BÖLÜM 11

UNIX
KULLANIMI

Çok kullanıcılı bir işletim sistemi olan UNIX ortamında çalışmak için her şeyden önce sistem üzerinde tanımlanmış bir kullanıcı olmak gerekir. Ancak, önceden tanımlanmış kullanıcılar sisteme giriş yapabilir. UNIX ilk yüklendiğinde *root* adlı kullanıcı tanımı vardır; yönetici yetkilerine sahip olan bu kullanıcı, sistem üzerinde herşeyi yapmaya yetkilidir: yeni kullanıcılar oluşturabilir, yeni programlar yükleyebilir...; var olan kullanıcı haklarını belirleyebilir.

Sisteme Girmek

UNIX'li sisteme birkaç şekilde bağlanılır ve orada sunulan ortam kullanılabilir:

❶ Kendisine doğrudan bağlı terminal üzerinden
❷ Uzaktaki bir bilgisayarda terminal emülasyonu koşturularak
❸ TCP/IP'nin *telnet* protokolu kullanılarak

Hangi şekille bağlanılırsa bağlanılsın, sisteme giriş yapmak için aşağıda gösterildiği kullanıcı adı (*userid:*) ve şifresi (*password:*) sorulur:

```
login :
password :
```

Ancak bu bilgiler doğru girildiği zaman sisteme giriş izni verilir ve kullanıcı kendisine ait olan çalışma alanına girer.

Komut İşareti

Sisteme doğru olarak giriş yapıldığı zaman kullanıcının karşısına aşağıda görüldüğü gibi bir komut işareti gelir ($, % vs.); ancak bu işaret varsa komut (veya komutlar) girilebilir, kullanılabilir.

```
% _
```

Komut işareti görünmediği zamanlar, yürütülmesi için komut girilmemelidir; kontrol, o anda başka bir moddadır.

Sisteme giriş yapıldıktan sonra hangi çalışma alanında bulunulduğu görülmek istenirse, `pwd` komutu aşağıdaki gibi kullanılır:

```
% pwd ↵
user/batuhan

% _
```

Buradan kullanıcının *batuhan* olduğu öğrenilir; batuhan'ın o andaki çalışma alanı ise *user/batuhan*'dır.

Yazım Hatasını Düzeltmek

Komut yazılırken, yazım hatası yapıldığında klavye üzerinde bilinen tuşlarla düzeltme yapılabilir; ancak bazı sistemlerde bu tuşlar geçersizdir. Bu durumda kontrol karakterleri kullanılarak düzeltme yapılabilir:

#	İmlecin bir solundaki karakteri siler; ard arda 3 kez basılmış ise, 3 karakter silinmiş varsayılır. Ancak, silinen karakterler olduğu yerde görülür...
@	Yazılan tüm karakterleri iptal eder.

```
% pwwd##d ↵
user/batuhan
```

Sistemden Çıkış

Bir kullanıcının sistemden çıkması için Ctrl+D tuşlarına basması yeterlidir. Bu durumda, o kullanıcı için ayrılan standart sistem kaynakları serbest bırakılır ve kullanıcının başlattığı prosesler sonlandırılır.

```
Ctrl+D
```

Sistemi Kapatmak

UNIX'li sistem tamamen kapatılmak isteniyorsa, ilk önce sisteme girmiş olan kullanıcılar uyarılır ve sistemden çıkmaları istenir. Sistem doğrudan güç kaynağı kesilerek kapatılmaz! Aksi durumda dosya sistemi zarar görebilir. Bu nedenle, güç kaynağı kesilmeden önce, dosya sistemi güncellenmeli ve açık olan dosyalar, giriş çıkış birimleri varsa kapatılmalıdır; sistemle bağlantısı kesilmelidir. Bunları yapacak komutları yürütmek için yönetici yetkilerine sahip olunmalıdır!

Sisteme bağlanmış dosyalar `unmount` komutu ile sistemden ayrılır; ancak daha öce `sync` komutu ile dosya sistemi güncellemesi yapılmalıdır.

> **Not** : UNIX dosya sistemi, çalışırken güçlü ve güvenilir bir ortam sunar; sistem kilitlenmelerine yol açmaz. Ancak beklenmeyen kesintiler, beklenmedik şekilde sistem kapatılmaları durumunda bozulabilir. Verilerin güvenliği açısından dosya sisteminin yedeklenmesi belirli aralıklarla yapılmalıdır.

Çokprosesli Sistem

UNIX çok kullanıcılı ve çokprosesli bir sistemdir. Aynı anda birçok kullanıcı sisteme girebilir ve kendilerine ait programları çalıştırabilir. Programlar UNIX'te proses bazında çalışır; bir program bir veya birden çok proses yaratabilir. Herhangi bir anda, sistem üzerinden birçok proses çalışır (aktif) durumdadır. Herhangi bir program durdurulmak istenirse, ona ait proses sonlandırılır. Bunun için önce proses'in numarası öğrenilmelidir; ardından `kill` komutu ile proses sonlandırılır.

a.out Dosyası

Bu dosya özel bir işleve sahiptir. Şöyle ki, bazı komutlar yürütülürken, bir dosyaya sonuç yazılması gerekir. Eğer, sonucun hangi dosyaya yazılacağı belirtilmemiş ise, *a.out* dosyasına yazılır. Bir çeşit *default* çıkış dosyasıdır.

Komut Yapısı

UNIX komutları çok geniş kapsamlıdır; komutların bir adı ve genel olarak parametreleri vardır. Parametre, komutun işlevini yerine getirmesi için kullanılan değerdir. Bazı komutlar, yalın olup, yalnızca adının yazılıp Enter (↵) tuşuna basılması yeterlidir.


```
% komutAdı   parametre1  parametre2 ...  ↵
```

Komut adı ile parametreler birbirlerinden boşluk karakteri ile ayrılırlar. Komutlar, izin verilen ölçüde kullanılır. Özellikle sistem komutları için sistem yöneticisi haklarına sahip olunması gerekir. Satırın en sonuna & karakteri yazılırsa, komuta ait proses arka alanda yürütülür.

Komut Yorumlayıcı *Shell*

UNIX'te, kısaca *sh, csh, bash, ksh* olarak adlandırılan birçok komut yorumlayıcı vardır; üst düzey kullanımda (derinlemesine olmayan) aynı gibi görünse de alt düzey kullanımlarda farklılıklar vardır. Kullanıcı, sisteme ilk girdiğinde bunlardan biri geçerlidir; istenirse çalışırken değiştirilebilir. Bunun için komut satırından yorumlayıcı kısa adı yalın olarak girilir.

```
% csh  ↵
```

İlk girildiğinde hangi komut yorumlayıcının koşacağı kullanıcı hesabı açılırken belirtilir.

▪ Dosya Komutları

Dosyalar üzerinde işlem yapan komutlardır; parametrelerden en az bir tanesi dosya adıdır. Dosya yaratmak, silmek, kopyalamak, içerisinde birşeyler aramak vs. gibi işlerin kotarılması için kullanılır.

Çizelge-11.1. UNIX dosya komutları

Dosya listeleme	`ls`	bkz. sayfa 175
Dosya içeriklerini görme	`more`	bkz. s.176
Dosya kopyalama	`cp`	bkz. s.176
Dosya silme	`rm`	bkz. s.176
Dosya adını değiştirme	`mv`	bkz. s.176
Erişim izni değiştirme	`chmod`	bkz. s.176
Dosya Karşılaştırma	`cmp, diff`	bkz. s.178
Dosya tipini sorgulama	`file`	bkz. s.178
Dosya içinde sözcük arama	`grep`	bkz. s.178
Dosya içindekileri sıralama	`sort`	bkz. s.179
Dosya içeriği istatistiklerini öğrenme	`wc`	bkz. s.179

Dosya dökümü almak	cat	bkz. s.180
Dosya Parçalama	split	bkz. s.180
Dosya içeriğini parçalama	cut	bkz. s.181
Dosya kayıtlarını birleştirme	paste	bkz. s. 182
Dosya satırlarına numara verme	nl	bkz. s.182
Tekrarlanan satırları silme	uniq	bkz. s.182
Karakter dönüşümü yapma	tr	bkz. s.183
Dosyanın belirli kısmını görme	tail, head	bkz. s.183
Dosyaları ilişkilendirme	ln	bkz. s.183
Dosya yerini bulma	find	bkz. s.183

Dosyaların Listelenmesi ls

ls komutu alan içindeki dosya ve altalanları listeler; hiçbir parametre verilmeden kullanılırsa çalışma alanı içindeki dosya ve altalan isimleri listelenir. Daha ayrıntılı listeleme istenirse yanına parametre verilir:

```
% ls ⏎        (yalnızca dosya adları listelenir)

total 37
oldu  duyuru proje  deney.c     network   ...
```

Daha ayrıntılı liste görmek için,

```
% ls -l ⏎

total 37
 drwxr-xr-x  3   Oguzhan  147  64   Jun 13  15:30  oldu
 -rwxrwxrwx  1   Oguzhan  147  32   Sep 12  16:02  duyuru
 -r--r--r--  1   Oguzhan  147  471  Feb 15  12:05  proje
 -rwx------  1   Oguzhan  147  32   Feb 13  14:22  deney.c
 -rwx--x--x  1   Oguzhan  147  232  Feb 11  9:52   network
 ...
```

Yukarıdaki örnekte *–l* seçeneği verilerek daha ayrıntılı liste alınmıştır; *-d* ile yalnızca alanlar, *-t* ile işlem tarihine göre sıralanarak, *-s* ile işgal ettiği belleğin blok sayısını vererek, *-u* ile son erişim tarihine göre sıralayarak listeler.

Dosya İçeriklerini Görme more

more, dosya içeriğini ekranda sayfa sayfa gösterir; ekran dolduğunda okunması için bekler. Boşluk tuşu bir sonraki sayfaya atlatırken, *q* tuşu göstermeyi sonlandırır.

```
% more  proje ⏎
% more  /usr/begum/proje ⏎
```

Dosya Kopyalama cp

cp, bir dosyanın kopyasını oluşturmak için kullanılır. Dosya aynı alan içerisinde kopyalanacak ise kopyaya yeni isim verilmelidir!

```
% copy  index.htm   index2.htm ⏎
% copy  index.htm   /etc/webroot ⏎
```

Dosya Silme rm

rm komutu dosya silmek için kullanılır. Silinen dosyalar bir daha elde edilemez.

```
% rm   index.htm ⏎
% rm  /bin/yedek ⏎
```

Dosya Adını Değiştirme mv

mv, dosya adını değiştirmek veya dosyayı bir altalandan bir başka altalana taşımak için kullanılır.

```
% mv   index.htm  ana.html ⏎
% mv  ana.html   /etc/webroot/htmpage ⏎
```

Dosya Erişim İznini Değiştirme chmod

Dosya erişim izinleri dosya sahipleri tarafından chmod komutuyla, değiştirilebilir. Örneğin *batuhan* adlı kullanıcı *proje* adlı dosyasının erişim iznini değiştirmek istesin: yeni erişim izni, kendisi için hem okuma hem yazma,

grup üyeleri ve diğer kullanıcılara yalnızca okuma izni vermek istesin bu durumda chmod komutunu aşağıdaki gibi yürütülebilir:

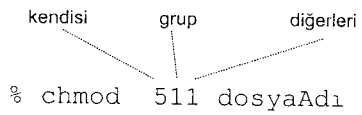
```
% chmod  644 proje ⏎
```

İzin verme işleminde kullanılan parametre, yukarıdaki örnekte olduğu gibi hem sayısal hem de simgesel verilebilir.

Sayısal izin verme:
3 haneli 8 tabanında bir sayı verilir; ilki sahibine, ikincisi grubuna ve üçüncüsü de diğer kullanıcılara verilecek izni gösterir. Yukarıdaki örnekte 6 sahibine 4 grup üyelerine ve ikinci 4 ise diğer kullanıcılara verilecek izni belirtir. Bu sayısal değerler aşağıdaki çizelgeden verilen sayılardan bulunabilir.

Çizelge-11.2. Erişim izni hesaplama sayıları ve simgeleri

İzin Türü	Sayısal	Simgesel
Okuma (read)	4	r
Yazma (write)	2	w
Çalıştırma (excute)	1	x



Aynı anda, hem okuma, hem yazma hem de yürütme izinleri değiştirilebilir. Bunun için karşılıkları olan sayılar toplanır. Örneğin, 4, 2 ve 1 sayıları toplandığında 7 yapar ve tüm izinlerin verildiğini gösterir; yalnız başına 1 sayısı yalnızca çalıştırma izni verildiğini belirtir:

İzinler değiştirilirken, simgesel yazım biçimi de kullanılabilir. Çizelge-11.2.'de görüleceği gibi *r*, *w* ve *x* izin verme karakterleri olarak kullanılır; u (user), g (group) ve o (others) karakterleri de sırasıyla sahip, grup ve diğerleri anlamında kullanır; a (all) karakteri ise her üçü için demektir. Simgesel yazım formatı (+ : izin verme, - : izin kaldırma):

```
% chmod  {a u g o} {+ - } {r w x }  dosyaAdı ⏎
```


Örnekler:

```
% chmod  a+r  dosyaAdı ⏎      herkes okuyabilir.
% chmod  og-x dosyaAdı ⏎  diğerleri ve grup üyeleri çalıştıramaz.
% chmod  u+rwx  dosyaAdı ⏎  sahibi herşeyi yapabilir.
% chmod  o-rwx   dosyaAdı ⏎  diğerlerine hiçbir izin yok.
```

Dosya Karşılaştırma cmp veya diff

İki dosyayı karşılaştırmak için cmp veya diff komutu kullanılır. cmp komutu iki dosya arasındaki farkı belirlemeye yararken, diff komutu iki dosyayı satır karşılaştırmak için kullanılır. Parametre verilerek raporlama şekli değiştirilebilir.

```
% cmp  [-option]  dosya1  dosya2
% diff  [-option]  dosya1  dosya2

% cmp   proje  network ⏎
% diff   proje  veri.c ⏎
```

Dosya Tipini Sorgulama file

Dosya türünü sorgulayıp raporlamak için file komutu kullanılır; dosyanın ne tür veri içerdiğini belirler. Dosya adı yerine * karakteri yazılırsa, çalışma alanındaki tüm dosyaların türünü döker.

```
% file * ⏎
programs: directory
proje: ascii text
deneme: empty
```

Dosya İçinde Sözcük Arama grep

grep, bir sözcüğün dosya içerisinde geçip geçmediğini aramak için kullanılan komuttur; bulduğu satırı standart çıkışa (ekrana) aktarır.

```
% grep  [-option]  sözcük  [dosya(lar)]
% grep   main()   veri.c ⏎
```

Aramada büyük ve küçük harf farklı kabul edilir; aynı kabul edilmesi için *–i* parametresi kullanılır.

```
% grep -i  GAP  proje ⏎
```

Sonucun başka bir dosyaya yazılması istenirse, yönlendirme karakteri (>) kullanılabilir:

```
% grep GAP  proje > sonuc ⏎
```

Sözcüğün geçtiği satır numaralarının da gösterilmesi istenirse, *-n* parametresi kullanılır:

```
% grep   main()   veri.c ⏎
```

Dosya İçindekileri Sıralama `sort`

Dosya içindekilerin alfabetik olarak sıralanması için kullanılır. Sıralanan dosya eski halini korur; sonucun saklanması isteniyorsa *sonuc* dosyası belirtilmelidir:

```
% sort   personel  >  sonuc ⏎
```

Sıralama işlemi satır düzeyinde veya satır içindeki alan (field) düzeyinde yapılabilir. Satır düzeyinde yapılması isteniyorsa yukarıdaki gibi yazılması yeterlidir. Alan düzeyinde yapılması istenirse, alanları birbirinden ayıran karakter (-t ile " " arasında) ve alan sayısı (+ ile) belirtilmelidir (alanlar soldan sağa 0, 1, 2... diye sıralanır; alan, aslında sütun gibi değerlendirilebilir). Örneğin alan ayırma karakteri : olan satırları 2'nci (numarası 1) alana göre sıralamak için `sort` komutu aşağıdaki gibi kullanılır:

```
% sort  -t":"  +1   personel  >  sonuc ⏎
```

Ters yönde sıralama yapmak için –r parametresi kullanılır.

Dosya İçeriği İstatistiklerini Öğrenme `wc`

Dosya içeriği hakkında satır, sözcük, sayfa sayısı gibi istatistiksel bilgileri hesaplar ve rapor eder:

```
% wc  proje ⏎
```


Dosya Dökümü Almak `cat`

`cat` komutu dosya içeriğini görmek, yeni dosya oluşturmak ve dosyaları birbirine eklemek gibi birkaç değişik amaçla kullanılır. Yeni bir dosya oluşturmak için aşağıdaki gibi yazılabilir; standart girişten (klavye) girilenleri *komutum* adlı dosyaya yazar.

```
% cat  > komutum ⏎
ls -l | grep alican
echo "aradim:"
^D
```

Dosya içindekileri görmek için doğrudan komut adı ve dosya adı aşağıdaki gibi yazılır:

```
% cat  proje ⏎
```

Dosyaları birbirine eklemek için, aşağıda görüldüğü gibi eklenecek dosyalar aralarında boşluk karakteri bırakılarak yazılır ve yönlendirme karakteri (>) ile *sonuc* dosyası belirtilir (Ekleme dosyaların yazıldığı sırada yapılır):

```
% cat   proje1 proje2  >  pro ⏎
```

Dosya Parçalamak `split`

Büyük boyutlu dosyaların parçalanıp küçük dosyalar olarak saklanmasını sağlar; parçaların boyları için *default* değerler vardır; ancak parametre verilerek değiştirilebilir:

```
% split  pro parcalar⏎
```

Yukarıdaki gibi yazılırsa `pro` adlı dosya parçalara ayrılır; her bir parçaya *parcalaraa*, *parcalarab*, *parcalarac*,... *parcalarzz* gibi isimler verilir. Parçaların saklanacağı dosya adı verilemez ise *x* adı verilmiş kabul edilir ve parçalar *xaa*, *xab*, *xac* şeklinde isimlendirilir.

Parçalama yapılırken herbir parçanın kaç satırdan oluşması istendiği parametre verilerek belirtilebilir: *-n* satır, *-b* blok sayısını belirtir. Örneğin aşağıdaki örneklerin birincisinde, herbir parça 12 satır, ikincisinde 512 blok büyüklüğünde olur.

```
% split  -12n  pro parcalar⏎
% split  -b  pro parcalar⏎
```

Dosya İçeriğini Alan Düzeyinde Parçalama `cut`

Dosya içerisine sütunlar (alanlar) şeklinde girilmiş verinin, istenen bir veya birkaç sütunu üzerinde işlem yapma, listeleme, kopyalama gibi işlemler `cut` komutu ile gerçekleştirilir. Genel yazım biçimi,

```
% cut  - cSütunBilgileri   dosyaAdı␣
% cutt  -fSütunBilgileri -d SütunAyıraçı dosyaAdı␣
```

şeklindedir; *-c* parametresi karakter, *-f* alan düzeyinde işlem yaptırır. Örneğin soldan *2* ve *5nci* karakter arası listelenmek istenirse,

```
% cut   -c2-5   dosyaAdı ␣
```

şeklinde kullanılır. Yalnızca *2nci* ve *5nci* karakterlerin olduğu sütunlar listelenmek istenirse *2-5* yerine *2, 5* yazılır.

Alan düzeyinde işlem yaptırmak için *–f* parametresi kullanılır; *-d* ile alanları birbirinden ayıran karakter belirtilir (belirtilmez ise ayıraç karakteri boşluktur):

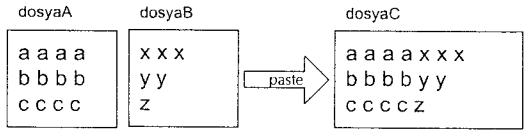
```
% cut   -f2-4   dosyaAdı ␣
% cut   -c2-4  -d:   dosyaAdı ␣
% cut   -c7,9   -d,   dosyaAdı ␣
```

Yukarıdaki birinci örnekte, ayıraç karakteri ' ', 2 ve 4 sütunlar arası; ikincide, ayıraç karakteri ':', 2 ve 4 sütunlar arası; üçüncü örnekte ise, ayıraç karakteri ',', yalnızca 7nci ve 9uncu sütunlar listelenir.

Not: Sonucunu sütunlar halinde veren komutların hemen arkasından `cut` komutu (| ile) kullanılarak istenen alanların listelenmesi sağlanabilir.

Dosya Kayıtlarını Birleştirme `paste`

İki dosya içerisinde sütunlar halinde bulunan verileri birleştirmek için kullanılır. Aynı satıra ait alanlar yanyana getirilir ve yeni bir dosya elde edilir.




```
% paste  dosyaA  dosyaB  > dosyaB  ␣
```

Dosya Satırlarına Numara Verme `nl`

`nl` komutu ile dosya içeriğine satır düzeyinde numara verilir; aşağıdaki örnekte deney adlı dosya satırları 1, 3, 5 diye sıralanır *(-v*: başlangıç sayıyı, *-i* artımı gösterir).

```
% nl -v1 -i2 deney ␣
```

Tekrarlanan Satırları Silme `uniq`

Dosya içerisinde ard arda tekrarlanan satırları arındırmak için kullanılır. İlk satır dışındakiler temizlenir. Aynı satırları belirlemek için yapılan karşılaştırma işlemi, *n* karakter veya *m* sütun içeriden başlatılabilir; böylelikle ilk *n* karakteri veya ilk *m* sütunu aynı olmayan satırlarda benzer sayılır.

```
% unig   dosyaAdı > sonucDosya ␣

% unig   +n dosyaAdı > sonucDosya ␣

% unig  -m dosyaAdı  > sonucDosya ␣
```

Not: Tekrarlanan satırları ard arda olmayan dosyaların fazlalıkları `unig` komutuyla temizlenemez; daha önce `sort` komutuyla sıralama yapılıp daha sonra `unig` komutu kullanılırsa temizlenebilir.

Karakter Dönüşümü Yapma `tr`

Dosya içeriği üzerinde değişiklik yapmak için kullanılır. Örneğin belirli karakterleri büyük harf yapmak, silmek, bir başkasıyla değiştirmek gibi işlemler `tr` komutu ile gerçekleştirilebilir. Aşağıdaki örneklerden ilki küçük harfleri büyük harfe dönüştürür; ikincisi, *istanbul* sözcüğünü dosyadan siler:

```
% tr  "[a-z]"  "[A-Z]"  < ilkDosya  > sonuc ⏎
% tr  -d  "istanbul"  < ilkDosya  > sonuc ⏎
```

Dosyanın Belirli Kısmını Görme `tail, head`

Bu iki komut uzunca dosyaların belirli bir kısmını görmek için yararlıdır; `tail` sondan, `head` baştan itibaren belirtilen değer kadar dosyaların içini döker:

```
% tail  -24   dosyaAdı ⏎     sondan 24 satırı gösterir
% tail  +33   dosyaAdı ⏎     33. satırdan dosya sonuna kadar ...
% head  -99  dosyaAdı ⏎      ilk 99 satırı gösterir
```

Dosyaları İlişkilendirme `ln`

İlişkilendirme, bir dosyayı başka isimler de kullanılmasına imkan veren bir özelliktir; böylece tek bir kopyası olan dosya içeriğine birden çok dosya ismiyle erişilebilir. Biri üzerinden yapılan değişiklik, ilişkili olduğu dosyalarda da yapılmış olur. İlişkilendirme yapılarak gerçekte çalışma alanında olmayan dosyaları üzerinde, çalışma alanındaymış gibi kullanma imkanı kazandırır. Örneğin aşağıdaki şekli ile, */etc/passwd* dosyası *sifre* ile ilişkilendirilmiştir. Artık */etc/passwd* üzerinden işlem yapılırken *sifre* adı kullanılabilir:

```
% ln  /etc/passwd  sifre ⏎
```

Dosya Yerini Bulma `find`

Aranan dosya veya dosyaların hangi alanlarda olduğunu aramak için kullanılır. Arama işleminin hangi alandan başlayacağı belirtilmelidir; verilen alan kök gibi varsayılır ve onun altındaki tüm altalanlar aranır; / karakteri kökü, . karakteri ise çalışma alanını belirtir. Sonucun ekrana yazılması için *–print* parametresi kullanılmalıdır:

```
% find   /  -name *.txt  -print ⏎
% find   /usr  -name *.c  -print ⏎
% find      -name  proje  -print ⏎
```


▪ Alan Komutları

Alan komutları, alan yaratma, silme ve çalışma alanını değiştirmek gibi işlerin yapılmasını sağlar.

Çizelge-11.3. Alan komutları

Çalışma alanını değiştirme	`cd`	bkz. s.184
Çalışma alanını öğrenme	`pwd`	bkz. s.184
Alan yaratma	`mkdir`	bkz. s.184
Alan silme	`rmdir`	bkz. s.185
Erişim izni değiştirme	`chmod`	bkz. s.185

Çalışma Alanını Değiştirme `cd`

Çalışma alanı aksi belirtilmediği sürece geçerli olan alandır; bu alanı değiştirmek için `cd` komutu kullanılır: Eğer yeni geçilecek alan çalışma alanı içerisinde ise doğrudan adı verilir; bir üst alana geçilecekse .. yazılır; herhangi bir alana geçilecekse adres verilir:

```
% pwd ⏎
/usr/batuhan
```

```
% cd  deneyler ⏎
% pwd ⏎
/usr/batuhan/deneyler
```

```
% cd .. ⏎
% pwd ⏎
/usr
```

Çalışma Alanını Öğrenme `pwd`

O anda hangi alanda çalışıldığı `pwd` komut ile öğrenilir:

```
% pwd ⏎
/usr/oguzhan
```

Alan Yaratma `mkdir`

Dosya sisteminde yeni bir alan yaratmak için `mkdir` komutu kullanılır: Yeni alan bir adres belirtilmemişse çalışma alanı içerisinde oluşturulur.

Adres belirtilerek çalışma alanı içerisinde de oluşturulabilir:

```
% mkdir  yazilim ⏎
% mkdir /usr/batuhan/deneyler ⏎
```

Alan Silme `rmdir`

Alanı silip dosya sisteminden kaldırmak için kullanılır; silinecek olan alanın içi boş olmalıdır ve silme yetkisine sahip olunmalıdır:

```
% rmdir yazilim ⏎
```

Alan Erişim İznini Değiştirme `chmod`

Alan erişim izinleri, dosya erişim izinlerinin değiştirilmesi ile aynıdır. Erişim izni değiştirilmesi parametrelerin verilmesi açısından biri sayısal, diğeri simgesel olarak iki ayrı şekilde yapılabilir.

```
% chmod  644   alanAdı  ⏎
% chmod  {a u g o} {+ - } {r w x }  alanAdı  ⏎
```

Örnekler ve açıklamalar için bkz. Dosya Erişim İzninin Değiştirilmesi.

▪ Sorgulama Komutları ve ...

UNIX sorgulama komutları, herhangi bir anda sistem durumu ve kaynakları hakkında bilgi edinmek için kullanılır; bu kısımda sorgulama komutlarına ek olarak proses sonlandırma, şifre değiştirme, zamana bağlı komut yürütme, yazıcıdan çıktı almak gibi komutlar verilmiştir.

Çizelge-11.4. Sorgulama Komutları

Aktif kullanıcıları öğrenme	who	bkz. s.186
Boş disk alanını sorgulama	df	bkz. s.186
Disk kullanımını sorgulama	du	bkz. s.186
Tarih sorgulaması	date	bkz. s.187
Yazıcıdan çıkış almak	lpr	bkz. s.187
Aktif prosesleri öğrenme	ps	bkz. s.187

Proses sonlandırma	kill	bkz. s.188
Şifre değiştirme	passwd	bkz. s.188
Zincirleme ara çıkış alma	tee	bkz. s.188
Yürütülme süresini öğrenme	time	bkz. s.190
Zamanlamalı komut yürütme	at	bkz. s.190
Sistem adını sorgulama	uname	bkz. s.190

Aktif Kullanıcıları Öğrenme `who`

Herhangi bir anda sisteme kimlerin girmiş olduğunu, kimlerin çalıştığını öğrenmek için `who` komutu kullanılır; yalın bir şekilde kullanırsa;

```
% who ⏎
root          ttyb   Sept   1       09:15
batuhan       tty01  Sept   1       03:22
oguzhan       tty03  Sept   1       11:02
```

Boş Disk Alanını Öğrenme `df`

`df` komutu, dosya sistemi bünyesinde bulunan disk gibi saklama birimlerinin, o anda boş alanını blok bazında gösterir. Yalın olarak kullanılırsa, dosya sistemlerinin hepsi için boş alan bilgilerini verir. Yanında, parametre olarak belirli bir saklama birim verilebilir.

```
% df ⏎
```

Disk Kullanımını Sorgulama `du`

Bir kullanıcının disk alanı kullanım büyüklüğünü görmek için `du` komutu kullanılır; yalın olarak kullanırsa, o andaki çalışma alanı ve onun altındaki altalanların disk kullanımı blok bazında listelenir. Parametre olarak belirli bir alan adresi verilirse, o alan ve onun altındaki altalanların disk kullanım miktarları da listelenir.

```
% du  ⏎
12      /usr/batuhan
4       /usr/batuhan/proje
8       /usr/batuhan/arge
```

Tarih Sorgulaması `date`

Tarih ve saat bilgisini öğrenmek için date komutu kullanılır. Yalın olarak kullanılırsa tarih ve zaman bilgisini aşağıdaki gibi verir:

```
% date ⏎
Sun Jun       13     15:20  TUR    1999
```

Yanına parametre verilerek tarih ve zaman bilgisi daha farklı şekillerde alınabilir. Örneğin %j parametresi verilirse yılın kaçıncı günü, %w verilirse ayın kaçıncı günü olduğu öğrenilir.

```
% date %j ⏎
164
```

Yazıcıdan Çıkış Almak `lpr`

Yazıcıdan çıkış almak için `lpr` komutu kullanılır; ağda birçok yazıcı varsa, hangi yazıcı olduğu da belirtilmelidir. Örneğin yazıcı mavi ise,

```
% lpr  -Pmavi  dosyaAdı ⏎
```

şeklinde yazılır. `print` komutu da benzer işleve sahiptir.

Aktif Prosesleri Öğrenme `ps`

UNIX'li sistemde o anda aktif olan prosesleri öğrenmek için ps komutu kullanılır. Yalın olarak kullanılırsa,

```
% ps ⏎
PID    TTY    TIME   COMMAND
37     tty01  12:00  sh
36     tty04  13:00  deneme
1996   tty09  16:00  ps
```

En soldaki sütun proses numaralarını gösterir; *deneme* adlı prosesin numarası 1996'dır.

Daha ayrıntılı bir listeleme yapılması istenirse *-l* parametresi ile kullanılabilir.

Proses Sonlandırma `kill`

Bir UNIX prosesi, herhangi bir anda sonlandırılabilir; bu durumda o prosese atanan sistem kaynakları geri alınır ve proses yok edilir. Özellikle uçan programların sonlandırılması için kullanılabilir. Bir prosesin sonlandırılması için onun numarası bilinmelidir; `ps` komutu kullanılarak öğrenilebilir: 36 nolu prosesi sonlandırmak için,

```
% kill 36 ⏎
```

şeklinde yürütülür.

Şifre Değiştirme `passwd`

Kullanıcı şifresi `passwd` komutu ile değiştirilir. Yalın olarak kullanılır. İlk önce eski şifre sorulur ve ardından yeni şifre iki kez sorulur :

```
% passwd ⏎
Current Password : ********⏎
New Password : ******* ⏎
New Password (again) : ******* ⏎
```

Zincirlemede Ara Çıkış Alma `tee`

Çoğu UNIX komutu, aksi belirtilmediği sürece girişi klavyeden alır, çıkış ekrana yansıtır, Klavye ve ekran, çoğu komut için Giriş/Çıkış birimidir. Aslında, komutun yürütülmesi sırasında gerekli olan veri, standart giriş olarak adlandırılan bellek alanından; komutun sonucunu gösteren veri standart çıkış olarak adlandırılan bellek alanına yazılır. Bu nedenle, standart G/Ç nereye yönlendirilmişse komutun yürütülmesi sırasında gerekli G/Ç işlemleri oraya yapılır.

Standart G/Ç, klavye ve ekran yerine dosyalara yönlendirilebilir. Yani, komutun yürütülmesi sırasında kullanılacak veri, klavyeden değil de bir dosyadan okunabilir; veya tersi, sonuç ekrana değil de bir dosyaya yazılabilir. Bu amaçla yönlendirme karakterleri kullanılır: '<' karakteri girişin nereden, '>' karakteri çıkışın nereye yapılacağını gösterir.

```
% ls  >  liste ⏎
```

biçiminde yazılırsa, çalışma alanı listelemesi liste adlı dosyaya yapılır; liste adlı dosya yeni yaratılır ve sonuç oraya yazılır. Eğer daha önce liste adlı dosya varsa, çıkışın bu dosyaya eklenmesi istenirse '>' yerine '>>' kullanılır.

```
% ls  >>  liste ⏎
```

Zincirleme karakteri olan '|' bir çeşit G/Ç yönlendirmesi yapar. Yararı, bir komutun ürettiği sonuçları ardından yürütülen bir başka komuta giriş olarak aktarmasıdır. Çünkü bazı durumlar vardır ki, komutları ard arda kullanmak büyük esneklik sağlar. Örneğin `who` komutu o anda çalışan kullanıcıları listeler:

```
% who ⏎
```

Kullanıcı sayısı çok fazla ise, aradığımız bir kullanıcının o anda çalışıp çalışmadığını görmek zor olabilir. Ancak kullanıcı listesi ekrana değil de bir dosyaya yazılırsa, aranan kullanıcı `grep` komutu kullanılarak aranabilir:

```
% who  > sonuc ⏎
% grep batuhan  sonuc ⏎
```

Burada yapılmak istenen kalabalık bir kullanıcı listesinden *batuhan* adlı kullanıcıyı aramaktır. Aynı işlem zincirleme karakteri olan '|' ile daha kolayca gerçekleştirilir:

```
% who | grep batuhan  ⏎
```

Zincirleme komutunun diğer bir yararlı kullanım şekli, `ls` komutunun çok uzun olan listesini sayfa sayfa göstermek için `more` komutuyla birlikte kullanılmasıdır :

```
% ls | more ⏎
```

Birden çok zincirleme karakteri aynı komut satırında kullanılabilir; bu tamamen kullanıcı isteği ve mantığına bağlıdır. Aşağıdaki örnekte ilk komut çalışma alanındakileri listeler; ikinci komut bu liste içerisinde *proje* adlı dosyayı arar ve üçüncüsü de bulunanları sıralar ve *sonuc* adlı dosyaya yazar:

```
% ls | grep proje | sort > sonuc ⏎
```


Örneklerde görüldüğü gibi komutlar ard arda kullanılırken, komut aralarında çıkış almak için `tee` komutu kullanılır. Böylece bir komutun ürettiği *sonuc* bir sonraki komuta giriş olarak aktarılırken, ara çıkış ta ayrıca saklanır:

```
% ls | grep proje | tee arasonuc | sort > sonuc ⏎
```

Yürütülme Süresini Öğrenme `time`

Bir komutun yürütülürken ne kadar zaman aldığı, komutun time takısıyla birlikte çalıştırılmasıyla öğrenilebilir. Örneğin `ls` komutu time ile birlikte kullanılırsa, saniye olarak istatistiksel bilgileri verir:

```
% time ls  ⏎
```

Zamanlamalı Komut Yürütme `at`

Önceden belirlenen zamanlarda komut yürütülmesi `at` komutu ile kotarılır. genel yazım biçimi :

```
% at  zamanTarih  programAdı ⏎
```

Belirtilen program, yine belirtilen tarih ve zamanda yürütülür. Örneğin,

```
% at  1200 jun 13  merhaba ⏎
```

şeklinde yazılırsa, merhaba adlı program *13 Mayıs* saat *12:00*'de yürütülür.

Sistem Adını Sorgulama `uname`

Bu komut, o anda çalışılan sistem hakkında bilgi edinmek için kullanılır. *–a* parametresi ile kullanılırsa, daha ayrıntılı bilgi verir.

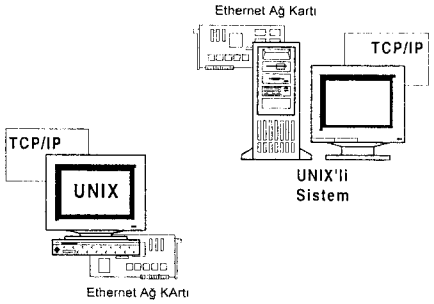
▪ TCP/IP Ağ Komutları

UNIX sistemlerinin bulunduğu ağlarda, ağı gözlemek, oluşan ağ problemlerini çözmek, fiziksel veya IP adreslerin fiziksel arayüzlere atanması, yönlendirme tablosu üzerinde değişiklik yapılması için birçok ağ komutu vardır.

Çizelge-11.4. TCP/IP ağ komutları

IP-fiziksel adres dönüşüm tablosunu görme	`arp`	bkz. s.193
Ağdaki kullanıcıları sorgulama	`finger`	bkz. s.194
Ağ arayüz değerlerini yerleştirme/ekleme	`ifconfig`	bkz. s.194
Ağ konfigürasyon değerlerini öğrenme	`ipconfig`	bkz. s.194
Ağ bağlantı/protokol istatistiklerini görme	`netstat`	bkz. s.194
İletişim bağlantısını sorgulama	`ping`	bkz. s.196
Yönlendirme tablosunu güncelleme	`route`	bkz. s.197

Özellikle sistemlerin ilk eklenmesi sırasında veya bir sorunla karşılaşıldığında bu yardımcı ağ komutlarının kullanılması kaçınılmaz olur. Bu Bölümde TCP/IP protokol kümesinin kullanıldığı ağlarda UNIX[1] işletim sistemlerinin sahip olduğu yardımcı ağ komutları incelenmiştir.

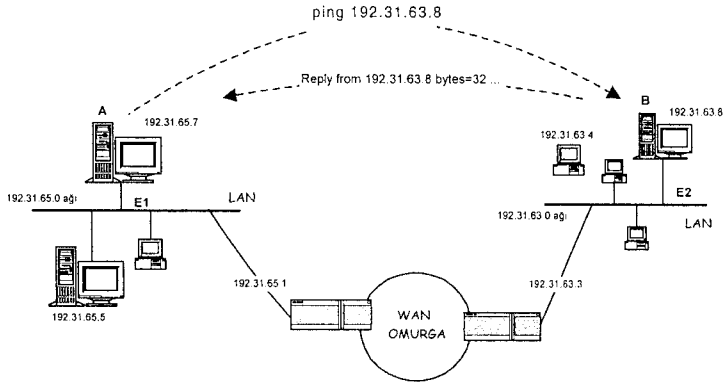


En çok kullanılan yardımcı komut `ping` komutudur; iletişim yapılmak istenen karşı sistemle iletişim bağlantısı olup olmadığını sınamak için kullanılır. Örneğin IP adresi 212.65.129.2 olan bir bilgisayara `ping` çekildiğinde, onunla o an için iletişim bağlantısı varsa, karşı sistem `ping`'e cevap verir. Böylece iki sistem arasında iletişim bağlantısı olduğu anlaşılır:

```
% ping  192.31.63.8 ⏎

Pinging 192.31.63.8  with 32 bytes of data:
Reply from 192.31.63.8:  bytes=32  time=17 ms   TTL=25
Reply from 192.31.63.8:  bytes=32  time=12 ms   TTL=25
Reply from 192.31.63.8:  bytes=32  time<10 ms   TTL=25
Reply from 192.31.63.8:  bytes=32  time=10 ms   TTL=25

% _
```



Eğer `ping` çekilen karşı sistemle o an için iletişim bağlantısı yoksa aşağıdaki gibi bir mesaj alınır:

```
% ping  192.31.63.4 ⏎

Pinging 192.31.63.4  with 32 bytes of data:
Request time out.
Request time out.
Request time out.
Request time out.

% _
```

[1] UNIX dışında TCP/IP protokol kümesi yüklenebilen diğer işletim sistemlerinde de benzer ağ komutları vardır. Çoğu zaman parametre kullanım şekilleri de birbirine çok benzer; ayrıntılı parametre listesi, ilgili sistemde yardım (`help`) komutu kullanılarak görülebilir.

IP-Fiziksel Adres Dönüşüm Tablosunu Görme `arp`

`arp` komutu, dönüşüm tablosunu (translation table) görmek veya içini değiştirmek için kullanılır. Dönüşüm tablosunda IP adreslerine karşılık gelen donanım adresleri (Ethernet veya Token Ring gibi ağa erişim arayüzlerinin fiziksel adresleri) tutulur; bunlar adres belirleme protokolu (Address Resolution Protocol, ARP) tarafından dönüşüm esnasında kullanılır.

Bilindiği gibi ağ katmanı düzeyinde adreslemede IP adresleri, fiziksel katman seviyesine fiziksel adres kullanılır. Dolayısıyla üst katmanlarda IP adresiyle gönderilen verinin fiziksel ortamdan aktarılabilmesi için, IP adreslerinin fiziksel karşılıkları bulunmalıdır. Bu işlem, fiziksel arayüz olarak Ethernet veya Jetonlu Yol (Token Bus) kullanılan LAN'lar için gereklidir. Genel yazımı ve yanında kullanılan parametreler aşağıda verildiği gibidir:

```
% arp  -parametre  Ipadr  (ya da Fzkadr) ⏎
```
bu yazım şekli sorgulama yapmak için kullanılır

```
% arp -parametre  Ipadr  Fzkadr  ⏎
```
yeni IP→fiziksel adres dönüşüm çifti eklemek için kullanılır

Parametreler :

-a	Adresi *Ipadr* ile belirtilen sistemin ARP dönüşüm değerlerini gösterir; *Ipadr* verilmezse hepsinin sahip olduğu ARP değerlerini gösterir.
-d	Adresi *Ipadr* ile verilen sistemin ARP dönüşüm değerlerini siler.
-s	Adres dönüşüm tablosuna yeni adres çiftleri eklemek için kullanılır. IP adresi bilinen noktalı formatta, fiziksel adres ise herbiri tire (-) ile ayrılmış onaltılık tabanda yazılmış 6 tane sayı çifti şeklinde verilmelidir.

Örnek :

```
% arp  -s  194.240.120.55  02-80-3c-22-04 ⏎
```


Ağdaki Kullanıcıları Sorgulama `finger`

`finger` komutu bir sistemde tanımlı kullanıcının o anda çalışıp çalışmadığını ve birtakım bilgilerini öğrenmek için kullanılır; sonuç vermesi için karşı sistemin bu hizmeti sunuyor olması gerekir. Genel yazım biçimi,

```
% finger  kullanıcı_kimliği@sistem_adresi ⏎
```

Ağ Arayüz Değerlerini Yerleştirme `ifconfig`

Bu komut, ağ arayüzü (network interface) üzerinde yerleştirim yapmak için kullanılır: arayüz parametrelerine atama yapılabilir (adres atanabilir); arayüz etkin veya pasif hale getirilebilir.

```
% ifconfig  en0  120.55.10.1  netmask  255.255.0.0  ⏎
```
en0 adlı arayüze 120.55.10.1 adresi, 255.255.0.0 maskesi atanır

```
% ifconfig  en0  120.55.10.1  broadcast 120.55.0.0  ⏎
```
en0 adlı arayüze 120.55.10.1 adresi, 120.55.0.0 yayın adresi olarak atanır

```
% ifconfig  en0 up ⏎
```
en0 adlı arayüzü etkin kılar; down ile de pasif olur ve o sisteme erişilemez

Ağ Konfigürasyon Değerlerini Öğrenme `ipconfig`

Bu bir sorgulama komutudur; o andaki bütün TCP/IP ağ konfigürasyon değerlerini gösterir. Yanına parametre verilmezse, IP adresi, altağ *(subnet)* maskesi gibi değerler listelenir.

Ağ Bağlantılarını ve Protokol İstatistiklerini Görme `netstat`

`netstat` bir sorgulama komutudur; ağ ile ilgili parametrelerin durumunu görmek için kullanılır; o an kurulmuş olan TCP/IP oturumları, TCP, UDP ve IP protokollerinin kullanım istatistikleri görülebilir. Genel yazım biçimi ve kullanılabilecek parametreler aşağıdaki gibidir:

```
% netstat  -parametre(ler) ⏎
```

Parametreler :

-a Oturum kurulmuş bütün bağlantılar ve o anda dinleme (**listening**) durumunda olan prosesler listelenir. Hangi 'daemon'-ların etkin olduğu görülmek isteniyorsa, yine bu parametre kullanılır.

-A Hatalı kurulmuş olan bağlantının durumunu gösterir. a parametresiyle gösterilenlere ek olarak protokol hakkında bilgi içeren PCB (Protocol Control Block) bilgisini de verir.

-h Kurulmuş olan bütün bağlantıları gösterir.

-i Ağ arayüzünün (Network Interface) konfigürasyonunu gösterir.

-n Ağ adreslerini listelerken, simgesel adres değil de sayısal adres gösterilir.

-r IP yönlendirme tablosunu gösterir.

-s Aktarımda oluşan hataların istatistiki bilgisini gösterir.

-p *protokol* *protokol* ile belirtilen protokolun bağlantı bilgisini gösterir.

Örnek :

```
% netstat  -a  ⏎

Proto  Recv-Q Send-Q  Local Address      Foreign Address (state)
tcp      0      0      sariyer.telnet     topkapi.4526  ESTABLISH
tcp      0      0      *.telnet           *.*           LISTEN
tcp      0      0      *.ftp              *.*           LISTEN
udp      0      0      *.snmp             *.*           -
```

- Proto : Bağlantısı kurulan protokolun adını belirtir.
- Local Address : O anki bilgisayarın simgesel IP adresi ve bağlantının kullandığı port numarası; port oluşturulmamışsa * karakteri görülür.
- Foreign Address: Bağlantı kurulan bilgisayarın simgesel IP adresini gösterir.
- (state): Yalnızca TCP bağlantılarının durumunu gösterir. Bu sütunda verilen durum bilgilerinden bazıları şöyledir:
  - LISTEN (dinlemede, istek bekliyor)
  - CLOSED (kapatılmış)
  - ESTABLISHED (oturum kurulmuş)


```
% netstat  -i  ⏎

ame    MTU    Network   Address Ipkts  Ieers    Opkts errs
en0    1500   pasha     sariyer 4560     0      3123   0
lo0     4096  Loopback          me      0       0     0
```

- Name : Fiziksel arayüz adı
- MTU : Arayüzün en büyük aktarım birimi (Max. Transmission Unit)
- Network, Address : İlki ağın, ikincisi sistemin adını gösterir
- Ipkts, Opkts : Gelen ve giden paketlerin sayısını gösterir
- Ierrs, Oerrs: İlki gelen, ikincisi giden hatalı paket sayısını gösterir.

İletişim Bağlantısını Sorgulama ping

Bu sorgulama komutu ağdaki bir bilgisayar veya ağ aygıtına erişilip erişilemeyeceğini öğrenmek için kullanılır. Genel yazım biçimi,

```
% ping  -parametre(ler)  sistem_IPadresi  ⏎
```

ping komutu karşı sisteme erişilip erişilmeyeceğini belirlemek için oraya ICMP paketleri gönderir ve kendisine yanıt vermesini bekler. Aksi belirtilmediği zaman çoğu UNIX'te gönderilen paketin boyu 64 sekizlidir; ard arda 4 tane paket yollar ve gelen yanıta göre aşağıdaki gibi mesaj verir. Örnek kullanımı,

```
% ping   164.120.55.1  ⏎

Pinging  toroslu  [164.120.55.1] with 64 bytes of data:

Reply from 164.120.55.1:  bytes=64  time=10 ms   TTL=255
Reply from 212.65.129.2:  bytes=64  time=12 ms   TTL=255
Reply from 212.65.129.2:  bytes=64  time <10 ms  TTL=255
Reply from 212.65.129.2:  bytes=64  time <17 ms  TTL=255
```

Parametreler :

-t Kullanıcı tarafından durdurulana kadar karşı sisteme paketi yollar ve sonucu ekrana yazar.

-a Ana makinaların adreslerini çözümler.

-n *sayı* Karşı sistemi sınarken *sayı* tane ECHO paketi gönderir; bu parametre kullanılmazsa bu sayı 4'tür.

-l *uzunluk*	ECHO paketlerinin boyunu değiştirmek için kullanılır; *uzunluk* paket boyunu gösterir. Varsayılanı 32 veya 64'tür.
-f	Paketin ağ aygıtlarında parçalanamayacağı belirtilir.
-i *ttl*	*ttl* ile belirtilen değeri paketin yaşam süresi içine yerleştirir.
-v tos	HT ile belirtilen değeri paketin hizmet türü alanına yerleştirir.
-j sistemListesi	Sınama paketlerini belirtilen düğümler üzerinden yönlendirir.
-W zamanaşımı	ZA ile milisaniye cinsinden zaman aşımı verilebilir.

Parametrelerin birkaç tanesi aynı anda kullanılabilir. Örneğin paket boyunu 512 sekizli olacak şekilde sürekli `ping` çekmek için aşağıdaki gibi yazılır:

```
 % ping   164.120.55.1  -t  - l  512  ⏎

Pinging  toroslu  [164.120.55.1] with 64 bytes of data:

Reply from 164.120.55.1:bytes=512  time=27 ms   TTL=255
Reply from 212.65.129.2:bytes=512  time=22 ms   TTL=255
Reply from 212.65.129.2:bytes=512  time=21 ms   TTL=255
Reply from 212.65.129.2:bytes=512  time=21 ms   TTL=255
    ...
```

Yönlendirme Tablosu Güncelleme `route`

`route` komutuyla sistemlerin yönlendirme tablosuna statik olarak yönlendirme bilgileri girilir; yeni statik yönlendirme bilgisi girilmesi, var olanların silinmesi gibi işlevi vardır. Bu komutun ayrıntılı örnekleri için bkz. Bölüm 12.

```
% route add  gideceği_adres  yönleneceği_adres  metrik ⏎
% route  add  default  yönleneceği_adres  metrik ⏎
```

Komut Dosyası Hazırlama

UNIX'te var olan komutlar, sistem ve parametre değişkenleri kullanılarak yeni komut dosyaları hazırlanabilir. Bu tür dosyalar özellikle, bir işin yapılması için ard arda komut yürütülmesi gerekli olan durumlarda kolaylık sağlar.

En basitinden bir komut dosyası aşağıdaki gibi oluşturulabilir:

```
% cat  > yenikomut  ⏎
echo "Kullanicilar :"
who | sort | more
echo "olarak tesbit edilmistir"

Ctrl+D
```

Yukarıda `yenikomut` adlı dosya bir komut dosyasıdır. Eğer dosya için çalıştırılma izni verilmişse (`chmod` ile yapılır), doğrudan `yenikomut` yazılarak çalıştırılabilir:

```
% yenikomut  ⏎
Kullanicilar :
muge, oguzhan, begum, batuhan ...
olarak tesbit edilmistir
```

Komut dosyası hazırlanırken, esneklik kazandıran değişkenler kullanılabilir. Böylece yeni komut dosyası içerisinde karşılaştırma işlemleri yapılabilir, komut satırından parametre aktarılabilir... (Sistem değişkenleri ve parametre değişkenleri için bkz. Bölüm 7 - Sistem Değişkenleri, Parametre Değişkenleri)

Aşağıdaki örnekte parametre değişkenleri kullanılarak bir komut dosyası hazırlanması verilmiştir:

```
% cat  > ara  ⏎
echo "1. Aranan : " $1
netstat  -a | grep  $1
echo "2. Aranan : " $2
netstat -rn | grep $2

Ctrl+D
```

Bu komut dosyası yürütülürken iki tane parametre verilmelidir; ilk parametre *$1*'e, ikinci parametre *$2*'ye aktarılır. `netstat` sistemin yönlendir-

me tablosu içeriği hakkında bilgi veren komuttur. Yeni komut dosyası ara, aşağıda görülen parametreler ile yürütülürse,

```
% ara  ESTABLISH  199.6.13.1  ⏎
```

ESTABLISH değeri *$1*'e, *199.6.13.1* değeri de *$2*'ye atanır. Birinci `netstat` komutu sonucu içerisinde *ESTABLISH* sözcüğü aranır; ikinci `netstat` komutu sonucu içerisinde ise *199.6.13.1* IP adresi olup olmadığı aranır.

UNIX Editörleri

Bütün işletim sistemleri, temel yazma işlemlerinin kotarılması amacıyla kendi bünyelerinde standart metin editörleri barındırırlar. Bu editörler kullanılarak sistemle ilgili dosyaların düzenlenmesi yapılır. Bu amaçla UNIX sistemlerde `vi`, `pico`, `ed` ve `ex` editörleri vardır. `vi` editörü UNIX sistem yöneticileri tarafından zaman zaman kullanılan bir editördür.

vi Editörü

`vi`, yalın bir metin editörüdür; yazılanlar dosyaya ekranda görüldüğü gibi kaydedilir. Editörü çalıştırmak ve bir dosyayı açmak için,

```
% vi dosyaAdı ⏎
```

yazılması yeterlidir. Eğer dosya çalışma alanında varsa, o dosya içindekiler görüntülenir; böyle bir dosya yoksa belirtilen isimde yeni bir dosya oluşturulur. Örneğin *network* adlı bir dosya oluşturmak için aşağıdaki gibi yazılması yeterlidir:

```
% vi network ⏎
```

Editör çalıştıktan sonra kullanıcının karşısına editör ekranı gelir. Editör ortamında çeşitli çalışma modları vardır. Örneğin yazma modu, silme modu, komut satırı modu gibi; dolayısıyla hangi işlem yapılmak isteniyorsa, editör ilk önce o moda sokulmalıdır.

a	ekleme modu
i	araya yazma modu
A veya $a	satır sonuna ekleme modu
I veya ^i	satır başından araya ekleme modu

Yeniden komut satırı moduna dönmek için *ESC* tuşuna basılır.

Metin Yazmak/Düzenlemek

Editör ortamında yazma veya düzeltme işlemleri için ilk önce ilgili moda geçilmelidir. İmleç hareketleri doğrudan klavye üzerindeki yönlendirme tuşları aracılığıyla gerçekleştirilir; bu tuşlar etkisiz ise, imleç hareketi için aşağıdaki karakterler kullanılabilir.

Yapılan değişikliğin iptal edilmesi için *u* komutu kullanılır. '/' karakteri ile dosya içerisinde arama yapılabilir. Örneğin */hosts ESC* yazıldığında *hosts* adlı sözcük aranır.

Çizelge-11.5. İmleç haraket tuşları

l	bir karakter sağa
h	bir karakter sola
j	bir satır aşağıya
k	bir satır yukarı
w	bir sonraki sözcüğün baş harfine
b	bir önceki sözcüğün son karakterine
e	sözcüğün son karakterine
)	bir sonraki cümlenin başına
)	bir önceki cümlenin başına
{	o andaki paragrafın başına
}	bir sonraki paragrafın başına
H	ekranda görülen sayfanın başına
	ekranda görülen sayfanın sonuna
M	ekranda görülen sayfanın ortasına
*n*H	n satır üste
*n*L	n satır alta

Komut Satırı Modu

Editörden çıkmak, dosyayı saklamak gibi işlemler komut satırı modunda gerçekleştirilir.

:w	son yazılanlar saklanır
:q	editörden çıkılır
:q!	saklama yapılmadan editörden çıkılır
:!	editör ortamında UNIX komutu yürütülmesini sağlar
:set all	seçenekler listelenir
:set num	yazıya satır numaraları ekler; none kaldırır
:set wm	sağ marjin ayarlaması yapar

`tar` Yardımcı Saklama Programı

`tar`, bir çeşit birden çok dosyayı dosya adları saklı kalmak kaydıyla birleştirip tek bir dosya haline getiren bir yardımcı programdır. Örneğin bir alan içindeki dosyalar `tar` ile birleştirildikten sonra herhangi bir sıkıştırma programı aracılığıyla sıkıştırılabilir. Bu durumda uzantısı, *.tar.Z*[1] olan bir dosya elde edilir. Bu dosya arşivlenerek saklanabilir veya FTP ile aktarılabilir. Aşağıda `tar` kullanımı için birkaç örnek görülmektedir:

```
% tar cvf   sonuçDosyaAdı.tar  dosyaVeyaAlanAdı ⏎
```

İkinci parametre olarak belirtilen dosya ve alanları birleştirip ilk parametrede verilen isim altında saklar.

```
% tar  xvf  tar'lıDosyaAdı.tar  dosyaAdı ⏎
```

Birleştirilmiş dosyadan belirtilen dosyayı yeniden elde eder.

```
% tar  xtf  tar'lıDosyaAdı.tar ⏎
```

Birleştirilmiş olan dosya adlarını listeler.

cvf gibi parametreler, birleştirmenin nasıl yapılacağını belirtir: *c*, birleştirilmiş dosya oluşturur; *x*, eski haline getirmek için açar; *t*, birleştirilmiş dosya içindekileri listeler.

[1] tar.Z'deki .Z uzantısı UNIX'teki compress sıkıştırma programının kullandığı uzantıdır;gzip programı .gz veya .z, yabba programı .Y ve freeze programı .F uzantılarını kullanırlar.