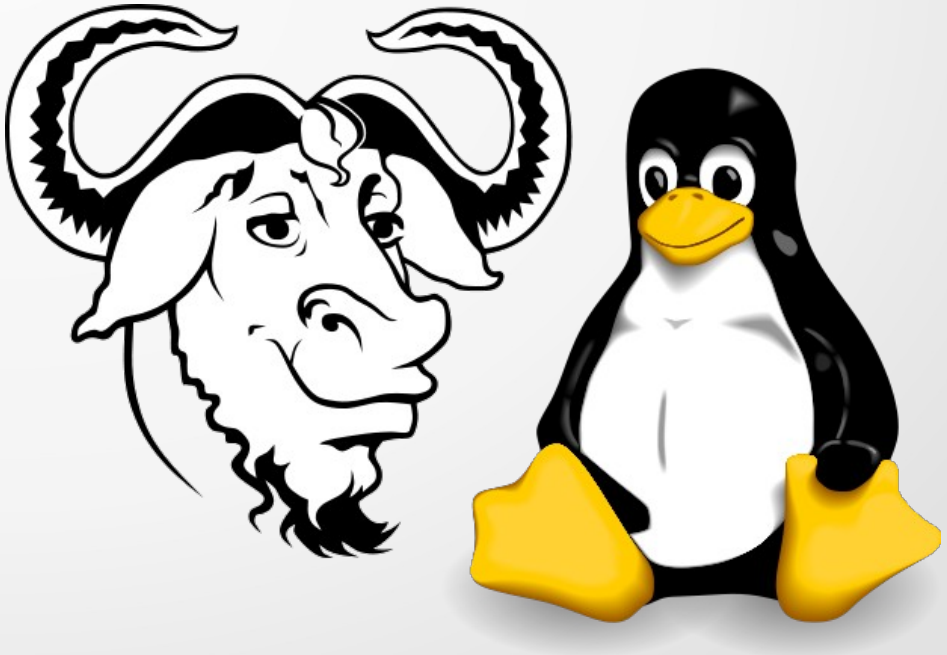
Komut Satırı Temel Komutlar

Mustafa Akgül Kış Kampı 2019

Eğitmenler: Duygu Ölmez
Emre Kondul

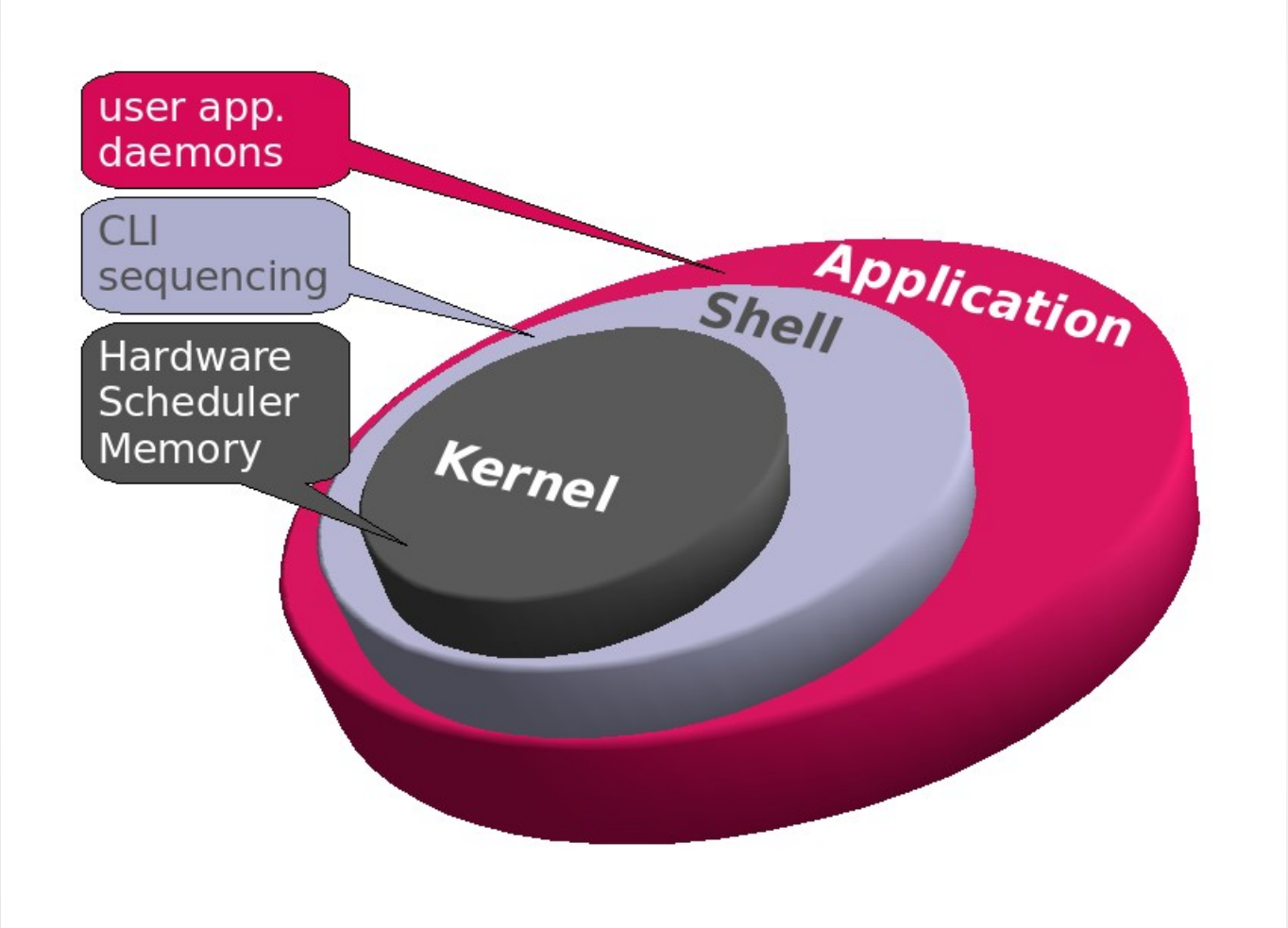


Konular

Uçbirimde yardım almak
Dosya türleri
Temel Komutlar

- **Dosyaların basit seviyede düzenlenmesi**
- **Dosya dizin arama işlemleri**
- **Dosya içerik manipülasyonları**
- **Donanım bilgisi toplama**

KABUK KAVRAMI



KABUK KAVRAMI

Kullanıcıların işletim sistemi ile etkileşebileceği iki ana ortam mevcuttur. Bunlardan birisi grafik masaüstü, diğeri de komut satırıdır. Komut satırını sunan ve komutları yorumlayıp çalıştıran kabuk programıdır.

En yaygın kullanılan kabuk programı **Bash** olmakla birlikte ksh, csh, zsh gibi farklı kabuklar da bulunmaktadır.

Sistemde kullanılabilecek tüm kabuklar /etc/shells dosyasında tanımlanmaktadır.

Sisteme giriş yaparken her kullanıcı için tanımlı bir kabuk çalışır. Kabuk programı kullanıcıya bir komut satırı (prompt da denir) sunar ve komut girmesi için bekler. Bash bu komutu yorumlayıp gerekli programları çalıştırır ve sonlanmalarını bekler. İşlemler bittikten sonra kontrol yeniden kabuğa geçer ve kabuk yeniden komut girişi için bekler. Kabuktan çıkış yapılana kadar bu döngü devam eder.

$ (dolar) işareti normal bir kullanıcı için varsayılan işarettir. Bir diğer adıyla sınırlı kullanıcı oturumunda bulunduğunu gösterir.
 # (diyez), root kullanıcısı için varsayılan işarettir.
echo $SHELL kullanılan kabuk için

Bourne-Again SHell (Bash)

Bash kabuğu ilk UNIX sistemlerinde hem sh ve ksh kabukları,
hem de bazı csh özellikleri için orjinal olarak geliştirilen özellikleri içerir.

Kullanılan kabuğu görüntülemek için aşağıdaki komut kullanılır.

```
$ echo $SHELL
/bin/bash
```

Kabuğu değiştirmek için chsh komutu kullanılabilr:

```
$ chsh -s /bin/sh
Changing shell for user1.
Password:
Shell changed.
```

Yeni kabuğun aktif hale geçmesi için sistemden çıkıp yeniden giriş yapmak gerekir.
Eğer kullanılan kabuk yalnızca mevcut oturum için değiştirilmek istenirse doğrudan
kabuk çalıştırılır:

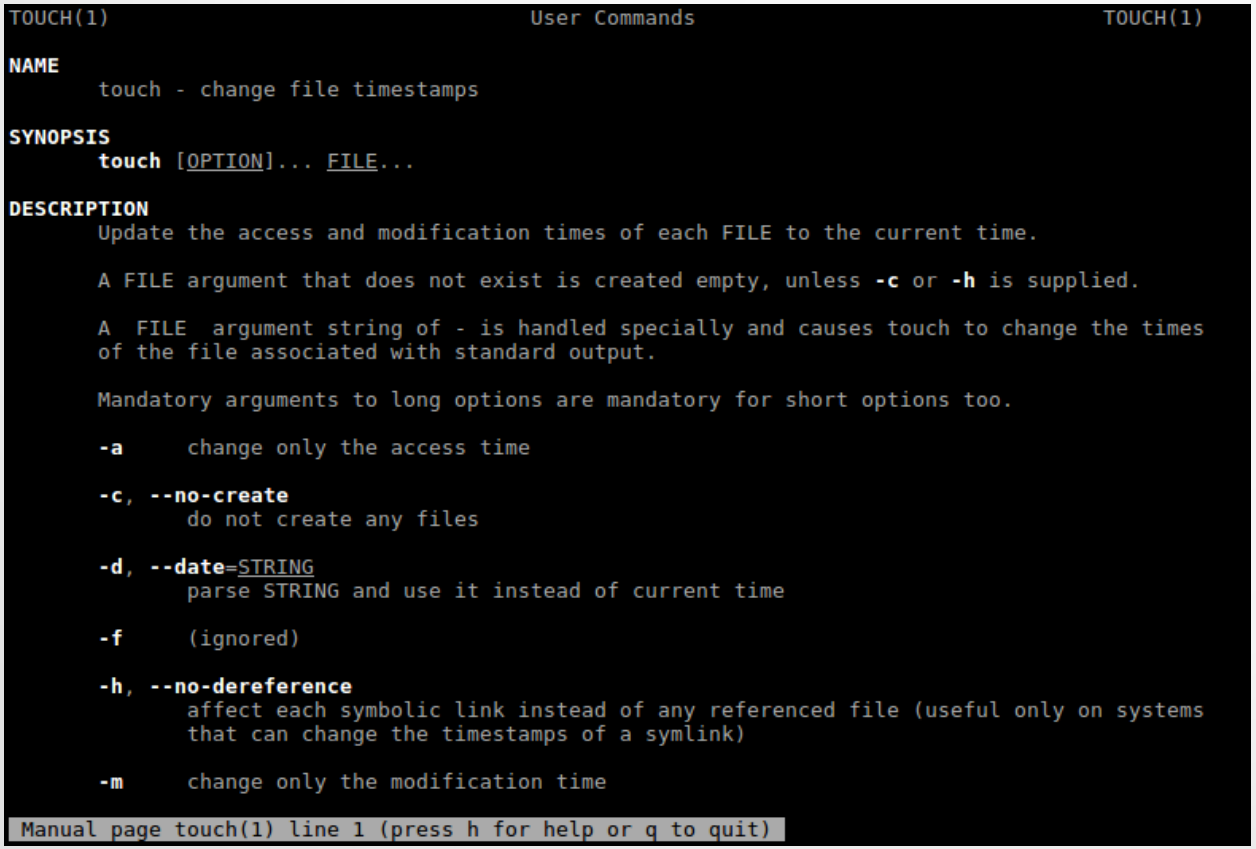
```
$ bash
user1@pardus $
```

Uçbirimde yardım almak

man & -help

Bir komutun nasıl kullanılacağını ve hakkında daha fazla bilgi edinmek için man komutu kullanılır. Örneğin, “man touch” touch komutunun manuel sayfalarını gösterir. Bir komutu yazıp arkasına “–help” eklersek, yine manual çıktısı ile aynı sonuca ulaşırız. (“man -k & apropos”)

$man komut



Man sayfası okunurken ok tuşları kullanılarak aşağı ve yukarı ilerlenebilir.
‘/’ işaretine basıldığında yardım sayfası içinde arama yapılabilir. Bir sonraki arama sonucu için **‘n’**, bir ekran aşağı kaydırmak için **SPACE**, bir ekran yukarı çıkmak için **‘b’** ve yardım sayfasından çıkmak için **‘q’** tuşuna basılır.

Dosya türleri

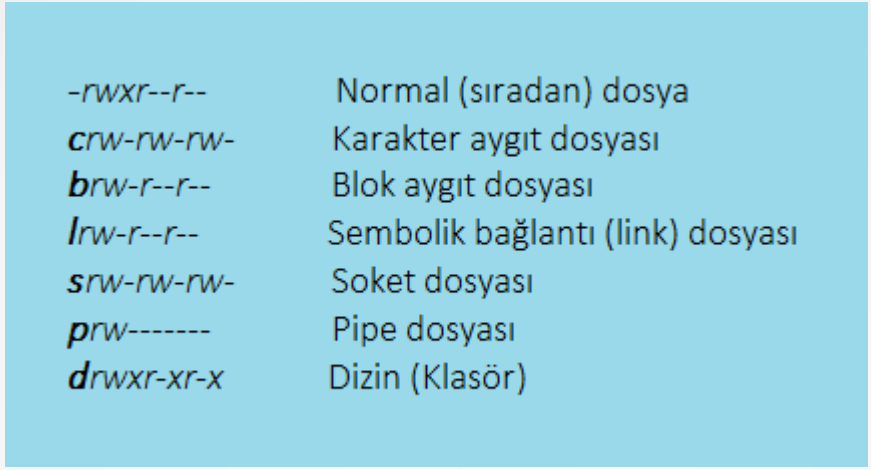
Linux dosya tipleri Aygıt dosyaları ve Sıradan dosyalar olmak üzere 2'ye ayrılır.

1) **Aygıt dosyaları:** donanım ile haberleşmeyi sağlayan dosyalardır.

- Karakter aygıt dosyaları: Byte bazında veri alışverişi yapan aygıtlar (Örn: Klavye, Fare vb.)
- Blok aygıt dosyası: Blok bazında veri alışverişi yapan aygıtlar (Sabit disk vb.)

2) **Sıradan dosyalar:** Aygıt dosyaları dışında kalan dosyalardır.

Dosya türleri



Yukarıda resimde yer alan dosya tiplerini ne tür bir dosya olduğunu ayırt etmenin kolay yolu baş harfine bakmak. Normal dosyaların başında herhangi bir harf bulunmaz.

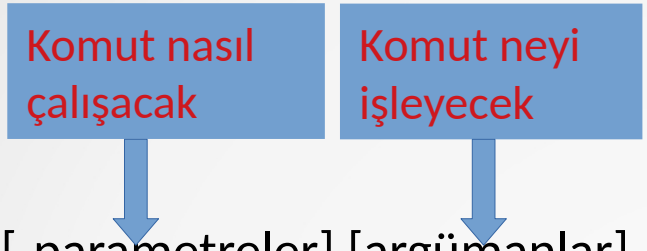
(Burada kastedilen rwxr—r—bölümü değildir. Baş harf r'nin önüne gelen harftir)

Karakter dosyaların başında c harfi bulunur yani İngilizce "character" kelimesinin baş harfidir.

Aynı şekilde Blok aygıt dosyasının da başında b harfi bulunur. Bu da İngilizce "block" kelimesinin baş harfidir. Bu şekilde baş harflerine bakarak hangi dosyanın ne tür bir dosya olduğunu bilebilirsiniz.

Temel Komutlar

Bütün komutlar için temel yapı:



Komut [-parametreler] [argümanlar]

```
ls –l /tmp
```

küçük harfle yazılırlar

Basit Komutlar

- id kendi kullanıcı ve grup bilgilerin
- uname -a çekirdek bilgisini gösterir
- w/who sistemdeki kullanıcıları listeler
- whoami giriş yapılan kullanıcıyı gösterir
- date tarihi görüntüleme
- cal takvimi görüntüleme
- passwd parola değiştirme
- clear ekranı temizleme
- echo ekrana metin basma
- uptime sunucu ne zamandır ayakta ve yükü ne?
- history komut geçmişini gösterir
- hostname bilgisayar adını gösterir

Temel Komutlar

Dizinler			
pwd			Çalışılan dizinin tam yulunu verir
cd		<dizin1>	İstenilen dizine geçiş yapar
cd		..	Bir önceki dizine gider
ls		<dizin1>	Dosya içeriğini listeler
	-a		Gizli dahil dosyaları göster
	-R		Alt dizinleri listele
	-r		Sıralama ters çevir
	-t		Son düzenlemeye göre listele
	-S		Dosya boyutuna göre listele
	-l		Geniş listeleme formatı
	-1		Tel satır, tek dosya
	-m		Çıktı virgül ile ayrılır
	-Q		Tırnak içerisine al

1	pwd
2	cd
3	ls
4	mkdir
5	touch / stat
6	rmdir
7	rm
8	cp
9	mv
10	rename
11	cat
12	tac
12	head
13	tail

14	wc
15	nl
16	cut
17	expand, unexpand
18	fmt
19	pr
20	od
21	paste
22	join
23	split
24	sort
25	uniq
26	< > >> >> |
27	grep
28	ln

29	locate
30	find
31	which
32	wheris
33	vi, nano, more, less
34	?, *, [ ] , \
35	file
36	tee
37	xargs
38	sed
39	awk
40	tr
41	dmidecode, lscpu,lsusb
42	df, du

Temel Komutlar

cd change directory (type cd)

Bu komutu çalıştırdığımızda bulunduğumuz dizin içerisindeki dizin1 dizinüne geçiş yapacaktır.
Bu geçişi terminal ekranından da görebilirsiniz. Artık bulunduğumuz dizini belirten alan değişmiş olacaktır.
Dizin sisteminde . (nokta) bulunduğumuz dizini ifade eder. ..(iki nokta) ise bir üst dizini ifade eder. * ise o dizindeki tüm dosyaları ifade eder.

. ve.. ifadelerini de cd komutu ile kullanabiliriz.
Bulunduğumuz dizine gider: cd .
Bulunduğumuz dizinin bir üst dizinüne gider: cd ./..
Bir üst dizine gider (Üstteki ile aynı işi yapar) : cd ..
Bir üst dizinin içerisindeki dizin1'e gider : cd ../dizin1
Bir üst dizinin içerisindeki dizin1'in içerisindeki dizin2'ye gider : cd ../dizin1/dizin2
cd ~

direct path   / ile başlayan
relative path .. & / ile başlamayan

Temel Komutlar

mkdir

Dizin oluşturmak için kullanılır.

-p Oluşturulacak dizinin üst dizinleri yoksa onları da oluşturur.

$ mkdir arsiv
$ mkdir –p 2013/02/17/18/00

Temel Komutlar

touch / stat (zaman bilgisi)

Dosya oluşturmak için
Eğer belirtilen dosya mevcut ise dosya erişim ve değiştirilme zamanlarını günceller.
Eğer belirtilen dosya mevcut değilse dosyayı oluşturur.

-a Erişim zamanını günceller.
-m Değiştirilme zamanını değiştir.
-t Erişim ve değiştirilme zamanını belirtildiği şekilde değiştir:

MMDDhhmm[.ss]
CC Yılın ilk iki hanesi
YY Yılın son iki hanesi
MM Yılın ayı. 1'den 12'ye.
DD Ayın günleri. 1'den 31'e.
hh Günün saati. 0'dan 23'e.
mm Saatin dakikaları. 0'dan 59'a.
SS Dakikanın saniyeleri. 0'dan 59'a.

18 Ocak 2008 12:05:09 tarihli bir dosya oluşturmak için:
$ touch -t 200801181205.09 deneme

Temel Komutlar

rmdir

Boş dizin silmek için kullanılır. Dizin içinde dosya varsa bu komut dizini silmez. Bunun yerine rm –fr komutu verilmelidir.

$ rmdir mydir

-p parametresi iç içe dizinleri sırasıyla siler.

$ rmdir –p a/b/c

Yukarıdaki komut şu üç işlemi yapar:

rmdir a/b/c, rmdir a/b, rmdir a

rm

Dosya veya dizin siler.

-f Silerken sormaz, varolmayan bir dosya için bilgi vermez.
-i Silmeden önce sorar.
-R -r Dizinlerin ve alt dizinlerin içeriklerini ardarda siler.
-v Yapılan işlem hakkında daha ayrıntılı bilgi verir.

$ rm –f beni.oku
$ rm –fr dizin1
$ rm -v ciro.txt
removed `ciro.txt’

Temel Komutlar

cp

Dosya ve dizin kopyalar. Bir dosyayı başka bir dosya olarak kopyalayacağı gibi birkaç dosyayı bir dizine de kopyalayabilir.

-i Eğer hedef dosya mevcut ise kullanıcıyı uyarır. Eğer kullanıcı ‘y’ tuşuna basarsa kopyalama gerçekleştirilir. ‘n’ tuşuna basarsa kopyalama iptal edilir.

-p Dosya özelliklerini korur. Korunan dosya özellikleri: değiştirilme zamanı, erişim zamanı, dosya bayrakları, dosya erişim modu, kullanıcı ID’si ve grup ID’si.

-r Dizini alt dizinlerle birlikte kopyalar.

-v Ekrana ne yaptığına dair bilgi basar.

```
$ cp -pv *.txt arsiv/
`11-01-2013.txt’ -> `arsiv/11-01-2013.txt’
`11-02-2013.txt’ -> `arsiv/11-02-2013.txt’
`ozet.txt’ -> `arsiv/ozet.txt’
$ cp –pr dizin1 dizin2
```

Temel Komutlar

mv

Bir dosyayı veya dizini başka bir dosya veya dizin olarak taşır. Veya birkaç dosya veya dizini başka bir dizine taşır. Dizin veya dosya adı değiştirilebilir.

$ mv kaynak hedef

rename

Dizin veya dosya adı değiştirilebilir.

rename 's/old/new/' files
rename [options] 's/old/new/' files

rename 's/perl/pl/' *.perl
rename -v 's/perl/pl/' *.perl

Temel Komutlar

cat

Dosyaları birleştirmek ve standart çıktıya basmak için kullanılır.

```
$ cat 1.txt
        Ben 1.dosyayim
$ cat 2.txt
        Ben 2.dosyayim
$ cat 1.txt 2.txt
        Ben 1.dosyayim
        Ben 2.dosyayim
```

Dosyalar komut satırında belirtildiği sırada okunur. Eğer dosya olarak eksi (-) karakteri kullanılırsa standart girişten okur (yani klavyeden yazmanızı bekler). Eğer dosya olarak UNIX domain soket verilmiş ise sokete bağlanır ve EOF(Dosya sonu, End Of File) görene kadar soketten okur.

-b parametresi boş olmayan satırları **–n** parametresi ise tüm satırları numaralandırır.

```
$ cat -b beni.oku
        1 Ben bir dosyayim.
        2 Bende toplam 3 satir vardir.
3 Iste bu da son satir.
$ cat -n beni.oku
        1 Ben bir dosyayim.
        2 Bende toplam 3 satir vardir.
        3
        4 Iste bu da son satir.
```

NOT: tac *komutu cat komutunun tam tersi standart çıktı basar*

Temel Komutlar

head

Verilen dosyanın ilk satırlarını görüntüler. **–n** ile ilk kaç satırın görüntülenmesi gerektiği belirtilir. Eğer satır sayısı verilmez ise ön tanımlı olarak ilk 10 satırı gösterir.

$ head –n 2 beni.oku
Ben bir dosyayim.
Bende toplam 3 satir vardir.

head ile dosya başından kaç byte okuyacağı belirtilebilir.
$ head -c 3 beni.oku
Ben

Eğer birden fazla dosya verilirse veriliş sırasına göre dosyaların ilk satırları gösterilir ve her dosyanın satırlarının başına '==> XXX <==' biçiminde ayraç koyar.

$ head -n 1 1.txt beni.oku
==> 1.txt <==
Ben 1.dosyayim
==> beni.oku <==
Ben bir dosyayim.

Temel Komutlar

tail

Dosyanın son kısımlarını gösterir. –c ile byte sayısı, -n ile satır sayısı verilebilir. Eğer dosya verilmez ise standart girişten okur.

$ cat beni.oku
Ben sevimli bir dosyayim.
Bende toplam 3 satir vardir.
Iste bu da son satir.

$ tail -c 7 beni.oku
Satir.

$ tail -n 1 beni.oku
Iste bu da son satir .

Eğer –f parametresi verilirse tail komutu EOF (dosya sonu) karakterini okuyunca durmaz. Yeni veriler için bekler ve veri hazır olur olmaz okuyup ekrana basar. Genellikle sürekli akan bir log dosyasını görüntülemek için kullanılır:

 # tail –f /var/log/messages

-r parametresi çıktıyı ters sırada basar.
$ tail -r beni.oku
Iste bu da son satir.
Bende toplam 3 satir vardir.
Ben sevimli bir dosyayim.

Temel Komutlar

wc

Kelime, satır, karakter ve byte sayar. Girdiyi standart girişten veya parametre olarak verilen dosyadan alır.
-c Karakter sayısını verir.
-l Satır sayısını verir.
-m Karakter sayısını verir.
-w Kelime sayısını sayar.

```
$ cat beni.oku
Ben bir dosyayim.
Bende toplam 3 satir vardir.
Iste bu da son satir.

$ cat beni.oku | wc –l
4
$ wc –w beni.oku
13
$ cat beni.oku | wc -c
70
```

Temel Komutlar

nl

Satır başlarına satır numarasını ekler.

```
$ cat a.txt
linux
101
pardusbusiness
academy
linux
101
$ nl a.txt
1 linux
2 101
3 pardus
4 business
5 academy
6 linux
7 101
```

Temel Komutlar

cut

Standart girişten okuduğu satırlar içindeki istenen sütunları gösterir. Sütun seçme işlemi bir ayraca göre yapılabileceği gibi sabit boyda da yapılabilir.

-c Karakter olarak seçilecek sütunları belirler.

-d Ayraç belirlemek için kullanılır.

-f Ayraç göre sütunlara ayrıldığında görüntülenmek istenen sütunları seçer.

Aşağıda /etc/password dosyasının bir kısmı görülmektedir. Görüldüğü gibi bilgiler ':' karakteri ile birbirinden ayrılmış. –d ile ayraç olarak ':' belirtildiğinde soldan sağa kullanıcı ismi 1. sütun, parola 2.sütun, UID 3. sütun, GID 4.sütun, açıklama 5.sütun, ev dizini 6.sütun ve kabuk 7.sütun olarak yerleşir. Örneğin kullanıcı adı ve kabuk bilgilerini ekrana yazmak için:

$ cat /etc/passwd

....
sshd:x:117:65534::/var/run/sshd:/usr/sbin/nologin
postfix:x:118:123::/var/spool/postfix:/bin/false
ahmet:x:1001:1001::/data/ahmet:/bin/sh
pardus:x:1004:1006::/home/pardus:/bin/sh

$ cut -d : -f1,7 /etc/passwd

....
sshd:/usr/sbin/nologin
postfix:/bin/false
ahmet:/bin/sh
pardus:/bin/sh

Temel Komutlar

expand, unexpand

Bu komut, giriş metnindeki sekmeleri(TAB) boşluklarla değiştirir.
$ expand dosya.txt > cikti.txt

unexpand tersine olarak boşluk karakterlerini sekmelere dönüştürür.
$ unexpand out

Temel Komutlar

fmt

Paragraf metnini yeniden biçimlendirir.
Bir satırda ön tanımlı olarak en fazla 75 karakter olacak şekilde biçimlendirme yapar.
$ fmt myfile.txt

-w: Satır genişliği
$ fmt –w 80 myfile.txt

s: Uzun satırları böler fakat kısaları uzunlara eklemez.
$ fmt –s myfile.txt

-u: Boşluklar tekdüze olsun. Kelimeler arası tek, cümleler arası iki boşluk.
$ fmt –u myfile.txt

Temel Komutlar

pr

Baskı için çıktıyı biçimlendirir, sayfalandırır. Satır veya sütün sayısını düzenleyen, kenar boşluklarını ayarlayan, satırları numaralandıran, sayfa başlığı ekleyen seçeneklere sahiptir.

Çıktıyı çift boşlukla biçimlendirme:
$ pr –d myfile.txt

Kenar boşluğu 5, satır boyu 65 olan sayfalar halinde çıktı üretmek için:
$ pr -o 5 --width=65 myfile.txt | more

Temel Komutlar

od

cat veya diğer komutlarla bir girdi görüntülendiğinde bazı zamanlar çıkışta anlamsız karakterler ve hizalama bozulmaları görür. Bunun sebebi okunabilir bir anlamı olmayan kontrol karakterleridir. Bunların ne olduğu cat benzeri komutlarla anlaşılmaz. Bu karakterlerin ne olduğunu görmek için od (Octal Dump) komutu kullanılır.
Od komutu öntanımlı olarak octal gösterime göre çıktı üretir. –A parametresi kullanılarak çıktı sayı sistemi o (octal, 8 tabanında), d (decimal, 10'luk tabanda), x (hexadecimal, 16'lık tabanda) belirtilebilinir. Od normal çalışmasında satır başlarına girdinin kaçıncı byte'ında olduğunu gösteren offset koyar. –n ile offset gösterimi iptal edilir. –t ile çıktı biçimlendirilebilir.

$ od text2
0000000 004471 066160 066565 031412 061011 067141 067141 005141
0000020 030061 060411 070160 062554 000012
0000031

Temel Komutlar

od

Aşağıdaki çıktıda offset bulunmaktadır ve kontrol karakterleri ters bölü (\) gösterimi ile yazılmıştır.

```
$ od -A d -t c text2
0000000 9  \t   p   l   u   m  \n   3  \t   b   a   n   a   n   a  \n
0000016 1   0  \t   a   p   p   l   e  \n
0000025
```

Aşağıdaki örnekte offset olmadığı gibi –t ile bazı kontrol karakterlerin yerine özel isimleri çıktıya konulmuştur. (nl: new line, ht: horizontal tab gibi.)

```
$ od -A n -t a text2
   9  ht   p   l   u   m  nl   3  ht   b   a   n   a   n   a  nl
   1   0  ht   a   p   p   l   e  nl
```

Çıktı genişliğini (offset) sınırlandırmak için –w kullanılır.

```
$ od -w1 -c -Ad input
0000000   1
0000001  \n
0000002   2
0000003  \n
0000004   3
0000005  \n
0000006   4
0000007  \n
0000008   5
0000009  \n
```

Temel Komutlar

paste

İki dosyayı satır satır veya sütun sütun birleştirmek için kullanılır.

```
$ cat 1.txt
GNU
Linux
Debian
Mint
Ubuntu
```

Hiçbir parametre verilmezse paste komutu cat komutu gibi davranır.

```
$ paste 1.txt
GNU
Linux
Debian
Mint
Ubuntu
```

Temel Komutlar

paste

Bütün satırları bir dosyada birleştirmek için:

```
$ paste -s 1.txt
GNU Linux	Debian	Mint Ubuntu
```

Birleştirmeyi bir ayraçla yapmak için:

```
$ paste -d: -s 1.txt
GNU:Linux:Debian:Mint:Ubuntu
```

Üç sütunlu bir çıktıda birleştirmek için:

```
$ paste - - - < 1.txt
GNU Linux
Debian
Mint Ubuntu
```

```
$ paste - - -d: < 1.txt
GNU:Linux
Debian:Mint
Ubuntu:
```

Temel Komutlar

paste

İki dosyayı karşılıklı (sütun sütuna) birleştirmek için:

$ cat 2.txt

1 A
2 B
3 C
4 D

$ paste 1.txt 2.txt

GNU 1 A
Linux 2 B
Debian 3 C
Mint 4 D
Ubuntu

İki dosyayı satır satır birleştirmek için \n ayracı kullanılır:

$ paste -d'\n' 1.txt 2.txt

GNU
1 A
Linux
2 B
Debian
3 C
Mint
4 D
Ubuntu

Temel Komutlar

join

Paste gibi iki dosyayı birleştirmekle birlikte biraz daha ileri bir komut olan join, iki dosyadaki sütunları eşleştirerek birleştirir. Hiç bir parametre verilmediğinde aşağıdaki gibi bir sonuç ortaya çıkar:

$cat A.txt
A
B
C

$cat B.txt
1 Ali
2 Yusuf
3 Selim

$join A.txt B.txt
1 A Ali
2 B Yusuf
3 C Selim

Temel Komutlar

join (Alan seçme)

Dosyalardaki ortak alanlar kullanılarak birleşme yapılabilir.
Aşağıdaki örnekte 1.dosyanın 1.sütunu ile 2.dosyanın 1.sütunu örtüşecek şekilde dosyalar birleştirilir.

```
$ cat 1.txt
06 ANKARA 1
21 DIYARBAKIR 2
34 ISTANBUL 3
61 TRABZON 4

$ cat 2.txt
06 CANKAYA
21 KULP
34 USKUDAR
61 AKCAABAT

$ join -1 1 -2 1 1.txt 2.txt
06 ANKARA 1 CANKAYA
21 DIYARBAKIR 2 KULP
34 ISTANBUL 3 USKUDAR
61 TRABZON 4 AKCAABAT
```

Görüldüğü gibi plaka numaraları ortak alan olarak kullanılarak illerle ilçeler eşleştirilmiş oldu.

Temel Komutlar

join (Ayraç Belirtme)

Yukarıdaki örneklerde alanlar (diğer deyişle sütunlar) birbirinden boşluk ile ayrılmıştır. Boşluk ve TAB öntanımlı ayraç olduğundan join komutu sütunları bu ikisinden hangisi varsa ona göre ayırır ve numaralandırır. Eğer sütunlar başka bir ayraçla ayrılmışsa (virgül, noktalı virgül, iki nokta üst üste gibi) –t parametresi kullanılarak ayraç belirtilebilir.

```
$ join -t, -1 3 -2 1 A.txt B.txt
1,Ali,A,A
2,Yusuf,B,B
3,Selim,C,C
```

Temel Komutlar

join (Harf Duyarlılığı)

Birleştirme için eşleştirme yapılırken normalde küçük harf ile büyük harf eşleşmez. Örneğin bir dosyadaki A ile diğerindeki a karakteri eşleşmez. Büyük-küçük harf duyarlılığını iptal etmek için –i parametresi kullanılır.

join (Çıktı Biçimlendirme)

join normalde dosyalardaki sıraya göre satırları ve sütunları sıralar. Sadece sıralama kriteri olan ortak sütunu başa alır. Alanları istenilen sırada çıktıya yazmak için –o parametresi kullanılır.

Aşağıdaki örnekte 1.dosyanın 1.sütunu ve 2.dosyanın 1.sütunu sırasıyla çıktı oluşur.

```
$ join -o 1.1 2.1 -1 2 -2 2 A.txt B.txt
Ali 1
Yusuf 2Selim 3
```

Temel Komutlar

split

split komutu dosyayı iki veya daha fazla parçaya bölmek için kullanılır.
-l parametresi ile satır sayısına göre **–b** parametresi ile byte miktarına göre parçalanır.

```
$ split split.zip
$ ls
split.zip xab xad xaf xah xaj xal xan

$ wc -l *
40947 split.zip
1000 xaa
1000 xab
1000 xac
```

Görüldüğü gibi öntanımlı olarak 1000 satırlı dosyalara böler. –l parametresi ile bölünmüş dosyaların satır sayısı değiştirilebilir. Veya –b parametresi ile dosya boyutu verilebilir.

```
$ split -l 3 a.txt dosya
$ ls dosya*
dosyaaa dosyaab dosyaac
$ cat dosyaaa
linux
101
Pardus
```

```
$ cat dosyaab
business
academy
Linux
$ cat dosyaac
101
```

Temel Komutlar

sort

sort komutu dosyadaki satırları sıralamak için kullanılır.

-n: sayısal(numeric) sıralama yapar.
-f: Büyük/küçük harf ayrımı olmadan sıralar.
-r: sıralama sonucunu tersler listeler.

```
$ cat a.txt
Linux
101
pardus
business
academy
```

```
$ sort a.txt
101
Academy
business
Linux
pardus
```

```
$ sort -n a.txt
Academy
business
Linux
pardus
101
```

Temel Komutlar

uniq

uniq komutu birden fazla olan aynı satırları tek satıra düşürmek için kullanılır.
Genellikle sort komutu ile sıralama yapıldıktan sonra aynı çifte kayıtları silmek için kullanılır.

```
$ sort a.txt
101
101
academy
business
Linux
Linux
pardus

$ sort a.txt |uniq
101
Academy
business
Linux
pardus
```

Temel Komutlar

GÇ Yönlendirmeleri

komut < dosya1

> dosya1 içeriği komuta girdi verir

komut1 < (komut2)

> *komut2* çıktısı, *komut1* girdi verir

komut > dosya1

> *komut* çıktısı dosyanın içeriğine yazılır

komut > /dev/null

> *komut* çıktısını yazdırmaz

komut >> dosya1

> *komut* çıktısı dosyanın sonuna ekler

komut 2> dosya1

> *komut* hata çıktısını *dosya1*'e yazdırır

komut &> dosya1

> Komut tüm çıktılarını *dosya1*'e yazdırır

Özel İşaretler

komut1 | komut2

> *komut1*'in çıktısını *komut2*'ye girdi alır

komut1 |& komut2

> *komut1*'in hata çıktısı *komut2*'ye girdi

Temel Komutlar

Komut Listeleri	
komut1; komut2	Önce komut1, sonra komut2 çalışır
komut1 && komut2	komut1 başarılı çalışırsa komut2 çalıştır
komut1 || komut2	komut1 başarısızsa komut2 çalıştır
komut1 &	Alt kabukta komut1 çalıştır

Temel Komutlar

ln

Sembolik ve Hard Bağlantılar
Dosyalar ve dizinler arasında link oluşturmak için ln komutu kullanılı. Bu komut, orijinal dosya ile aynı yetkilendirmeye sahip yeni bir dosya kaydı oluşturur. Link oluşturma genelde bulunan dosya sisteminde disk alanı yetersiz hale geldiğinde dosyaları başka bir dosya sistemine taşıdıktan sonra eski dosya/dizin yolunu kullanan kişi ve uygulamaların hala eski yol üzerinden erişmeleri için kullanılır. İki çeşit bağlantı vardır. Sıkı (hard) bağlantılarda yapılan değişikler orijinal dosyayı etkiler. Sembolik (soft) bağlantı sadece orijinal dosyayı gösteren bir isimdir. MS Windows sistemlerindeki karşılığı kısa yol oluşturmaktır. Sembolik bağlantının silinmesi orijinal dosyayı silmez. Orijinal dosya silinirse de sembolik link işlevsiz hale gelecektir. Sembolik link üzerinden dosyaya erişilemez. Sembolik link oluşturmak için –s parametresi kullanılır. Hiç parametre kullanılmadan kullanılırsa hard link oluşturulmuş olur.

```
# ln seçenek mevcut_dosya/dizin olusturulacak_bağlantı
root@pardus:~# ln  /data/pardus /var/data/pardus
```

Yukarıdaki komut ile /var/data/pardus dizinine yapılan tüm erişimler aslında /data/pardus dizinde uygulanacak. Hard linkte ise orijinal dosya ile link oluşturulan dosyanın birebir aynısı iki farklı dosya oluşur. Birisinde yapılan işlem diğerine de yansır. Bu durumu disklerde kullanılan RAID-1 teknolojisine benzetebiliriz. Orijinal dosya silinse bile verilere diğer dosya üzerinden erişilebilir. Katı linkteki bu özellik Linux inode yapısındaki dosyaya yapılan link sayısı özelliği ile sağlanmaktadır. Dosyanın biri silindiğinde link sayısı değeri bir azaltılmakta. Diğer dosya(lar) silindiğinde ise link sayısı sıfır olduğunda dosya tamamen silinmektedir. Dolayısıyla hard link dosya sistem ile doğrudan alakalıdır. Farklı dosya sistemlerinin(disk bölümleri) farklı inode yapısı olduğu için hard link farklı dosya sistemleri arasında yapılamaz. Sadece aynı dosya sistemi üzerinde yapılabilir. Farklı dosya sistemleri arasında link oluşturulacaksa sadece sembolik link kullanılabilir.

Temel Komutlar
Dosya içerik arama işlemleri (grep)

grep

https://tr.wikibooks.org/wiki/Linux_%C4%B0%C5%9Fletim_Sistemi/Linux_Komutlar%C4%B1/D%C3%BCzenli_ifadeler

Grep komutu Linux' ta dosya içerisinde arama yapmak için kullanılan bir komuttur. Bu komut ile istediğimiz herhangi bir stringin hangi dosyalar içerisinde geçtiğini bulabiliriz. Öncelikle içinde arama yapacağımız herhangi bir dosya oluşturalım. Dosya oluşturmak için çeşitli komutlar var biz bunlardan NANO komutunu kullanacağız:

$nano dosya1.txt

Bunu çalıştırdığımız zaman karşımıza editör çıkacaktır. Buraya ben aşağıdaki satırları yazıyoruz:

THIS LINE IS THE 1ST UPPER CASE LINE IN THIS FILE.

this line is the 1st lower case line in this file.

This Line Has All Its First Character Of The Word With Upper Case.

Two lines above this line is empty.

And this is the last line.

1. İstenilen string' i tek bir dosyada arama

Aşağıdaki komut çalıştığı zaman bize içerisinde this geçen satırları verir.

$ grep "this" dosya1.txt

this line is the 1st lower case line in this file.

Two lines above this line is empty.

And this is the last line.

Temel Komutlar
Dosya içerik arama işlemleri (grep)

2. İstenilen string' i birden fazla dosya içerisinde arama

Aşağıdaki komut ile biz istediğimiz string' i tek bir dosya içinde değil de birden fazla dosya içerisinde arama yapmamıza imkan sağlar. Bunun için aynı dosya içeriğini

başka bir dosyaya kopyalayalım:

```
$cp dosya1.txt dosya2.txt
```

Kopyalama işleminden sonra şimdi komutumuza geçelim:

```
$ grep “this” *.txt
dosya1.txt:this line is the 1st lower case line in this file.
dosya1.txt:Two lines above this line is empty.
dosya1.txt:And this is the last line.
dosya1.txt:this line is the 1st lower case line in this file.
dosya1.txt:Two lines above this line is empty.
dosya1.txt:And this is the last line.
```

Temel Komutlar
Dosya içerik arama işlemleri (grep)

3. Harfe duyarlılığı kaldırarak arama yapmak

Öndeki komutlarımızda arama yaptığımızda büyük küçük harf ayrımına dikkat ediyordu. Ama biz böyle bir ayrımı ortadan kaldırmak istersek şöyle kullanmamız gerekmektedir:

$ grep -i “the” dosya1.txt

THIS LINE IS THE 1ST UPPER CASE LINE IN THIS FILE.

this line is the 1st lower case line in this file.

This Line Has All Its First Character Of The Word With UpperCase.

And this is the last line.

4. Regular Expression (Düzenli İfade) arama

Regular Expression bir ifade anlamına gelmektedir. Yani biz dosya içerisinde herhangi bir kelime değil de bir ifade aramak isteyebiliriz.

Örneğin “is” ile başlayıp “ine” ile devam eden bir ifadenin olduğu satırları bulmak istediğimizde komutumuzu aşağıdaki gibi kullanmalıyız:

$ grep “is.*ine” dosya1.txt

this line is the 1st lower case line in this file.

This Line Has All Its First Character Of The Word With

Sonucun ilk satırında ilk is olan yeri değil de son is olan yeri aldığına dikkat edelim.

Temel Komutlar
Dosya içerik arama işlemleri (grep)

5. Tam kelime eşleşmesini aramak

Şimdiye kadar gördüklerimizde sadece istenilen kısmın eşleşmesi yetiyordu bunun kelimenin tamamı olup olmadığının bir önemi yoktu.

Fakat biz kelimenin birebir eşleşmesini istersek aşağıdaki gibi bir kullanım yapmamız gerekmektedir.

$ grep -iw “is” dosya1.txt

THIS LINE IS THE 1ST UPPER CASE LINE IN THIS FILE.

this line is the 1st lower case line in this file.

Two lines above this line is empty.

And this is the last line.

Temel Komutlar
Dosya içerik arama işlemleri (grep)

6. Aranan string' den önceki (B) sonraki (A) ya da etrafındaki (C) satırların da getirilmesi

6.1. Eşleşmeden sonraki N satırı da getirmek için

Öncelikle buna uygun bir dosya3.txt adında bir dosya oluşturalım. İçeriğine de şunları yazalım:

WORD – WORD consists of a sequence of non-blank characters, separated with white space.

word – word consists of a sequence of letters, digits and underscores.

Example to show the difference between WORD and word

* 192.168.1.1 – single WORD

* 192.168.1.1 – seven words.

Dosyamızı oluşturduktan sonra şimdi komutumuza geçelim (aradığımız string' den sonraki istediğimiz kadar satırı bize getirir.):

$ grep -A 3 -i “example” dosya3.txt

Example to show the difference between WORD and word

* 192.168.1.1 – single WORD

* 192.168.1.1 – seven words.

Temel Komutlar
Dosya içerik arama işlemleri (grep)

6.2. Eşleşmeden önceki N satırı da getirmek için

Aşağıdaki komut da aradığımız string' den önce görüntülemek istediğimiz satır sayısı için kullanılır:

$ grep -B 2 “single WORD” dosya3.txt

Example to show the difference between WORD and word

* 192.168.1.1 – single WORD

6.3. Eşleşmenin etrafındaki N satırı getirmek için

Aşağıdaki komut da bize eşleşmeden önce ve sonra kaç satır görüntüleyeceğimizi sağlar:

$ grep -C 2 “Example” dosya3.txt

word – word consists of a sequence of letters, digits and underscores.

Example to show the difference between WORD and word

* 192.168.1.1 – single WORD

Temel Komutlar
Dosya içerik arama işlemleri (grep)

7. Rekürsif olarak tüm dosyalarda arama

Eğer bir dizin altındaki tüm dizinleri de aramaya dahil etmek istiyorsak bunun için kullanım aşağıdaki şekilde olmalıdır:

$ grep -r “aranacak_ifade” *

Burada komutun sonundaki yıldız işareti tümünü temsil ediyor.

8. Eşleşmelerin sayısını bulmak

Aradığımız ifadenin kaç satırda eşleştiğini bulmak için de aşağıdaki komutu kullanıyoruz.

$ grep -c “of” dosya3.txt

2

9. Sadece hangi dosyalarda eşleşme olduğunu bulmak

Bu kullanımda da aradığımız ifadenin geçtiği dosyaların sadece isimlerini bize verir.

$ grep -l “this” dosya*

dosya1.txt

dosya2.txt

10. Sadece eşleşen stringi gösterme

Sadece aradığımız ifadeyle eşleşen string’ i görüntülemek istediğimiz zaman komutu aşağıdaki gibi kullanmalıyız:

$ grep -o “is.*line” dosya1.txt

is line is the 1st lower case line

is line

is is the last line

Temel Komutlar
Dosya, dizin arama işlemleri

locate (updatedb), find, which, whereis

locate

find komutu arama yaparken her seferinde belirtilen dizinde tarama yaptığı için çok fazla dosya olan sistemlerde yavaş çalışabilir. Daha hızlı dosya aramak için locate komutu kullanılabilir. Bu komut dosya adı aramalarını direkt dosya sisteminde tarama yapmak yerine özel bir veritabanından getirmektedir. Locate komutu aramayı /var/cache/locate/locatedb veritabanı dosyasında yapmaktadır. Bu dosya locate dışında ayrı olarak çalışan updatedb komutu tarafından güncellenmektedir. Eğer en güncel dosyalarında aramaya dahil olmasını istiyorsanız belirli aralıklarla updatedb komutu çalıştırılmalıdır. Her gece veritabanı dosyasını güncellemek için Pardus tarafından otomatik olarak /etc/cron.daily/locate dosyasındaki cron işi çalıştırılmaktadır. Arama yapmak için dosyanın adı veya adının geçtiği kelimeleri yazmak yeterlidir.

```
root@pardus:/ # locate apache
/etc/apache2
/etc/apache2/apache2.conf
/etc/apache2/conf.d
/etc/apache2/conf.d/charset
/etc/apache2/conf.d/javascript-common.conf
/etc/apache2/conf.d/localized-error-pages
```

Temel Komutlar
Dosya, dizin arama işlemleri

find

Dosya ve dizin aramak için kullanılır.
Kullanımı:
find dizin seçenekler
Dizin adı belirtilmezse bulunulan dizin ifade edilir.

Seçenekler:
-name isim: Aranılacak dosyanın ismi.
-iname isim: Aranılacak dosya ismi büyük/küçük harf duyarsız
-type tip: Aranan dosyanın tipini belirler. Tip aşağıdaki değerlerden biri olabilir.
f: Normal Dosya
d: Dizin

Temel Komutlar
Dosya, dizin arama işlemleri

which

$PATH değişkeni içerisinde tanımlı çalıştırılabilir komutların tam yolunu gösterir. Sadece komutlar ve shell scriptler için çalışır.

```
# which cut
/bin/cut
```

whereis

Parametre olarak verilen program, kaynak kodu ve yardım dosyalarının yerlerini gösterir.

```
# whereis pwd
pwd: /bin/pwd /usr/include/pwd.h /usr/share/man/man1/pwd.1.gz
```

Temel Komutlar
Dosyaların incelenmesi ve düzenlenmesi

Dosyaların İncelenmesi ve Düzenlenmesi		
vi	<dosya1>	
	ESC + :w	Değişiklikleri kaydeder
	ESC + :wq	Değişiklikleri kaydedip çıkar
	ESC + :q	Değişik yapılmadıysa çıkar
	ESC + :q!	Değişiklikleri kaydetmeden çıkmaya zorlar
	a	Bir sonraki karakterden yazmaya başlar
	i	Bulunduğu yerden yazmaya başlar
	dd	Tüm satırı siler
nano	<dosya1>	
	CTRL + X	Çıkış
	CTRL + W	Arama
	CTRL + K	Kes
	CTRL + U	Yapıştır
more	<dosya1>	dosya1 içeriğini görüntüler
	enter	Satır satır ilerler
	space	Sayfa sayfa ilerler
	b	Bir önceki sayfa
	q	Çıkış
less	<dosya1>	dosya1 içeriğini görüntüler
	aşağı/yukarı yön tuşu	Satır satır ilerler
	space	Bir sonraki sayfa
	b	Bir önceki sayfa
	q	Çıkış

Temel Komutlar

Dosya İsmi Örüntüleri

Linux altında dosya isimleri açıkça yazılabildiği gibi belirli kurala uyan dosyaların tamamını ifade etmek üzere örüntüler de kullanılabilir.

? Herhangi bir tek karakteri gösterir.

hd?, hda, hdb, hdc, hdd gibi hd ile başlayıp devamın tek karakter olan bütün dosya isimleriyle eşleşir.

***** Hiç veya herhangi sayıdaki karakteri gösterir.

*1.jpg, dosyaadı 1 olan veya 1 ile biten tüm jpg uzantılı dosyalarla eşleşir.

[ ] Aralık gösterir.

a[1-9], a1, a2...a9 isimleriyle eşleşir.
a[6,8], a6 ve a8 isimleriyle eşleşir.

{ } Virgülle ayrılmış seçenekleri gösterir.

{*.doc,*.pdf} .doc ve .pdf dosyaları gösterir.

**** Özel karakterleri korumak için kullanılır.

```
$ ls *.jpg  # Uzantısı jpg olan tüm dosyalar
$ ls ?.jpg  # Dosya adı tek karakter olan jpg dosyalar
$ rm [A-Z]*.pdf  # Büyük harfle başlayan pdf dosyalar
```

Temel Komutlar

file

Dosya türünü belirlemek için kullanılır. /usr/share/file/magic dosyasındaki dosya türlerinin imzası olan sihirli numaraları kullanır.

```
$ file test.tar.gz
test.tar.gz: gzip compressed data, deflated,
last modified: Sun Sep 16 13:34:51 2001, os: Unix

$ file -z test.tar.gz
test.tar.gz: GNU tar archive (gzip compressed data, deflated,
last modified: Sun Sep 16 13:34:51 2001, os: Unix)
```

Temel Komutlar

tee

Komut çıktısını hem dosyaya hem de Stdout'a yazar.
-a:
Dosyanın sonuna ekleme yapar, mevcut dosyayı sıfırlamaz.
Aşağıdaki komut dizisinde ls'in çıktısı hem dosyaya hem de Stdout'a yazılır.
$ ls | tee file.txt
Stdout'a yazdığı için Stdin'den okuyan herhangi bir komuta da çıktıyı gönderebilir.
Aşağıdaki komut dizisinde çıktı hem dosyaya yazılır hem de sort komutuna girdi olarak geçilir.

$ ls | tee file.txt | sort

Çıktı birden fazla dosyaya yazılabilir:

$ ls | tee file1.txt file2.txt

Temel Komutlar

xargs

Bir komutu parametrelerini standart girdiden(stdin) alıp başka komutlara geçebilecek şekilde çalıştırır. Böylece bir defada işleyebileceği parametreden daha fazlası komuta aktarılabilir. Genelde ls veya find komutu tarafından üretilmiş uzun dosya listesini, bu dosyaların herbirinde tek tek işlem yapacak bir başka komuta parametre olarak geçmek için kullanılır.
Örnek: Bir patterni bütün dosyalarda aramakta için:
 # find /etc | xargs grep pattern

Xargs'ın en uygun kullanım yerlerinden birisi de bir dizindeki belli dosyaları silmektir. Bunlar tek satırda rm komutuna parametre olarak verilemediğinden xargs'a ihtiyaç vardır:
$ ls "*.tmp" | xargs rm

-n: Çalıştırılacak komuta eklenecek parametre sayısı
Örneğin diff komutu iki dosyayı karşılaştırır. Mevcut dizindeki dosyaları ikişer ikişer gruplar halinde karşılaştırmak için:
$ ls | xargs –n2 diff
$ echo 1 2 3 4 | xargs -n 2
1 2
3 4
xargs mevcut komutun parametrelerini tamamlamak için de kullanılabilir:

Temel Komutlar
İçerik manipilasyonları

sed, awk, tr, vi

sed

Sed (veya GNU versiyonu olan gsed) , oldukça yetenekli bir yazı düzenleme editörüdür. Tek komutla sizi büyük bir iş yükünden kurtarabilir, işte bazı kullanım örnekleri:

BOŞLUK KOYMA

İçinde "deneme" geçen satırların üzerinde bir boş satır oluştur:

$sed '/deneme/{x;p;x;}'

İçinde "deneme" geçen satırların altında bir boş satır oluştur:

$sed '/deneme/G'

İçinde "deneme" geçen satırların hem üzerinde hem altında bir boş satır oluştur:

$sed '/deneme/{x;p;x;G;}'

Her 5 satırdan sonra bir boş satır ekle:

$gsed '0~5G' (GNU sed)

$sed 'n;n;n;n;G;' (Diğer sed sürümleri)

Temel Komutlar
İçerik manipilasyonları

Sed komutu

a\	Aktif satırın aşağısına ekleme yapma.
c\	Aktif satırı verilen cümle/kelime ile değiştirme.
d	Yazıyı silme.
i\	Aktif satırın yukarısına ekleme yapma.
p	Yazıyı gösterme, yazdırma.
r	Dosya okuma.
s	Bul ve değiştir.
w	Bir dosyaya yaz.

Sed opsiyonları:

-e script	Komut satırında birden fazla sed opsiyonu kullanma.
-f	Sed komutlarının olduğu bir script dosyasını sed ile beraber çalıştırmak.
-n	Sessiz mod.
-V	Versiyon bilgilerini göster ve çık.

Temel Komutlar
İçerik manipilasyonları

sed

NUMARALANDIRMA

Her satırın başına satır numarası yaz, araya bir "tab" boşluk koy:

sed = dosya_adı | sed 'N;s/\n/\t/'

Her satırın başına satır numarası yaz ancak eğer satır boşsa numara görünmesin:

sed '/./=' dosya_adı | sed '/./N; s/\n/ /'

Satır sayısını yazdır

sed -n '$='

Temel Komutlar
İçerik manipilasyonları

sed

YAZI DÖNÜŞTÜRME

UNIX altında Windows satır sonlarını (CR/LF) Unix formatına dönüştür:

sed 's/.$//'

Windows/Dos altında Unix satır sonlarını Windows formatına dönüştür:

sed "s/$//"□

her satırın önündeki boş alanı (boşluk, tab) kaldır:

sed 's/^[\t]*//'

Her satırın sonundaki boş alanı (boşluk, tab) kaldır:

sed 's/[\t]*$//'

Her satırın başına 5 boşluk karakteri koy:

sed 's/^/ /'

Yazıyı 79-sütunluk genişliğin ortasına taşı (yöntem 1'de satır başındaki boşluklar önemsizdir, yöntem 2'de onlar da taşınır):

sed -e :a -e 's/^.\\{1,77\\}$/ & /;ta' (yöntem 1)

sed -e :a -e 's/^.\\{1,77\\}$/ &/;ta' -e 's/\\(*\\)\1/\1/' (yöntem 2)

Temel Komutlar
İçerik manipilasyonları

Yazıdaki "foo"ları bul ve "bar" olarak değiştir:

sed 's/foo/bar/' (satırdaki ilk "foo"yu değiştirir)

sed 's/foo/bar/4' (satırdaki 4. foo'yu değiştirir)

sed 's/foo/bar/g' (satırdaki tüm foo'ları değiştirir)

sed '/baz/s/foo/bar/g' (yalnızca "baz" kelimesini bulduğu satırlarda "foo"ları "bar" olarak değiştirir)

sed '/baz/!s/foo/bar/g' (satırda "baz" kelimesi varsa işlem yapmaz, yoksa "foo"ları "bar" olarak değiştirir)

Satırları başaşağı et; ilk satır sona, son satır başa:

sed '1!G;h;$!d' (yöntem 1)

sed -n '1!G;h;$p' (yöntem 2)

Her satırı ters çevir (ahmet > temha):

sed '/\n/!G;s/\(.\)\(.*\n\)/&\2\1/;//D;s/.//'

Eğer bir satır "\" ile biterse, altındaki satırı ona ekle:

sed -e :a -e '/\\$/N; s/\\\n//; ta'

Eğer bir satır "=" ile başlıyorsa, onu önceki satırın sonuna ekle, "=" işaretini de boşluk ile değiştir

sed -e :a -e '$!N;s/\n=/ /;ta' -e 'P;D'

Temel Komutlar
İçerik manipilasyonları

KIRPMA

Dosyanın ilk 10 satırını yazdır:

sed 10q

Dosyanın son 10 satırını yazdır:

sed -e :a -e '$q;N;11,$D;ba'

Sadece içinde "deneme" bulunan satırları yazdır:

sed -n '/deneme/p' (yöntem 1)

sed '/deneme/!d' (yöntem 2)

Sadece içinde "deneme" bulunmayan satırları yazdır:

sed -n '/deneme/!p' (yöntem 1)

sed '/deneme/d' (yöntem 2)

Sadece içinde "deneme" bulunan satırın bir üstündeki satırı yazdır:

sed -n '/deneme/{g;1!p;};h'

Sadece içinde "deneme" bulunan satırın bir altındaki satırı yazdır:

sed -n '/regexp/{n;p;}'

İçinde AAA, BBB ve CCC olan satırları yazdır (sıra gözetme):

sed '/AAA/!d; /BBB/!d; /CCC/!d'

İçinde sırasıyla AAA, BBB ve CCC olan satırları yazdır:

sed '/AAA.*BBB.*CCC/!d'

İçinde AAA veya BBB veya CCC bulunan satırları yazdır:

sed -e '/AAA/b' -e '/BBB/b' -e '/CCC/b' -e d (Diğer sed'ler)

gsed '/AAA\|BBB\|CCC/!d' (GNU sed)

AAA içeren paragrafları yazdır (paragraf=boş satırlar arası):

sed -e '/./{H;$!d;}' -e 'x;/AAA/!d;'

İçinde AAA veya BBB veya CCC bulunan paragrafları yazdır

sed -e '/./{H;$!d;}' -e 'x;/AAA/b' -e '/BBB/b' -e '/CCC/b' -e d

gsed '/./{H;$!d;};x;/AAA\|BBB\|CCC/b;d' (GNU sed)

65 karakter veya daha uzun olan satırları yazdır:

sed -n '/^.\{65\}/p'

65 karakterden daha kısa olan satırları yazdır:

sed -n '/^.\{65\}/!p' (yöntem 1)

sed '/^.\{65\}/d' (yöntem 2)

Temel Komutlar
İçerik manipilasyonları

SİLME

Iowa ve Montana değerleri arasındakileri sil:

sed '/Iowa/,/Montana/d'

Birbirinin aynısı olan satırları sil (sadece ilk örnekleri kalır):

sed '$!N; /^\(.*\)\n\1$/!P; D'

Sadece birbirinin aynısı olan satırlar kalsın:

sed '$!N; s/^\(.*\)\n\1$/\1/; t; D'

Yazının ilk 10 satırını sil:

sed '1,10d'

yazının son satırını sil:

sed '$d'

Yazının son 10 satırını siler:

sed -e :a -e '$d;N;2,10ba' -e 'P;D' (yöntem 1)

sed -n -e :a -e '1,10!{P;N;D;};N;ba' (yöntem 2)

Her 8. satırda bir sil:

gsed '0~8d' (GNU sed)

sed 'n;n;n;n;n;n;n;d;' (Diğer sed'ler)

Yazıdaki tüm boş satırları sil:

sed '/^$/d' (yöntem 1)

sed '/./!d' (yöntem 2)

Yazının başındaki boş satırları sil:

sed '/./,$!d'

Yazının sonundaki boş satırları sil:

sed -e :a -e '/^\n*$/{$d;N;ba' -e '}'

Temel Komutlar
İçerik manipilasyonları

awk

awk, güçlü bir desen arama aracıdır. Sed gibi benzerlerinden ayıran özellik ise matematik işlemler, şartlı ifadeler, değişkenler ve dosya G/Ç işlemleri gibi script dillerinde olan temel işlevlere sahip olmasıdır. Genelde yapılandırma dosyalarını okumak veya komut çıktılarını işlemek için kullanılır. Dosyalarla işlem yapmak için –f parametresi kullanılır. Çağrılma şekli:

$ awk –f awk_script dosya

$ awk desen {aksiyon} dosya

Eğer desen belirtilmezse bütün satırlar eşleşir. Sonda verilen dosya parametresi yerine metin akımı üzerinde de işlem yapabilir.

Belirtilen dosyadaki 2 ve 3 numaralı sütunu ekrana basmak için (sütunlar $2 ve $3 ile belirtildi):

$ awk < dosya '{ print $2, $3 }'

Bash ile biten satırları göstermek için:

$ awk ‘/bash$/’ /etc/passwd

Temel Komutlar
İçerik manipilasyonları

awk

Awk ön tanımlı olarak boşluk ve Tab'a göre sütunlara ayırır. Bunu değiştirmek için –F parametresi kullanılır.
Bash ile biten satırlardaki birinci kolonu (kullanıcı adı) gösterir. Komuttaki $1, birinci kolonu gösteriyor. /etc/passwd dosyasında sütunlar : ile ayrıldığından ayraç olarak –F ile tanımlanmıştır.

```
$ awk < /etc/passwd -F: '{ print $6 }'root
mysql
cyrus
asterisk
spamfilter
user1
```

awk ile matematik işlemler yapılabilir. Aşağıdaki örnek Fahrenheit'ten Celsius'a sıcaklık çevrimi yapar:

```
$ awk '{ print ($1-32)*(5/9) }'
```

Temel Komutlar
İçerik manipilasyonları

tr

Tekrar eden karakterleri değiştirmek, silmek veya sıkıştırmak için kullanılır.
STDIN'den okur, STDOUT'a yazar.
$ tr [seçenek] SET1 [SET2]
Hem SET1, hem de SET2 belirtilmişse SET1'deki her bir karakter karşılık gelen SET2'deki karakterle değiştirilir. Örnekler:
1. Küçük harfleri büyük harflerle değiştirmek:
$ tr abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ
pardus
PARDUS
Aynı iş şu şekilde de yapılabilir:
$ tr a-z A-Z
pardus
PARDUS
2. Süslü parantezleri normal parantezlere dönüştürme:
$ tr '{}' '()' < girişdosyası > çıkışdosyası

Temel Komutlar
İçerik manipilasyonları

tr

3. Boşlukları TAB karakterine dönüştürme:
$ tr [:space:] '\t'
Pardus Linux
Pardus
Linux

4. Tekrar eden karakterleri –s ile sıkıştırma:
Birden fazla boşluğu tek boşluğa indirme:
$ echo "GNU
GNU Linux
Linux" | tr -s [:space:] ' '

5. Belirtilen karakterleri –d ile slime:
$ echo "1 GNU 2 Pardus 3 Linux" | tr -d [:digit:]
GNU
Pardus
Linux

Temel Komutlar

Donanım Bilgisi Toplama	
dmidecode -t <donanım>	Donanım bileşenleri ile ilgili bilgi
lscpu	İstemci bilgisini
lspci	PCI aygıtlarını listeler
lsusb	USB denetleyicisi bilgileri
df	Disk kullanımı ayrıntılı bilgi
du <dizin1>	dizin1'in diskteki boyutu

Teşekkürler....