



Linux - I

I. kısımda Linux'a giriş ve dosya-dizin işlemleri ele alınmıştır.

Giriş

Linux Nedir?

Linux, 1991 yılında Linus Torvalds tarafından geliştirilen ve hala gelişmeye devam eden, Unix mimarisine dayanan ücretsiz ve açık kaynaklı bir işletim sistemidir (OS). O zamandan beri küresel bir fenomene dönüßen Linux, süper bilgisayarlardan sunuculara, cep telefonlarından kişisel bilgisayarlara kadar her şeyi güçlendirmektedir. Stabilitesi, güvenliği ve esnekliği ile tanınan Linux, hem kişisel hem de profesyonel kullanım için popüler bir seçenekdir.

Linux Nasıl Çalışır?

Linux, Windows veya macOS gibi bir işletim sistemidir ancak nasıl çalıştığı ve ücretsiz, açık kaynak doğası açısından farklıdır. Kalbinde Linux çekirdeği bulunur, bu da sistemin temel parçasıdır. Çekirdek (kernel), bilgisayarın donanımını, CPU,

bellek ve çevre birimleri gibi yönetmekten sorumludur ve tüm yazılım uygulamalarının fiziksel donanımla etkileşimde bulunmasını sağlar.

Çekirdek, yazılım uygulamaları ile bilgisayarın donanımı arasında bir köprü görevi görerek çalışır. Bir yazılım uygulaması, bir dosyayı kaydetmek veya ekranda bir şey göstermek gibi donanımla ilgili bir şey yapmak istediğiinde, bir istek çekirdeğe gönderir. Çekirdek, bu isteği donanımın anlayacağı talimatlara çevirir.

Linux Dosya Sistemi Hiyerarşisi (FHS)

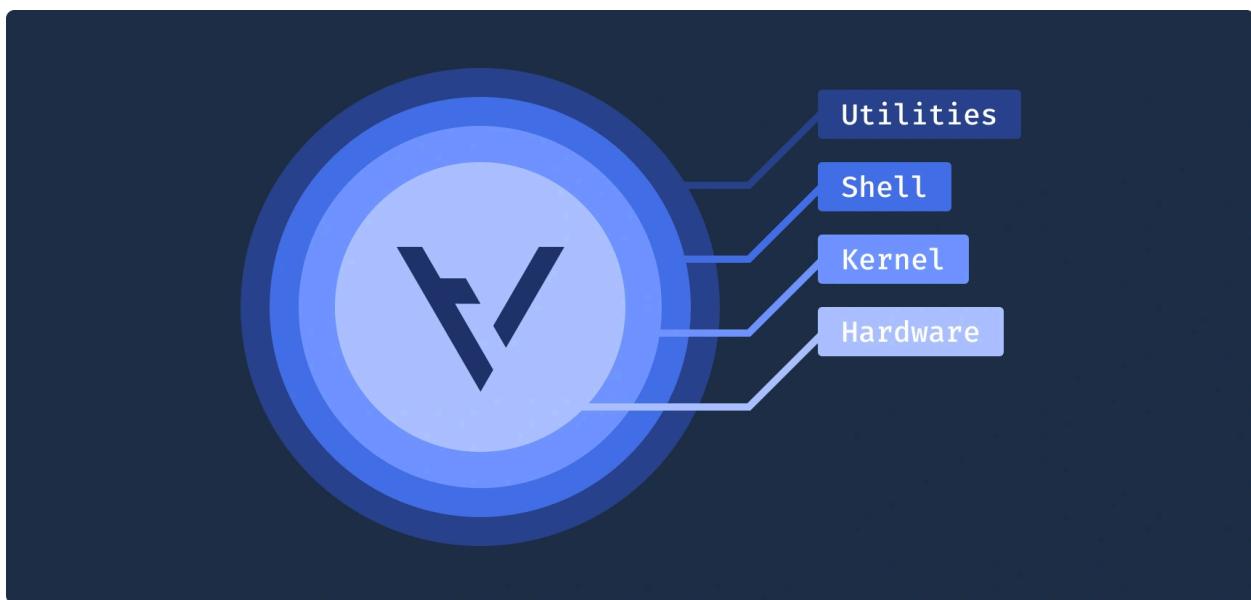
Linux, Windows'taki C:\, D:\ gibi sürücü harfleri yerine, / (root) dizininden başlayan tek bir ağaç yapısı kullanır. Bu yapı **Filesystem Hierarchy Standard (FHS)** ile standartlaştırılmıştır.

Dizin	Açıklama	İçerik Örneği
/ (Root)	Dosya sisteminin başlangıç noktasıdır.	Tüm sistem.
/bin	Temel kullanıcı komutlarını barındırır. (Binary)	ls, cp, cat, bash
/boot	Sistemin açılması için gereken dosyaları içerir.	vmlinuz (Kernel), initrd, GRUB
/dev	Donanım aygit dosyalarını içerir. (Devices)	/dev/sda (Disk), /dev/tty (Terminal)
/etc	Sistemin yapılandırma dosyalarının bulunduğu merkezdir.	/etc/passwd, /etc/ssh/sshd_config
/home	Kullanıcıların kişisel dosyalarının saklandığı yerdır.	/home/mehmet, /home/ali
/lib	Programların ihtiyaç duyduğu kütüphane dosyaları.	libc.so, Kernel modülleri
/mnt & /media	Geçici bağlanan diskler ve USB bellekler.	/media/usb-disk
/opt	Elle kurulan 3. parti uygulamalar.	Google Chrome, TeamViewer vb.
/proc	Sanal dosya sistemi. Sistem bilgisini tutar.	/proc/cpuinfo, /proc/meminfo

Dizin	Açıklama	İçerik Örneği
/root	Sistem yöneticisinin (root) ev dizinidir.	Root'un özel dosyaları.
/sbin	Sadece yöneticinin kullandığı kritik komutlar.	iptables, fdisk, reboot
/tmp	Geçici dosyalar içindir.	Uygulama önbellekleri.
/usr	İkincil hiyerarşi. Kullanıcı araçları buradadır.	/usr/bin/python, /usr/share
/var	Boyutu sürekli değişen dosyalar.	Loglar (/var/log), Web sitesi (/var/www)

Linux Mimarisi

Linux işletim sistemi, bilgisayarın kaynaklarını yönetmek ve kullanıcı etkileşimini kolaylaştırmak için katmanlı bir yapıdan oluşur.



- 1. Donanım Katmanı (Hardware):** CPU, RAM, Disk, Ağ Kartı gibi fiziksel bileşenlerdir.
- 2. Çekirdek Katmanı (Kernel):** İşletim sisteminin beynidir.
 - Kernel Space (Çekirdek Alanı):** Sadece çekirdeğin erişebildiği, donanımla doğrudan konuşan güvenli bellek alanıdır. Sürücüler burada çalışır.

- **User Space (Kullanıcı Alanı):** Kullanıcı uygulamalarının çalıştığı kısıtlı alandır. Donanıma erişmek için çekirdekten izin isterler (System Call).
3. **Sistem Kütüphaneleri (System Libraries):** Çekirdeğin karmaşık fonksiyonlarını uygulamaların anlayacağı basit komutlara çeviren çevirmenlerdir (örn: glibc).
 4. **Kabuk Katmanı (Shell):** Kullanıcıdan komut alan ve bunu çekirdeğe ileten arayüzüdür (Bash, Zsh).
 5. **Uygulama Katmanı (Applications):** Tarayıcılar, metin editörleri, veritabanı sunucuları gibi kullanıcının çalıştığı programlardır.
-

Linux Açılmış Süreci (Boot Process)

Bir Linux sisteminde güç düğmesine basıldığında sırasıyla şunlar olur:

1. **BIOS/UEFI:** Donanım testi (POST) yapar ve önyüklenebilir disk arar.
 2. **MBR/GPT:** Diskin ilk sektöründeki önyükleme kaydını okur.
 3. **Bootloader (GRUB):** İşletim sistemi seçme ekranını gösterir ve seçilen Linux çekirdeğini (Kernel) RAM'e yükler.
 4. **Kernel:** Donanımları tanır, sürücülerini yükler ve kök dosya sistemini (/) bağlar.
 5. **Init System (Systemd/SysVinit):** İlk işlem olarak (PID 1) çalışır. Arka plan servislerini, ağı ve giriş ekranını başlatır.
-

Linux Dağıtımları

Linux Dağıtımları Nedir?

Bir Linux dağıtımını, Linux çekirdeği etrafında inşa edilmiş tam bir işletim sistemidir. Çekirdeği, geniş bir yazılım uygulamaları, kütüphaneler ve isteğe bağlı bir grafik kullanıcı arayüzü (GUI) içerir. Distrolar, kullanım kolaylığı, stabilité, güvenlik veya özelleştirme gibi çeşitli ihtiyaçlara odaklanarak farklı organizasyonlar, topluluklar ve hatta bireyler tarafından geliştirilir.

Neden Farklı Dağıtımlar Var?

Linux dağıtımlarının çeşitliliği, özgürlük ve esneklik felsefesinden kaynaklanır. Farklı kullanıcıların, kişisel bilgisayarlardan sunuculara, başlangıç seviyesinden uzmanlara ve eski donanımlar için hafif sistemlerden modern makineler için özelliklerle dolu ortamlara kadar farklı ihtiyaçları vardır. Dağıtımlar, özel yazılım paketleri, kullanıcı arayüzleri ve yönetim araçları sunarak bu çeşitli gereksinimleri karşılar.

Hangi Dağıtımlı Seçmeliyim? (Karar Rehberi)

İhtiyacınıza en uygun dağıtımlı seçmek için aşağıdaki tabloyu kullanabilirsiniz:

İhtiyaç / Kullanıcı Tipi	Önerilen Dağıtım	Neden?
Yeni Başlayanlar	Linux Mint veya Ubuntu	Kurulumu çok kolaydır, Windows'a benzer, sürücü sorunu yaşamaz.
Siber Güvenlik / Hacker	Kali Linux veya Parrot OS	İçinde yüzlerce saldırı ve analiz aracı yüklü gelir.
Sunucu (Server)	Ubuntu Server , Debian , Rocky Linux	Grafik arayüzsüz gelir, çok kararlıdır, kaynak tüketimi azdır.
Geliştirici / Programcı	Fedora veya Manjaro	En güncel yazılım dillerini ve kütüphaneleri barındırır.
Eski Bilgisayarlar	Lubuntu veya Xubuntu	Çok hafiftir, eski donanımlarda bile hızlı çalışır.
Linux Uzmanları	Arch Linux	Her şeyi sıfırdan elle kurarsınız, tam kontrol sağlar.

Paket Yönetim Sistemine Göre Ana Aileler

Linux dünyası kullandıkları paket formatına (.deb, .rpm) göre büyük ailelere ayrılır. Bir ailedeki komutlar diğerinde çalışmaz.

Aile	Paket Formatı	Paket Yöneticisi	Örnek Dağıtımlar	Özellik
Debian	.deb	apt, dpkg	Ubuntu, Kali, Mint, Debian	En yaygın ailedir. Geniş topluluk desteği vardır.

Aile	Paket Formatı	Paket Yöneticisi	Örnek Dağıtımlar	Özellik
Red Hat	.rpm	dnf, yum, rpm	Fedora, CentOS, RHEL, Rocky	Kurumsal dünyada standarttır.
Arch	.pkg.tar.zst	pacman	Arch, Manjaro	Sürekli günceldir (Rolling Release).

Popüler Linux Dağıtımları

Ubuntu

Debian tabanlıdır.

- **Hedef Kitle:** Herkes (Masaüstü ve Sunucu).
- **Özellikler:** En popüler dağıtımdır. Sorun yaşadığınızda internette çözümü en kolay bulunan sistemdir.
- **Masaüstü:** Varsayılan olarak GNOME kullanır.

Fedora

Red Hat tabanlıdır.

- **Hedef Kitle:** Geliştiriciler.
- **Özellikler:** Yenilikçi teknolojileri ilk kullanan dağıtımdır. Linus Torvalds'ın da kullandığı dağıtımdır.

CentOS / Rocky Linux

Red Hat (RHEL) tabanlıdır.

- **Hedef Kitle:** Sunucular.
- **Özellikler:** Ücretli RHEL'in ücretsiz kopyasıdır. Çok kararlıdır, yıllarca kapatmadan çalışabilir.

Debian

- **Hedef Kitle:** Kararlılık isteyenler.

- **Özellikler:** "Bozulması çok zor" bir sistemdir. Paketler çok test edildiği için biraz eski sürümler olabilir ama çok güvenilirdir.

Kali Linux

Debian tabanlıdır.

- **Hedef Kitle:** Sızma Testi Uzmanları (Pentester).
- **Özellikler:** Günlük kullanım için uygun değildir. İçinde Wireshark, Metasploit, Nmap gibi araçlar hazır gelir. Root yetkisiyle çalışmaya meyillidir.

Arch Linux

- **Hedef Kitle:** İleri düzey kullanıcılar.
- **Özellikler:** Kurulum sihirbazı yoktur, komutlarla kurulur. "Rolling Release" modelini kullanır, yani versiyonu yoktur, hep en son sürümüdür.

Linux Kabuğu (Shell)

Kabuk Nedir ve Neden İhtiyacımız Var?

Bir bilgisayarın donanımı (işlemci, ram, disk) tek başına bir işe yaramaz. Onları yönetecek bir çekirdeğe (Kernel) ihtiyaç duyuyoruz. Ancak çekirdek de sadece 0 ve 1'lerden anlayan karmaşık bir yapıdır. İşte burada **Kabuk (Shell)** devreye girer.

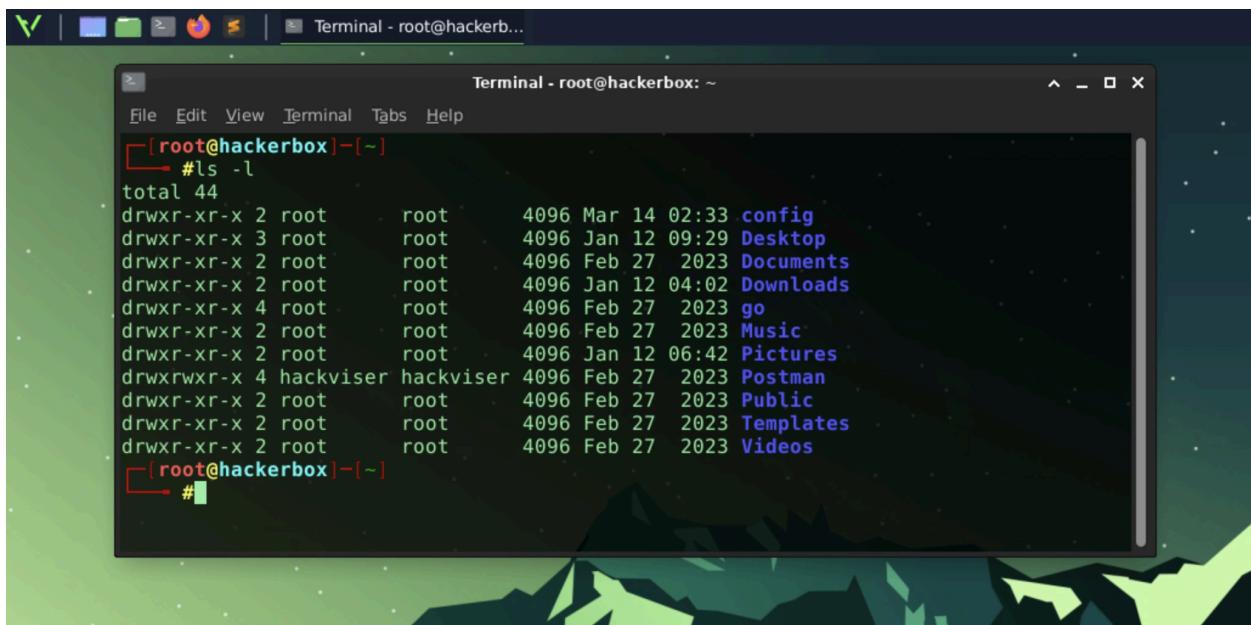
Kabuğun Görevi:

Sizin yazdığınız "insan diline yakın" komutları (örn: ls - liste), çekirdeğin anlayacağı "makine diline" çevirir. Çekirdek işlemi yaptıktan sonra sonucu alır ve tekrar sizin anlayacağınız şekilde ekrana basar. Yani bir nevi **tercümanlık** yapar.

Linux'ta iki ana tip kabuk vardır: **Komut Satırı (CLI)** ve **Grafik Arayüzü (GUI)**.

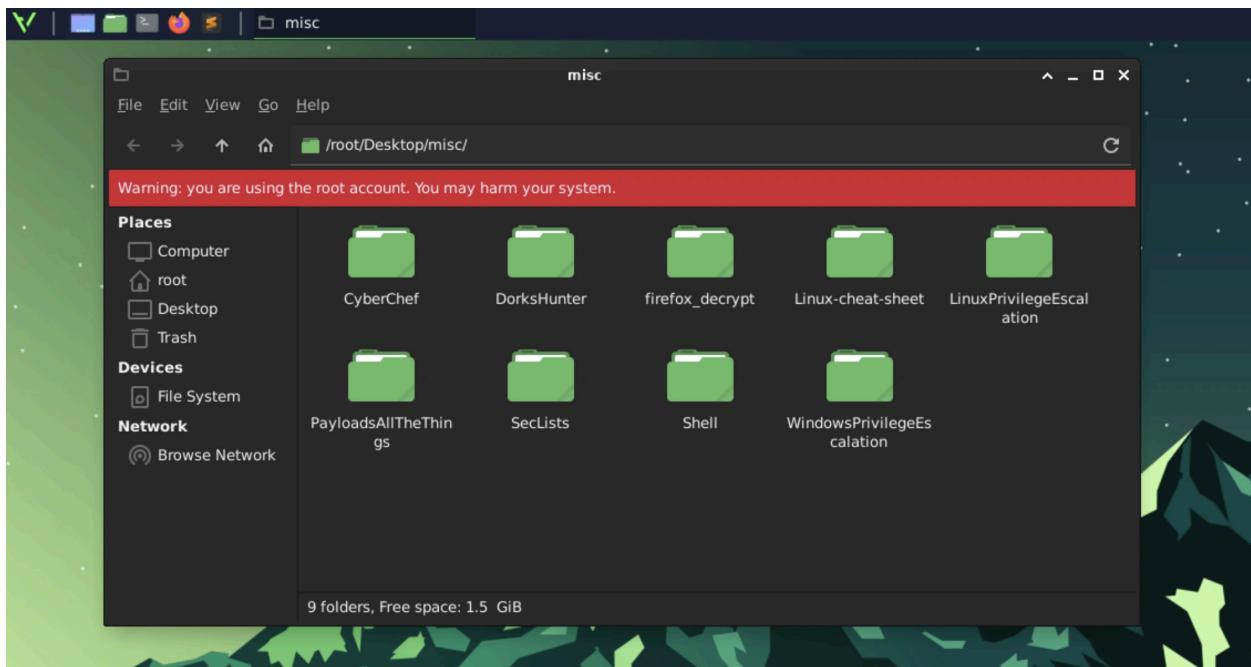
Komut Satırı (CLI)

- Komut satırı, metin tabanlı bir iletişim yöntemidir.
- Terminal adı verilen bir pencere açar ve komutları yazarak bilgisayarı yönetirsiniz.



Grafik Arayüzü (GUI)

- Windows veya macOS'ta olduğu gibi pencereler, ikonlar ve fare ile yönetilen arayüzdür.



Kabuk Çeşitleri (Bash, Zsh, Fish)

Linux'ta tek bir kabuk yoktur. İhtiyacınıza göre değiştirebilirsiniz.

1. **Bash (Bourne Again Shell)**: En yaygın ve standart kabuktur. Çoğu Linux dağıtımında varsayılan olarak gelir.
2. **Zsh (Z Shell)**: Bash ile uyumludur ancak daha gelişmiş otomatik tamamlama ve tema desteği sunar (macOS varsayılanı).
3. **Fish**: Kullanıcı dostudur, komutları yazarken renklendirir ve önerilerde bulunur.

Hangi kabuğu kullandığınızı öğrenmek için:

```
user@hackerbox:~$echo$SHELL  
/bin/bash
```

Kabuk Konfigürasyon Dosyaları (.bashrc, .zshrc)

Terminal her açıldığında kabuk, ev dizinizdeki (/home/kullanici) bazı gizli ayar dosyalarını okur. Bu dosyalara kendi kısayollarınızı ekleyebilirsiniz.

- **Bash kullanıyorsanız**: .bashrc
- **Zsh kullanıyorsanız**: .zshrc

Örnek Senaryo: Her seferinde clear yazmak yerine sadece c yazarak ekranı temizlemek istiyoruz.

1. Dosyayı nano editörüyle açın: nano ~/.bashrc
2. En alta şu satırı ekleyin: alias c='clear'
3. Kaydedip çıkm (CTRL+O, Enter, CTRL+X).
4. Ayarları yükleyin: source ~/.bashrc

Şimdi terminale c yazalım:

```
user@hackerbox:~$ c
```

Komut çalışacak ve ekran temizlenecektir. Bu yöntemle uzun ve karmaşık komutları kısaltabilirsiniz.

Çevre Değişkenleri (Environment Variables)

İşletim sisteminin ve programların çalışma şeklini etkileyen dinamik değerlerdir.

En Önemli Değişkenler ve Örnek Çıktıları:

Değişken	Açıklama
\$HOME	Kullanıcının ev dizini.
\$USER	O anki kullanıcı adı.
\$PWD	Bulunulan dizin (Present Working Directory).
\$SHELL	Kullanılan kabuk programı.
\$PATH	Komutların arandığı dizinler listesi.

Pratik Örnek: Değişkenleri Görüntüleme

```
user@hackerbox:~$ echo$HOME  
/home/user
```

```
user@hackerbox:~$ echo$USER  
user
```

```
user@hackerbox:~$ echo$PWD  
/home/user/Desktop
```

```
user@hackerbox:~$ printenv PATH  
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
```

echo komutu değişkenin değerini ekrana basar. \$PATH çıktısında, aralarında iki nokta (:) olan dizin yolları görürsünüz. Siz bir komut yazdığınızda Linux bu dizinlere sırasıyla bakar.

Betiklerin Başlangıcı (Shebang #!)

Bir metin dosyasının kabuk betiği (script) olduğunu belirtmek için ilk satırına **Shebang** (#!) eklenir.

```
#!/bin/bash  
echo"Merhaba Dünya"
```

Bu satır sisteme "Bu dosyayı /bin/bash programını kullanarak çalıştır" der.

Dizinlerde Gezinti

Dizin ve Yol (Path) Nedir?

Linux'ta her şey bir dosyadır ve bu dosyalar **Dizin (Directory)** dediğimiz dizinlerin içinde düzenlenir.

- **Kök Dizin (Root /):** Tüm sistemin başlangıç noktasıdır. Windows'taki C:\ gibidir ama Linux'ta tek bir kök vardır.
- **Yol (Path):** Bir dosyanın adresi demektir.
 - **Mutlak Yol (Absolute Path):** Adres tarifine en baştan (/) başlamaktır. Örn: /home/mehmet/resimler/tatil.jpg
 - **Göreceli Yol (Relative Path):** Adres tarifine "şu an bulduğum yerden" başlamaktır. Örn: resimler/tatil.jpg

Komutlar

- `pwd` : Şu an dosya sisteminin neresinde olduğunuzu gösterir.
- `realpath` : Bir dosyanın tam (mutlak) yolunu gösterir.
- `basename` : Yolun sadece dosya ismini verir.
- `dirname` : Yolun sadece dizin kısmını verir.

Komut	Açıklama	Örnek Çıktı (Etkisi)
<code>cd [dizin]</code>	Belirtilen dizine girer.	Kullanıcıyı belirtilen dizine taşır.
<code>cd /tam/yol</code>	Mutlak yol ile gider.	Kullanıcıyı tam yoldaki dizine taşır.
<code>cd ..</code>	Bir üst dizine çıkar.	/home/user/Downloads → /home/user
<code>cd ~</code>	Ev dizinine döner.	/var/www → /home/user
<code>cd -</code>	Bir önceki konuma döner.	/etc → /var/www

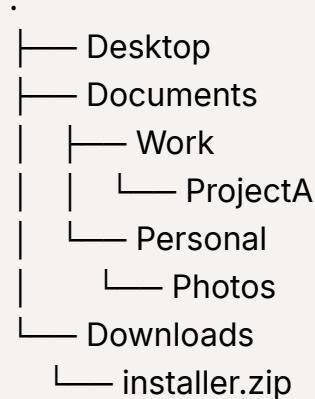
- `ls` : Bir dizinin içindeleri gösterir. Tek başına kullanıldığında az bilgi verir, bu yüzden parametrelerle (bayraklarla) kullanılır.

Sık Kullanılan Parametreler Tablosu:

Parametre	Açıklama	Örnek Çıktı
<code>-l</code>	Uzun Liste (Long): İzinleri, sahibi, boyutu ve tarihi detaylı gösterir.	drwxr-xr-x 2 user user 4096 ...
<code>-a</code>	Hepsi (All): Gizli dosyaları (ismi . ile başlayan) da gösterir.	.bashrc .profile .
<code>-h</code>	Okunabilir (Human): Dosya boyutunu Byte yerine KB, MB olarak gösterir.	4.0K, 23M
<code>-t</code>	Zaman (Time): Dosyaları değiştirilme tarihine göre sıralar (en yeni en üstte).	(Tarihe göre sıralı liste)
<code>-r</code>	Ters (Reverse): Sıralamayı tersine çevirir.	(Z-A sıralı liste)
<code>-R</code>	Özyineli (Recursive): Alt dizinlerin içini de listeler.	(Tüm dizin ağaçları)
<code>-s</code>	Boyut (Size): Dosyaları boyutuna göre sıralar (Büyükten küçüğe).	(Boyuta göre sıralı liste)

- `tree` : Dizin yapısını bir ağaç dalı gibi görselleştirir.

```
user@hackerbox:~$tree -L2
```



Dizinlerin iç içe yapısını tek bakışta anlamamızı sağlar. `-L 2` parametresi sadece 2 seviye derine inmesini söyler.

Dosya ve Dizin İşlemleri

1. Dosya Oluşturma (touch)

- `touch` komutu dosya varsa tarihini günceller, yoksa boş bir dosya oluşturur.

```
user@hackerbox:~$touch notlar.txt  
user@hackerbox:~$ls -l notlar.txt  
-rw-r--r-- 1 user user 0 Aug 01 12:00 notlar.txt
```

ls -l çıktısında görüldüğü gibi boyutu 0 olan bir dosya oluşturuldu.

Toplu Oluşturma:

```
user@hackerbox:~$touch dosya{1..3}.txt  
user@hackerbox:~$ls  
dosya1.txt dosya2.txt dosya3.txt
```

Süslü parantez {} kullanarak tek seferde 3 dosya oluşturduk.

2. Dizin Oluşturma (mkdir)

```
user@hackerbox:~$mkdir projeler
```

3. Kopyalama (cp)

- Dosya veya dizinlerin kopyasını oluşturur.

Komut	Açıklama
<code>cp dosya.txt yedek.txt</code>	Dosyayı kopyalar.
<code>cp -r dizin/ yedek_dizin/</code>	Dizini ve İçindekileri kopyalar (Recursive).
<code>cp -i dosya.txt hedef/</code>	Hedefte aynı dosya varsa üzerine yazmadan önce sorar (Interactive).
<code>cp --backup dosya.txt hedef/</code>	Hedefte aynı dosya varsa, eskisinin adını değiştirek yedekler (örn: dosya.txt~).

4. Taşıma ve Yeniden Adlandırma (mv)

Linux'ta "Yeniden Adlandırma" komutu yoktur, bu işi `mv` (Move) yapar.

- **İsim Değiştirme:** mv eski.txt yeni.txt
- **Taşıma:** mv dosya.txt /tmp/

Örnek Senaryo: Dosya Taşıma

```
user@hackerbox:~$mv notlar.txt Belgeler/
user@hackerbox:~$ls Belgeler/
notlar.txt
```

Dosya artık bulunduğu yerden silindi ve Belgeler dizinine taşındı.

5. Silme (rm)

DİKKAT: Linux terminalinde "Geri Dönüşüm Kutusu" yoktur. Silinen dosya gider.

Komut	Açıklama
rm dosya.txt	Dosyayı siler.
rm -i dosya.txt	Silerken "Emin misiniz?" diye sorar (Güvenli mod).
rm -r dizin/	Dizini ve içindekileri siler.
rm -rf dizin/	Dizini zorla ve sormadan siler (Çok tehlikelidir!).
rmdir dizin/	Sadece İçi boş dizini siler (En güvenli yöntem).

Güvenli Silme

```
user@hackerbox:~$ rm -i önemli_belge.txt
rm:remove regular file 'önemli_belge.txt'? n
```

*Güvenli silmek için **-i** şarttır.*

```
user@hackerbox:~$rm -r projeler/
```

*İçi dolu dizinleri silmek için **-r** şarttır.*

6. Dosya Bilgisi ve Inode

- **file:** Dosyanın türünü (resim, metin, binary) belirler.

```
user@hackerbox:~$file /bin/bash  
/bin/bash: ELF64-bit LSB shared object, x86-64  
user@hackerbox:~$file notlar.txt  
notlar.txt: ASCII text
```

Uzantısı ne olursa olsun, dosyanın gerçek içeriğini söyler.

- **stat:** Dosya hakkında çok detaylı bilgi verir (Inode, zaman damgaları).

```
user@hackerbox:~$ stat notlar.txt  
File: notlar.txt  
Size:0      Blocks:0          IO Block:4096   regular empty file  
Inode:123456    Links:1  
Access:2023-10-20 12:00:00.000000000 +0300  
Modify:2023-10-20 12:00:00.000000000 +0300  
Change:2023-10-20 12:00:00.000000000 +0300
```

7. Dosya Bağlantıları (Links)

Inodeları görmek için `ls -i` kullanılır:

```
user@hackerbox:~$ ls -i notlar.txt  
123456notlar.txt
```

8. İleri Düzey Kopyalama (dd)

- `dd` komutu düşük seviyeli kopyalama yapar. Disk imajı almak veya yazdırma için kullanılır.

Örnek: ISO Dosyasını USB'ye Yazdırma

```
sudo dd if=linux.iso of=/dev/sdb bs=4M status=progress
```

- if: Kaynak (Input File - ISO dosyası)
- of: Hedef (Output File - USB bellek)

- `bs`: Blok boyutu (Hız için).
- `status=progress`: İlerleme çubuğunu gösterir.

Linux'ta Dosya ve Dizinleri Bulma Yöntemleri

find Komutu ile Arama

`find` komutu, belirli kriterlere göre dosya ve dizinleri aramak için kullanılır. Derinlemesine arama yapabilme yeteneği ile, büyük ve karmaşık dosya sistemlerinde etkili sonuçlar üretir.

Genel Kullanım:

```
find [NEREDE] [KRİTER] [EYLEM]
```

Temel Aramalar

Kriter	Örnek Komut	Açıklama
İsim	<code>find / -name "notlar.txt"</code>	Tam ismi "notlar.txt" olanı bulur.
İsim (Harf Duyarsız)	<code>find . -iname "ReSiM.Jpg"</code>	Büyük/küçük harfe takılmadan bulur.
Tür	<code>find /home -type d</code>	Sadece dizinleri (d) bulur. (f: dosya).
Boyut	<code>find / -size +100M</code>	100 Megabyte'tan büyük dosyaları bulur.
Kullanıcı	<code>find /var -user www-data</code>	Sahibi www-data olan dosyaları bulur.

Örnek Senaryo: İsimle Arama

```
user@hackerbox:~$find . -name "*.txt"
./Belgeler/notlar.txt
./Masaüstü/yapılacaklar.txt
./İndirilenler/liste.txt
./cache/gecici.txt
```

Bulduğumuz dizin ve altındaki tüm dizinlerde sonu.txt ile biten dosyaları listeledik.

Örnek Senaryo: Türé Göre Arama

Sadece ssh ismindeki dizinleri bulmak için:

```
user@hackerbox:~$find /etc -type d -name"ssh"  
/etc/ssh
```

Eğer -type d demeseydik, ssh ismindeki dosyaları veya linkleri de bulabilirdi.

Örnek Senaryo: Boyuta Göre Arama

1GB'dan büyük dosyaları bulmak için:

```
user@hackerbox:~$find / -size +1G  
/var/log/syslog.1  
/home/user/film.mkv  
/var/lib/mysql/ibdata1
```

Disk dolduğunda yer açmak için en çok kullanılan yöntemdir.

Bulunan Dosyaya Komut Uygulama (exec ve xargs)

En güçlü özellikle. Bulduğu her dosya için bir komut çalıştırır.

Örnek:

.tmp uzantılı tüm dosyaları bul ve sil.

```
user@hackerbox:~$find . -name"*.tmp" -exec rm {} \;
```

- {}: Bulunan dosya isminin yerini tutar.
- \;: Komutun bittiğini belirtir.

xargs Kullanımı:

Çok fazla dosya bulunduğuanda xargs daha performanslıdır.

```
user@hackerbox:~$find . -name"*.log" | xargs rm
```

locate Komutu ile Arama

locate komutu, sistemdeki dosyaları bulmak için kullanılır. locate, updatedb adlı bir veritabanını kullanır. Bu veritabanı, sisteminizdeki dosyaların bir dizinini içerir ve locate komutu bu veritabanında hızlı bir şekilde arama yapar.

- **Temel Kullanım:**

```
user@hackerbox:~$ locate notlar.txt  
/home/user/Belgeler/notlar.txt  
/home/user/Yedek/notlar.txt
```

locate komutunun hızlı sonuçlar vermesi, dosya sistemindeki değişikliklerin updatedb tarafından periyodik olarak güncellenmesine bağlıdır. Bu yüzden, çok yeni dosyaları bulamayabilir. Veritabanını manuel güncellemek için sudo updatedb komutu çalıştırılabilir.

which Komutu ile Komut Dosyalarını Bulma

which komutu, belirli bir komutun yolu hakkında bilgi edinmek için kullanılır. Bu, özellikle birden fazla sürümü yüklü olan programların hangisinin kullanıldığını anlamak için faydalıdır.

- **Temel Kullanım:**

```
user@hackerbox:~$ which python  
/usr/bin/python
```

Bu komut, python komutunun hangi yolda olduğunu gösterir. Bu, genellikle sistemdeki varsayılan python sürümünün konumunu öğrenmek için kullanılır.

grep ile Dosya İÇİNDE Arama

Dosya ismini değil, dosyanın **İçindeki yazıyı** arar.

Örnek Senaryo:

server.log dosyasında "Error" geçen satırları bulalım.

```
user@hackerbox:~$ grep "Error" server.log  
[2023-10-01 10:00] Error: Connection timeout
```

[2023-10-0110:05] Error: Database unreachable

[2023-10-0110:10] Error: Service stopped

Regex (Düzenli İfade) Tablosu ve Örnekler:

Sembol	Anlamı	Örnek Komut	Eşleşen
^	Satır başı	grep "^Hata" log.txt	"Hata oluştu" (Evet), "Bir Hata" (Hayır)
\$	Satır sonu	grep "tamam\$" log.txt	"İşlem tamam" (Evet), "tamamlandı" (Hayır)
.	Herhangi bir karakter	grep "k.t" kelimeler.txt	kat, küt, kıt
*	Öncekinin tekrarı	grep "ab*" kod.txt	a, ab, abb, abbb

Pratik Örnek: Regex ile Arama

Satır başı "User" ile başlayan satırları bul:

```
user@hackerbox:~$ grep "^User" config.txt
User:admin
User: guest
```

Örnek: Tüm Dizinde Arama

/var/log içindeki tüm dosyalarda "Error" kelimesini ara (büyük/küçük harf duyarsız).

```
user@hackerbox:~$ grep -r -i"error" /var/log/
/var/log/syslog:Oct112:00:00 Error: Disk full
/var/log/auth.log:Oct112:05:00 Failed password
/var/log/kern.log:Oct112:10:00 I/O Error
```

 **Ömer Faruk BAYSAL**