sorulacak, parolayı girdikten sonrada uzaktaki makinede sizin için yeni bir oturum açılmış olacak.

19.15 sshd başlatmak

Eğer makinenize ssh ile ulaşılmasına izin vermek isterseniz sshd sunucusunu çalıştırmanız gerekir. Bunun için openssh paketiyle gelen rc-script'i kullanmalısınız. Şöyle ki:

```
# /etc/init.d/sshd start
```

veya

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

Eğer gerekirse, sshd yapılanışınızı /etc/ssh/sshd_config dosyası üzerinden değiştirebilirsiniz. Daha fazla bilgi almak ve diğer tercihleri görmek için sshd kılavuz sayfasını kullanabilirsiniz:

```
man sshd
```

19.16 Güvenli kopya

Openssh paketi ayrıca çok kullanışlı bir araç daha içerir. Güvenli kopya (secure copy), scp, olarak adlandırılan bu araçla ağınız üzerinde kendi makinenizden başka makineye "veya ters yönde güvenli kopyalama işlemleri yapabilirsiniz. Örneğin, /foo.txt dosyasını remotebox makinesindeki home dizinime kopyalamak istersem:

```
$ scp ~/foo.txt knoppix@remotebox:
```

Parola girme ve doğrulama işleminden sora kopyalama yapılacaktır. Tersi içinde, örneğin remotebox makinesinin /tmp dizinindeki bar.txt dosyasını, yerel makinemde, halihazırda çalıştığım dizine kopyalamak istersem:

```
$ scp knoppix@remotebox:/tmp/bar.txt
```

yazarım.

19.17 Güvenli kabuk doğrulama ve yetkilendirme tercihleri

Openssh'ın farklı doğrulama yöntemleri vardır. Doğru kullanılmak şartıyla, uzak sistemlere parola girmeksizin de ulaşabilirsiniz. Bu ayarlar yukarıda bahsettiğimiz /etc/ssh/sshd_config dosyası üzerinden yapılabilir.

20 NFS

20.1 NFS'e giriş

Ağ Dosya Sistemi (The Network File System - NFS), yerel bir ağda, birbirine bağlı UNIX ve Linux sistemlerin, aralarında dosya paylaşımını sağlayan bir teknolojidir. NFS, Linux ve UNIX dünyasında uzun süredir bilinen ve yaygın olarak kullanılan bir sistemdir. NFS'in temel kullanımı, kullanıcıların, ağ üzerinde herhangi bir makinede oturum açtıklarında kendi home dizinlerine ulaşmalarını sağlamaktır. NFS sayesinde uzak makinelerin dosya sistemi ve ağaç yapılarının, yerel olarak oturum açılan makineye bağlanması (mount) ve tam uyumluluğu sağlanabiliyor. Bu yeterlilik ve basitlik, NFS'i, Linux dünyasında en popüler ve kullanışlı ağ dosya paylaşım sistemi haline getirmiştir.

20.2 Temel NFS bilgileri

NFS üzerinden dosya paylaşımı için önce bir NFS sunucu kurulması gerekir. Bu sunucu dosya sistemlerini ihraç etme özelliğine sahip olacak. Bir dosya sisteminin ihraç edilmesi, ağ üzerindeki diğer sistemler tarafından kullanılabilir hale gelmesi anlamındadır. Böylece, NFS istemci olarak kurulmuş bir sistemden, bu ihraç edilen dosya sistemini standart "mount" komutuyla bağlayabilirsiniz. Bağlama işlemi tamamlanınca, uzaktaki sistem, yerel makineye bağlanmış herhangi bir sistem gibi (örneğin /mnt/cdrom) kullanılabilecektir. Bu yapıda, yerel sisteme bağlanan dosya sisteminin bir disk'ten değil, sunucudan alındığına dikkat edin. Bu durum Linux sistemlerinin zaten uyumlu olduğu bir uygulamadır, bu yüzden çoğu zaman sorunsuz çalışır.

20.3 NFS'in özellikleri

NFS dosya paylaşım sisteminin bazı ilginç özellikleri vardır. Bu özelliklerden ilki NFS'in "başına buyruk" tasarımıdır. İstemcilerin sunucuya bağlantıları, kendi başına olduğundan, NFS sunucusundaki herhangi bir nedenle kapanıp-açılma durumu, istemcilerin bağlantısında bir kopmaya sebep olmaz. Sadece geçici olarak duraklatılan uzak NFS dosya bağlantıları, sunucu tekrar eski haline geldiğinde, kaldığı yerden devam edecektir. Yine bu başına buyruk özellik sayesinde, bir sunucu, çok sayıda istemciyi, ek bir yük olmaksızın, rahatlıkla kaldırabilir. Fazla istemci olmasının getireceği tek yük, ağ üzerinde dosya transferleri sırasında ortaya çıkan yoğunluk olacaktır. Özetle şunu söyleyebiliriz ki, NFS'in performansı, ağ üzerindeki istemci sayısına değil, veri transferi yoğunluğuna bağlıdır.

20.4 Linux'ta NFS, sürüm 3

NFS kurulacağı zaman sürüm 2 yerine sürüm 3'ü tercih etmenizi öneririz. Sürüm 2'nin, dosya kilitleme ve bazı süreçlerde takılma konusunda olumsuz bir şöhreti vardı. Ama bu sorunlar sürüm 3'te çözüldü. Bu sürüm sağlam ve başarılı kabul ediliyor. Linux 2.2.18 ve üzeri çekirdekler NFS 3 sunucu ve istemcilerini destekliyorlar. Dolayısıyla NFS 3 varken, artık NFS 2 kullanmak için herhangi bir sebep yok.

20.5 NFS'te güvenlik

NFS sürüm 2 ve sürüm 3 te güvenlik nedeniyle bazı kısıtlamalar olduğunu belirtelim. NFS, özel olarak, güvenli bir ağ üzerinde çalışmak üzere tasarlanmış bir sistemdir. Yerel ağınızda güvenli olarak NFS kullanmak istiyorsanız, sunucu makinesinin root parolasının sadece ve sadece sistem yöneticilerinde olması gerekir. Eğer sıradan bir kullanıcı, root parolasına sahipse, NFS güvenlik duvarını kolayca aşabilir ve sunucu üzerinde, normalde ulaşamayacağı dosyalara ulaşarak sistem güvenliğini tehdit edebilir. Bu yüzden NFS sistemi plansız, gelişi güzel kurulmamalıdır. Yerel ağınızda NFS kullanmaya karar vermeniz çok iyi ama önce sunucunuz için bir kalkan (firewall) kurmalısınız. Ayrıca yerel ağınız dışından NFS sunucunuza ulaşımın mümkün olmadığından emin olun. Sonrasında da, yerel ağınızın güvenli olduğundan emin olup, bu güvenlik kontrollerini sık sık yapmalı ve yeri geldikçe geliştirmelisiniz. Artık güvenli olarak NFS kullanmaya hazırsınız.

21 NFS kurulumu

NFS 3 kurulumunun ilk adımı bir NFS sunucusu kurmaktır. Sunucu olarak seçilen makinenin çekirdeğinde NFS sunucu desteğini aktif hale getirmek gerekir. NFS 3 için 2.2.18 ve üzeri bir çekirdeğe (2.4 ve üzeri önerilir) sahip olmalısınız. Eğer çekirdeğinizi kendiniz derliyorsanız, /usr/src/linux dizinine gidip

```
# make menuconfig
```

komutunu çalıştırmalısınız. Burada "File systems section" seçeneğini, ardından da "Network File Systems" seçeneğini seçtikten sonra, aşağıdaki tercihleri aktif hale getirin:

```
<*> NFS file system support
[*]   Provide NFSv3 client support
<*> NFS server support
[*]   Provide NFSv3 server support
```

21.1 /etc/exports üzerindeki düzenlemeler

Çekirdeğinizi derleyip kurduktan sonra, bilgisayarınızı yeniden başlattığınızda sisteminiz NFS 3 sunucu ve istemci özelliklerine sahip olacaktır. Çekirdekte NFS desteğini sağladıktan sonra, sıra /etc/exports dosyasını düzenlemeye geldi. /etc/exports dosyası üzerindeki düzenlemeler, hangi dosya sistemlerinin ihraç edilebileceğini, bu dosya sistemlerine kimlerin ulaşabileceğini ve bu kullanıcıların okuma/yazma veya sadece okuma haklarıyla ulaşabileceklerini belirler. Ayrıca NFS'in çalışmasıyla ilgili bazı diğer ayarları da buradan kontrol edebiliriz.

/etc/exports dosyasını incelemeden önce, varolan bir problemden bahsedelim. Linux çekirdeğindeki NFS sistemleri, her bir dosya sisteminde tek bir dizinin ihraç edilmesine izin verir. Yani aynı ext3 dosya sisteminde (örneğin /dev/hda6 da) bulunan /usr ve /home dizinlerini birlikte ihraç edilmeleri için /etc/exports dosyasına yazamazsınız. Eğer her iki klasörü de /etc/exports dosyasına eklerseniz, bu dosya tekrar okunduğunda (örneğin NFS sunucu çalışırken exportfs -ra yazarsanız) aşağıdaki hatayı görürsünüz:

```
sidekick:/home: Invalid argument (Geçersiz değişken)
```

21.2 İhraç kısıtlamaları ile çalışma

Şimdi bu problemi nasıl çözeceğimize, yani hem /usr hemde /home klasörlerini aynı anda nasıl ihraç edeceğimize bakalım. Her iki dizinde aynı dosya sistemindeyse birlikte ihraç edemeyeceğimizi söyledik. O zaman / (kök) dizinini ihraç ederiz. Böylece NFS istemcileriniz artık hem /home hemde /usr dizinlerini NFS üzerinden bağlayabilirler. Ayrıca NFS sunucusundaki /etc/exports dosyası da her bir dosya sistemi için tek bir dizin ihraç edildiği için hata vermez. Linux NFS'in bu kurulumunu da gördükten sonra /etc/exports dosyasına bakabiliriz.

```
/etc/exports dosyası
```

/etc/exports dosyasının yapısını anlamanın en iyi yolu bir örneğini incelemektir. Bizim NFS sunucusundaki /etc/exports dosyasının içeriği şöyle:

```
# /etc/exports: NFS file systems being exported.
/ 192.168.1.9(rw,no_root_squash)
/mnt/backup 192.168.1.9(rw,no_root_squash)
See exports(5).
```

Görüldüğü gibi dosyanın ilk satırı bir açıklama. İkinci satırda kök ("/") dizinin ihraç edildiği belirtiliyor. Şunu unutmayın ki, kök dizin ihraç edildikten sonra aynı dosya sisteminde ki bir başka dizin ihraç edilemez. Örneğin NFS sunucumun /mnt/cdrom klasörüne bağlanmış bir CD sürücü varsa, bu sürücü /etc/exports'ta ayrıca bağlanmadığı sürece ulaşılamazdır. Bunun için dosyanın üçüncü satırına bakalım. Burada kök dizinden başka bir dosya sistemine ait olan ve sistemin yedeğinin tutulduğu /mnt/-backup dizini ihraç ediliyor. Her satırdaki "192.168.1.9(rw,no_root_squash)" bilgisi de, nfsd'ye, sadece 192.168.1.9 IP adresli istemcilerin, ihraç edilen dizinlere ulaşabileceğini ve bu istemcilerin okuyabildikleri gibi yazma haklarına da sahip olacaklarını söyler. Ayrıca bir NFS istemcisinde super kullanıcı hesabıyla dosya sistemine root haklarıyla ulaşmasına izin veriliyor.

21.3 Bir başka /etc/exports örneği

Buradaki /etc/exports dosyası da bir önceki örnekteki dosya sistemini ihraç etmek için oluşturulmuş. Ancak bir fark, buradaki sunucu, LAN üzerindeki (192.168.1.1 den 192.168.1.254 e kadar) tüm makinelere hizmet veriyor.

```
# /etc/exports: NFS file systems being exported.
/ 192.168.1.1/24(rw,no_root_squash)
/mnt/backup 192.168.1.1/24(rw,no_root_squash)
See exports(5).
```

Bu /etc/exports örneğinde, /24 maskesi kullanılarak, verilen IP adresindeki son 8 bit maskeleniyor. Burada IP adresi ifadesiyle "(" arasında boşluk olmamasına dikkat edin. Bu araya koyacağınız bir boşluk karakteri dosyanın yanlış yorumlanmasına sebep olur. "rw" ve "no_root_squash" dışında da bazı izin tercihleri vardır. Bunun için "man exports" yazarak tüm tercihleri görebilirsiniz.

21.4 NFS3 sunucusunu çalıştırmak

/etc/exports dosyası düzenlendikten sonra, NFS sunucunuzu çalıştırmaya hazırsınız demektir. Çoğu dağıtımda nfs çalıştırma komut dosyaları hazır geliyor. Yazmanız gereken:

```
/etc/init.d/nfs start
```

veya

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

NFS çalıştırıldıktan sonra, rpcinfo yazdığınızda aşağıdakine benzer bir çıktı almanız gerekiyor.

```
# rpcinfo -p
program vers proto port
100000 2 tcp 111 portmapper
100000 2 udp 111 portmapper
100024 1 udp 32768 status
100024 1 tcp 32768 status
391002 2 tcp 32769 sgi_fam
100011 1 udp 792 rquotad
100011 2 udp 792 rquotad
100011 1 tcp 795 rquotad
100011 2 tcp 795 rquotad
100003 2 udp 2049 nfs
100003 3 udp 2049 nfs
100021 1 udp 32770 nlockmgr
100021 3 udp 32770 nlockmgr
100021 4 udp 32770 nlockmgr
100005 1 udp 32771 mountd
100005 1 tcp 32770 mountd
100005 2 udp 32771 mountd
100005 2 tcp 32770 mountd
100005 3 udp 32771 mountd
100005 3 tcp 32770 mountd
100009 1 udp 821 yppasswdd
100004 2 udp 734 ypserv
100004 1 udp 734 ypserv
100004 2 tcp 737 ypserv
100004 1 tcp 737 ypserv
```

21.5 İhraç tercihlerini değiştirme

Eğer NFS servisi çalışırken bazı değişiklikler yaparsanız, bu değişikliklerin aktif olması için

```
exportfs -ra
```

yazmanız gerekir. Böylece NFS sunucunuz yeniden düzenlenmiş ve çalışır hale gelmiştir. Artık NFS istemcilerini yapılandırarak, ihraç ettiğiniz dosya sistemlerini bağlamalarını sağlayabiliriz.

21.6 NFS istemcilerini yapılandırma

NFS 3 istemcileri için çekirdek yapılandırılması NFS sunucu ile çok benzer. Tek fark, sadece aşağıdaki tercihlerin seçilmiş olması yeterlidir.

```
<*> NFS file system support [*] Provide NFSv3 client support
```

21.7 NFS istemci servislerinin çalıştırılması

NFS istemci servislerini başlatmak için, sunucudaki gibi bazı hazır komut dosyalarını, "nfslock" veya "nfsmount", çalıştırmak yeterlidir. Bu komutlar NFS istemcinin tek ihtiyacı olan rpc.statd servisini başlatacak. Bu servis, dosya kilitleme sisteminin doğru çalışmasını sağlar. Tüm istemci servisleri kurulduktan sonra, rpcinfo yazarsanız, aşağıdakine benzer bir çıktı alırsınız:

```
# rpcinfo
    program  vers  proto  port
     100000     2    tcp   111 portmapper
     100000     2    udp   111 portmapper
     100024     1    udp 32768 status
     100024     1    tcp 32768 status
```

Bu kontrolü uzaktaki bir sistemden yapmak isterseniz, komutu rpcinfo -p makine_adi parametreleriyle çalıştırmak gerekir:

```
# rpcinfo -p sidekick
   program vers proto port
    100000    2 tcp    111 portmapper
    100000    2 udp    111 portmapper
    100024    1 udp  32768 status
    100024    1 tcp  32768 status
```

21.8 İhraç edilen NFS dosya sistemlerini bağlama

Sunucu ve istemciler kurulduktan sonra ve NFS sunucusu, istemcinizden gelen bağlantılara izin vermek üzere yapılandırıldıysa, bir sonraki aşamaya, ihraç edilen bir dosya sistemini istemci makineye bağlama işlemine geçebilirsiniz. Aşağıdaki örnekte, inventor makinesi NFS sunucusu, sidekick ise (IP adresi 192.168.1.9) NFS istemcisi olarak kurulmuş. Inverntor'ın /etc/exports dosyasında aşağıdaki satır yer alıyor.

```
/ 192.168.1.1/24(rw,no_root_squash)
```

Böylece, 192.168.1 ağındaki tüm makinelere bağlantı izni verilmiş. Şimdi sidekick makinesine root olarak girin ve aşağıdakileri yazın:

```
# mount inventor:/ /mnt/nfs
```

Inverntor makinesinin kök dizini, sidekick makinesinde /mnt/nfs dizinine bağlanmış oldu.

```
cd /mnt/nfs
```

yazarak, inventor makinesinin dizinlerinde gezebilir ve dosyalarına bakabilirsiniz. Şunu tekrar hatırlatalım ki, inventor makinesinin kök ("/") dizini başka bir dosya sisteminde olsaydı, /mnt/nfs/home dizini hiçbirşey içermeyecekti. Bu durumda, buradaki bilgiye ulaşmak için yeni bir bağlama işlemine (aynı zamanda inventor makinesinin /etc/exports dosyasında yeni bir satıra) ihtiyaç duyacaktık.

21.9 İhraç edilen dizinlerin içinde yapılan bağlama işlemleri

/etc/exports dosyasındaki / 192.168.1.1/24(rw,no_root_squash) satırının kök dizini içindeki dizinlere de bağlama işlemi uygulamamıza izin verdiğini unutmayalım.

Örneğin, inventor makinesinde kök ("/") dizin ile /usr dizinleri aynı dosya sistemindeyse ve siz / yerine /usr dizinini bağlamak isterseniz, o zaman:

```
# mount inventor:/usr /mnt/usr
```

yazarsınız. Böylece inventor'un /usr dizin ağacı, sizin önceden varolan /mnt/usr dizininize bağlanmış oldu. Bunun için /etc/exports dosyasında bir değişikliğe ihtiyaç olmadığına dikkat edin. Kök dizininin ihraç edilmesi, onun altındaki dizinlere bağlama işlemi yapılmasını özgür bırakmıştı.