

# Bölüm 11: Linux

## İşletim Sistemleri

# Unix Tabanlı İşletim Sistemleri Evrimi

- Unics: 1960'ların başında Bell Laboratuarlarında çok kullanıcılı ve zaman paylaşımı sistem olarak geliştirildi.
- Unix: 1960'ların sonlarında çok görevli ve çok kullanıcılı işletim sistemi olarak
- PDP-11: Digital Equipment Corporation tarafından 1970 yılında geliştirilen mini bilgisayar. Unix başlangıçta bu platform için geliştirildi.
- Berkeley Yazılım Dağıtım (BSD): 1970'lerin sonunda Berkeley'deki California Üniversitesi'nde geliştirilen bir Unix türevi.
- Minix: Tanenbaum tarafından 1987'de Unix benzeri eğitimsel bir işletim sistemi olarak oluşturuldu.
- Linux: Linus Torvalds tarafından 1991 yılında ücretsiz ve açık kaynaklı. Kişisel bilgisayarlar için tasarlanmıştır, ancak sunucular, mobil cihazlar ve gömülü sistemler dahil çeşitli platformlar için benimsenmiştir.

# Unix Tarihçe

- UNIX, 1969'da Bell Laboratuvarlarında Ken Thompson ve Dennis Ritchie tarafından geliştirildi.
- Mini bilgisayarlar için çok kullanıcılı, çok görevli bir işletim sistemi olarak tasarlanmıştır.
- Orijinal UNIX işletim sistemi, Assembly dilinde yazılmıştır.
- 1972'de UNIX, taşınabilirlik için C'de yeniden yazıldı.
- 1984 yılında AT&T, ticari UNIX sistemleri için standart haline gelen UNIX System V'i piyasaya sürdü.
- 1980'lerin sonlarında ve 1990'lارın başlarında, açık kaynak hareketi, Linux ve BSD gibi çeşitli UNIX tabanlı işletim sistemlerinin geliştirilmesiyle sonuçlandı.

# Linux Tarihte

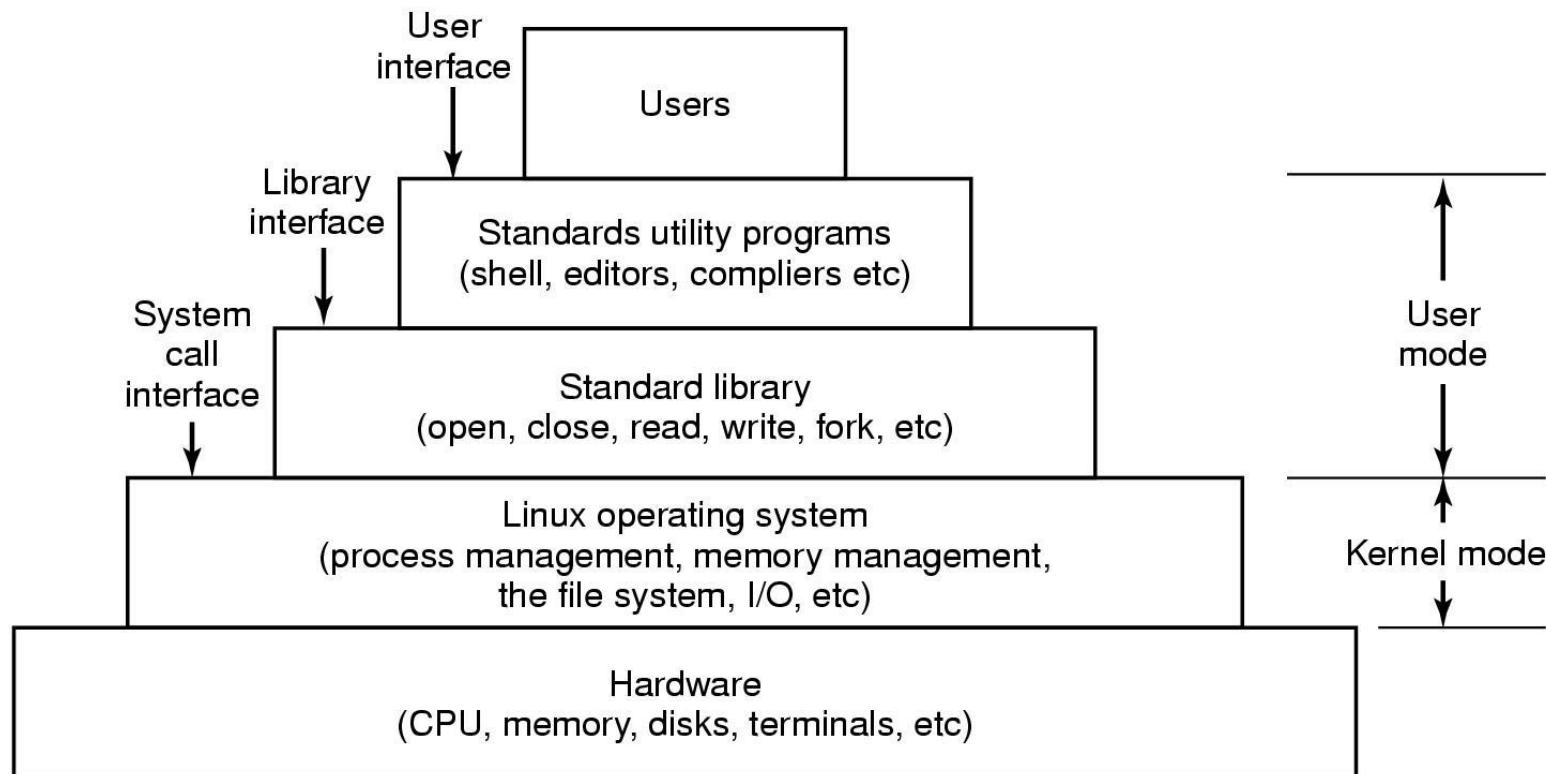
- 1991: Linus Torvalds tarafından bir hobi projesi olarak oluşturuldu.
- 1992: Linux sürüm 0.12 yayınlandı
- 1994: Linux'un ilk ticari sürümü (SLS Linux)
- 1998: Red Hat tarafından ilk Linux dağıtımı
- 2000: Linux halka açık bir şirket haline geldi (Red Hat)
- 2002: İlk yerleşik Linux cihazı piyasaya sürüldü (Sharp Zaurus PDA)
- 2005: Linux çalıştırın ilk Android cihazının lansmanı (HTC Dream)
- 2008: Linux, sunuculardaki en popüler işletim sistemi oldu

# Hedefler

- Programcılar tarafından programcılar için tasarlandı
- Basit
- Şık (elegant)
- Tutarlı (consistent)
- Güçlü (powerful)
- Esnek (flexible)

# UNIX Katmanlar

- .
- .



# UNIX Komutlar

- ls - dosya ve dizinleri listeler
- cd - mevcut dizini değiştir
- mkdir - yeni bir dizin oluştur
- rm - dosyaları veya dizinleri kaldırın
- touch - yeni bir dosya oluştur
- cp - dosyaları veya dizinleri kopyala
- mv - dosyaları veya dizinleri taşıyın veya yeniden adlandırın
- cat - bir dosyanın içeriğini göster
- echo - metni ekranada göster
- less - bir dosyanın içeriğini her seferinde bir sayfa olarak görüntüler

# UNIX Komutlar

- head - bir dosyanın ilk birkaç satırını görüntüler
- tail - bir dosyanın son birkaç satırını görüntüler
- grep - bir dosyada metin arayın
- find - dosyaları veya dizinleri arayın
- tar - bir tar arşivi oluşturun veya çıkarın
- ps - işlem bilgilerini göster
- head - sistem bilgilerini ve kaynak kullanımını görüntüleyin
- chmod - dosya veya dizin izinlerini değiştirir
- chown - bir dosyanın veya dizinin sahibini veya grubunu değiştirir
- ssh - uzak bir makineye güvenli kabuk girişi

# UNIX Komutlar

- scp - makineler arasında güvenli dosya kopyalar
- sync - dosyaları makineler arasında senkronize eder
- ping - ağ bağlantısını test edin
- ifconfig - ağ yapılandırma bilgilerini görüntüler
- route - yönlendirme bilgilerini görüntüleyin veya ayarlayın
- traceroute - bir ana bilgisayara giden ağ yolunu görüntüler
- netstat - ağ durumu bilgilerini görüntüler
- iptables - güvenlik duvarı kurallarını yapılandır
- curl - bir URL'den veri aktarın
- wget - web'den dosya indirme.

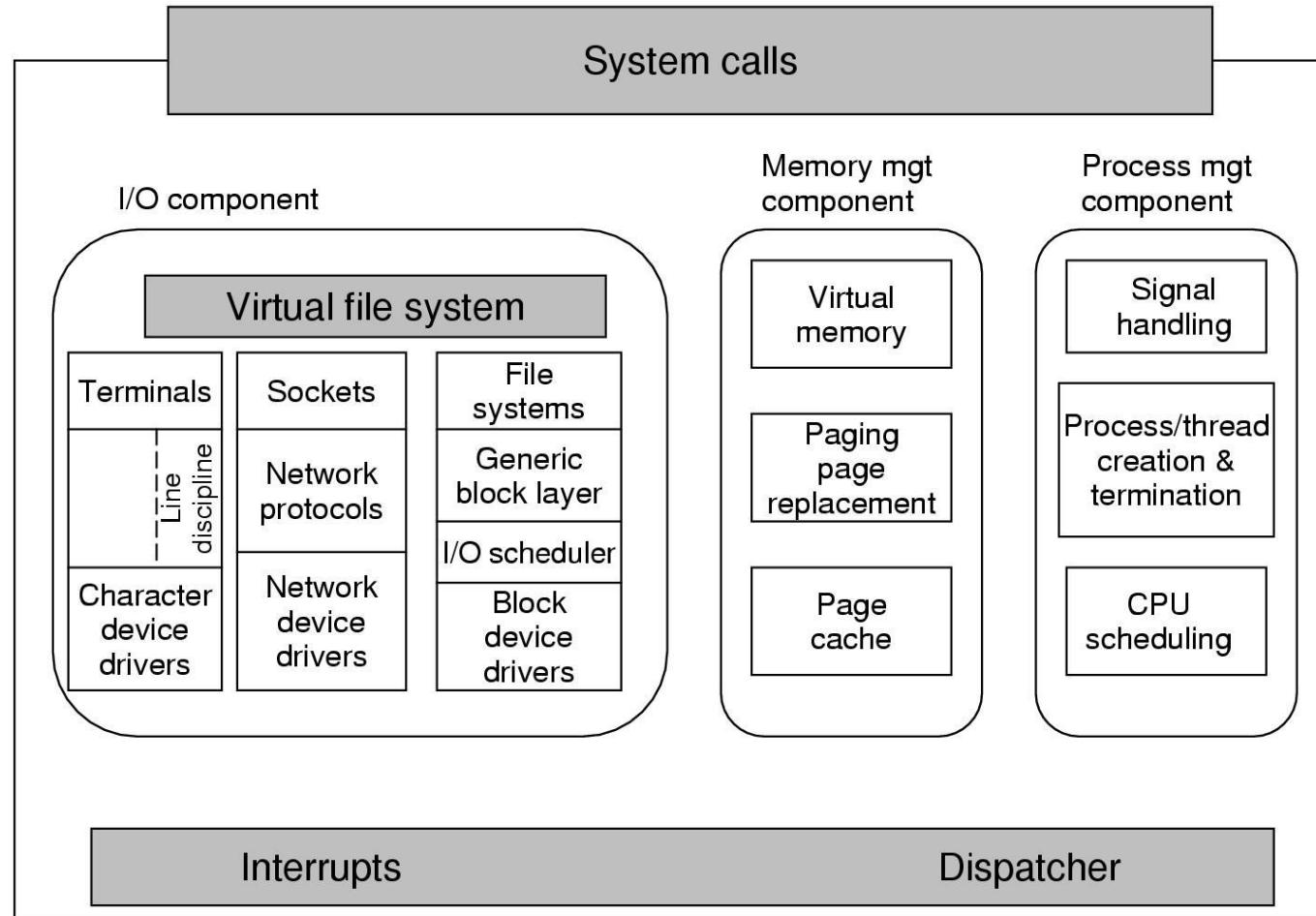
# UNIX Çekirdek

- .
- .

System calls					Interrupts and traps					
Terminal handling		Sockets	File naming	Map-ping	Page faults	Signal handling	Process creation and termination			
Raw tty	Cooked tty	Network protocols	File systems	Virtual memory						
	Line disciplines	Routing	Buffer cache	Page cache		Process scheduling				
Character devices		Network device drivers	Disk device drivers			Process dispatching				
Hardware										

# UNIX Çekirdek

- .
- .



# POSIX Sinyaller

- POSIX sinyalleri, Unix benzeri işletim sistemlerinde kullanılan bir süreçler arası iletişim (IPC) yöntemidir.
- Sinyaller, belirli bir sistem durumu gibi bir olay sürecini bildirmek veya normal yürütme akışını kesmek için kullanılabilir.
- Sinyaller senkronize veya asenkron olabilir ve sistem, diğer süreçler veya sürecin kendisi tarafından üretilebilir.
- POSIX sinyalleri, bir işlemin sonlandırılması, bir dosyanın durumundaki bir değişiklik veya sıfıra bölme hatası gibi kendi ilişkili davranışları olan sınırlı bir önceden tanımlanmış sinyal değerleri kümesine sahiptir.
- İşleyici, kaynakları serbest bırakmak veya bilgileri günlüğe kaydetmek gibi gerekli tüm işlemleri gerçekleştirebilir.

# POSIX Sinyaller

- SIGABRT: Sürecin bir hatayla karşılaştığında anormal şekilde sonlandırılması.
- SIGALRM: alarm() işlevi tarafından ayarlanan bir zamanlayıcı çaldığında üretilen çalar saat sinyali.
- SIGFPE: sıfıra bölmek gibi geçersiz bir işlem gerçekleştirildiğinde oluşturulan kayan nokta istisnası.
- SIGHUP: sistem bir bağlantı kaybı algılandığında üretilen kapatma sinyali.
- SIGILL: İşlem, geçersiz bir makine talimatıyla karşılaştığında oluşturulan yasadışı talimat.
- SIGQUIT: Çıkış tuşu (CTRL-) tarafından üretilir. İşlem, bir çekirdek dökümü üretecek sona erer.
- SIGKILL: Kill komutu tarafından üretilen öldürme sinyali. temizleme fırsatı olmadan sona erer.
- SIGPIPE: İşlem, kapatılmış bir boruya veya sokete yazmaya çalıştığında üretilen kırık boru sinyali.
- SIGSEGV: İşlem geçersiz bir bellek adresine eriştiğinde oluşturulan segmentasyon hatası.
- SIGTERM: kill komutu tarafından üretilen sonlandırma sinyali. kaynaklar temizlenerek sona erer.
- SIGUSR1: Kullanıcı tanımlı sinyal 1, programların uygun gördükleri şekilde kullanmaları için.

# Süreç Yönetimi için Sistem Çağrıları

- Linux çekirdeği tarafından sağlanan ve kullanıcı düzeyinde programın çekirdekle etkileşim kurması için bir arabirim sağlayan bir işlevdir.
- Sistem çağrıları, bir programın Linux işletim sisteminden hizmet talep etmesi için birincil mekanizmadır.
- Örnekler: `open()`, `read()`, `write()`, `close()`, `fork()`, `exec()`, `wait()`.
- Linux'ta dosya ve cihaz G/Ç, süreç yönetimi ve süreçler arası iletişim (IPC) dahil olmak üzere birçok sistem çağrısı türü vardır.
- Sistem çağrıları, Linux işletim sisteminin kritik bir bileşenidir ve kullanıcı düzeyindeki programların düzgün çalışması için gereklidir.

# Süreç Yönetimi için Sistem Çağrıları

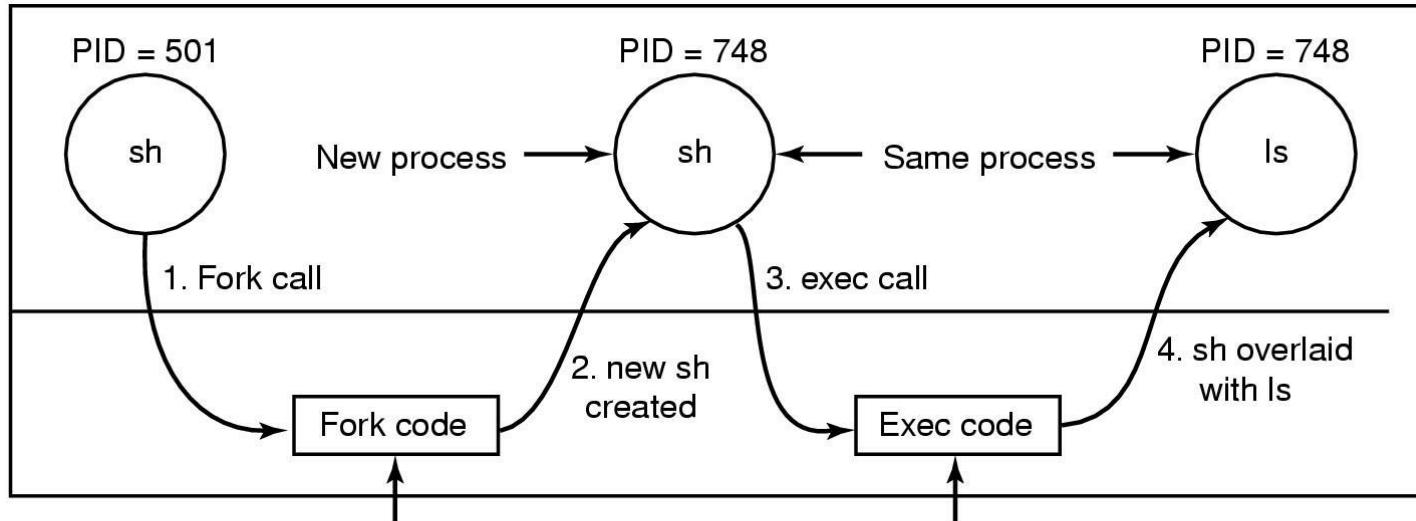
- fork(): Çağırılan işlemi çoğaltarak yeni bir süreç oluşturur.
- waitpid(): bir alt sürecin durumunu değiştirmesini bekler
- execve(): geçerli süreç görüntüsünü yeni bir süreç görüntüsüyle değiştirir
- exit (): çağrılan süreci sonlandırır ve durumu döndürür
- sigaction(): belirtilen bir sinyal için bir sinyal eylemi ayarlar
- sigreturn(): kontrolü sinyal işleyiciye döndürür
- sigprocmask(): çağrıran sürecin sinyal maskesini ayarlar
- sigpending(): çağrıran süreç için engellenen ve bekleyen sinyalleri döndürür
- sigsuspend(): çağrıran sürecin sinyal maskesini geçici olarak değiştirir
- kill (): belirtilen süreçte bir sinyal gönderir
- alarm(): belirtilen sayıda saniye için bir çalar saat ayarlar
- pause(): çağrıran sürecin bir sinyal alınana kadar uyumasına neden olur

# pthread İşlevleri

- `pthread_create`: Yeni bir iş parçacığı oluşturur.
- `pthread_exit`: Çağırılan iş parçacığını sonlandırır.
- `pthread_join`: Bir iş parçacığının sonlanmasını bekleyin.
- `pthread_mutex_init`: Bir mutex nesnesini başlatır.
- `pthread_mutex_destroy`: Bir mutex nesnesini yok eder.
- `pthread_mutex_lock`: Bir mutex nesnesinin sahipliğini alır.
- `pthread_mutex_unlock`: Mutex nesnesinin sahipliğini serbest bırakır.
- `pthread_cond_init`: Bir koşul değişkeni nesnesini başlatır.
- `pthread_cond_destroy`: Bir koşul değişkeni nesnesini yok eder.
- `pthread_cond_wait`: bir mutex'i serbest bırakır ve bir koşul değişkenini bekler.
- `pthread_cond_signal`: koşul değişkeni bekleyen iş parçacığının engellemesini kaldırır.

# ls (list) Komutu

- .
- .



Allocate child's process table entry  
Fill child's entry from parent  
Allocate child's stack and user area  
Fill child's user area from parent  
Allocate PID for child  
Set up child to share parent's text  
Copy page tables for data and stack  
Set up sharing of open files  
Copy parent's registers to child

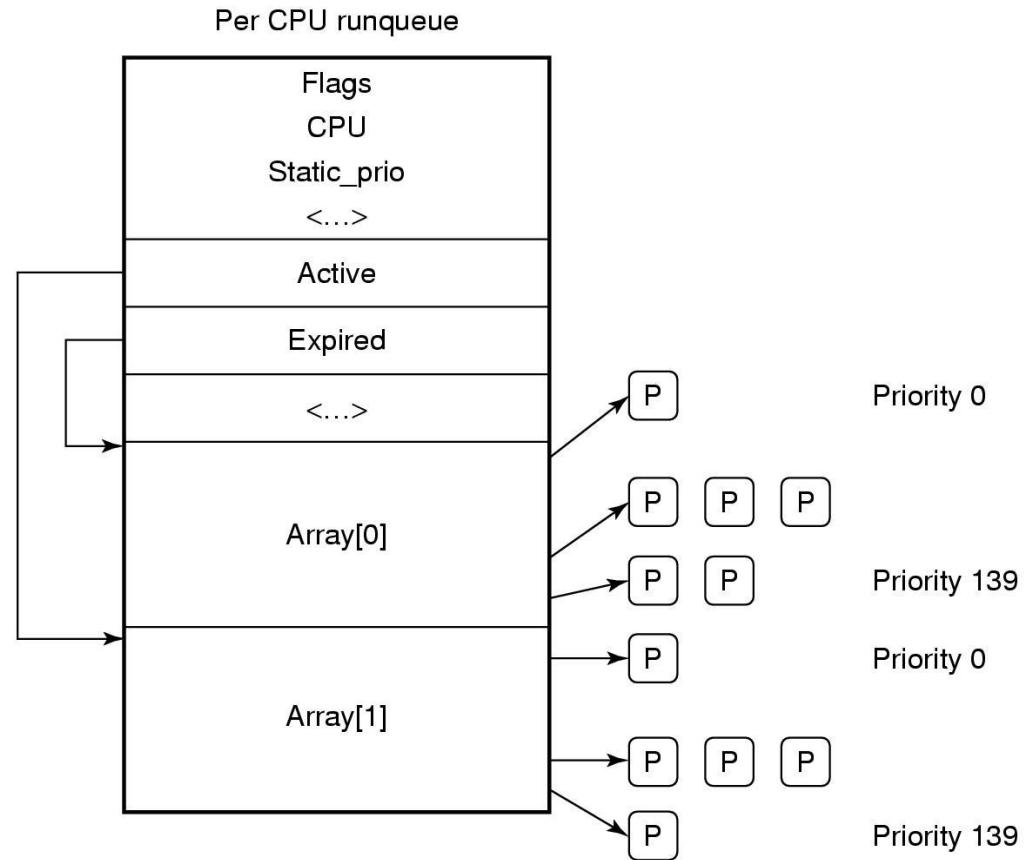
Find the executable program  
Verify the execute permission  
Read and verify the header  
Copy arguments, environ to kernel  
Free the old address space  
Allocate new address space  
Copy arguments, environ to stack  
Reset signals  
Initialize registers

# Linux Çizelgeleme

- Çizelgeleme amaçlı üç çeşit:
- Gerçek zamanlı ilk-giren ilk-çıkar (Real-time FIFO).
- Gerçek zamanlı sıralı (Real-time round robin).
- Zaman paylaşımı (Timesharing).

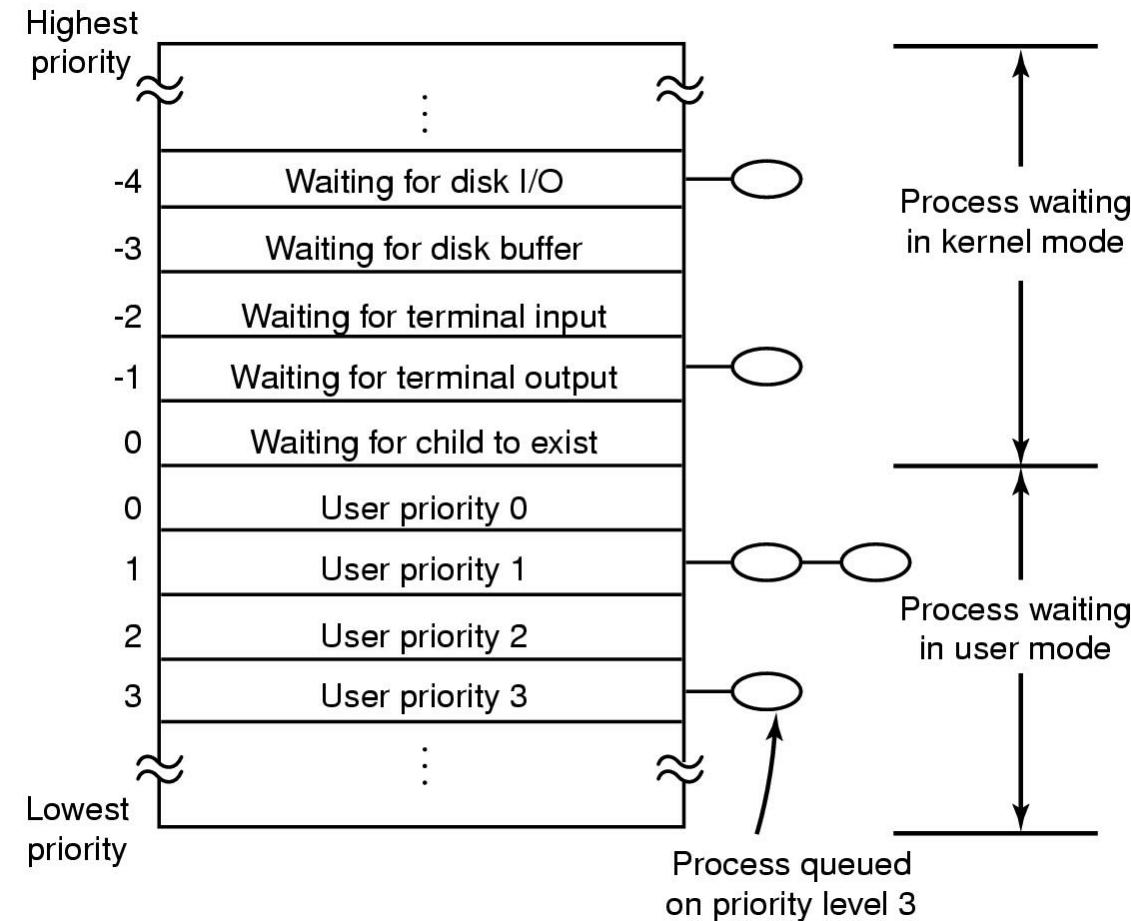
# Linux Çizelgeleme – Öncelik Dizileri

- 
- 



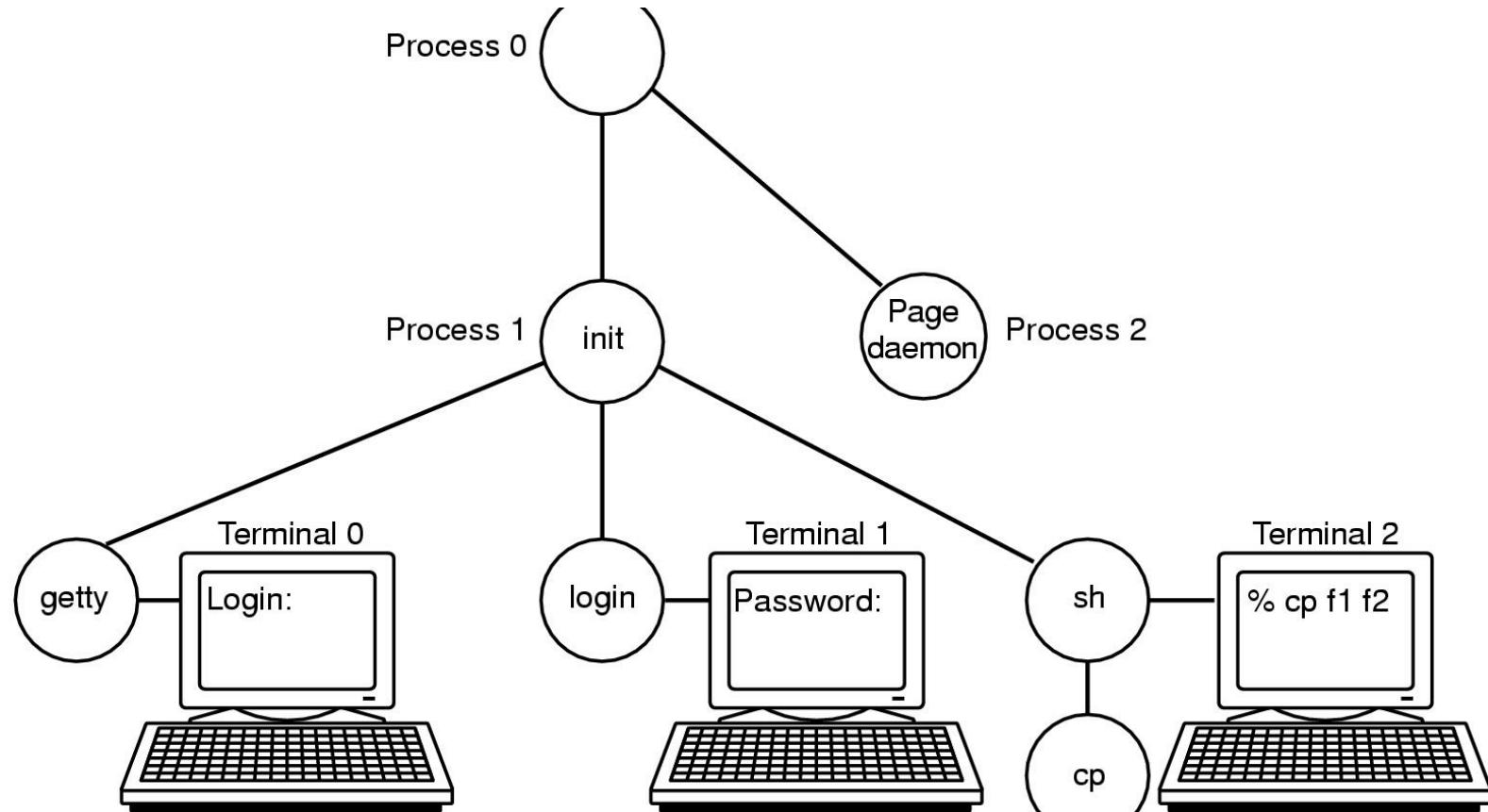
# Linux Çizelgeleyici

- .
- .



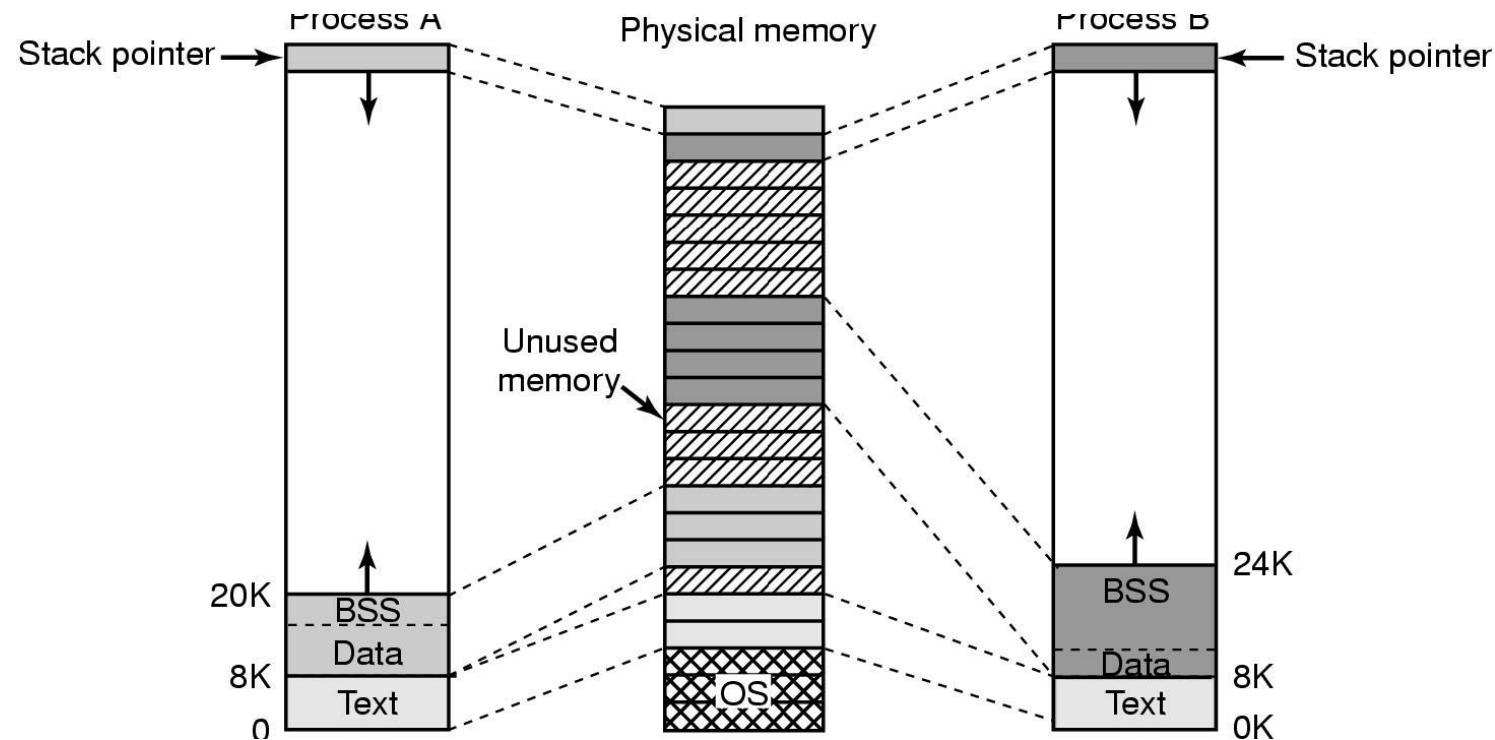
# Önyükleme (boot)

- ..

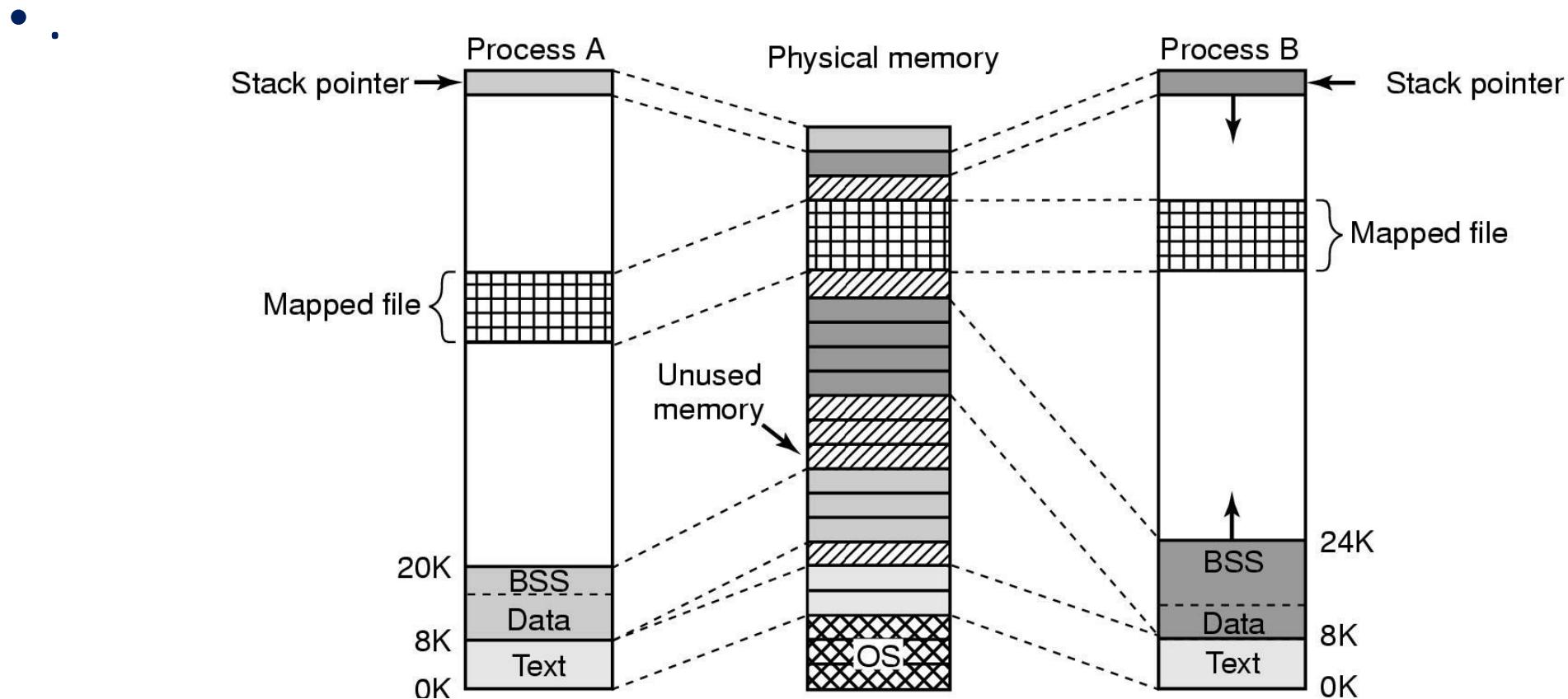


# Bellek Yönetimi

- Süreç A ve B'nin sanal adres alanı, fiziksel bellek,



# Dosya Paylaşımı

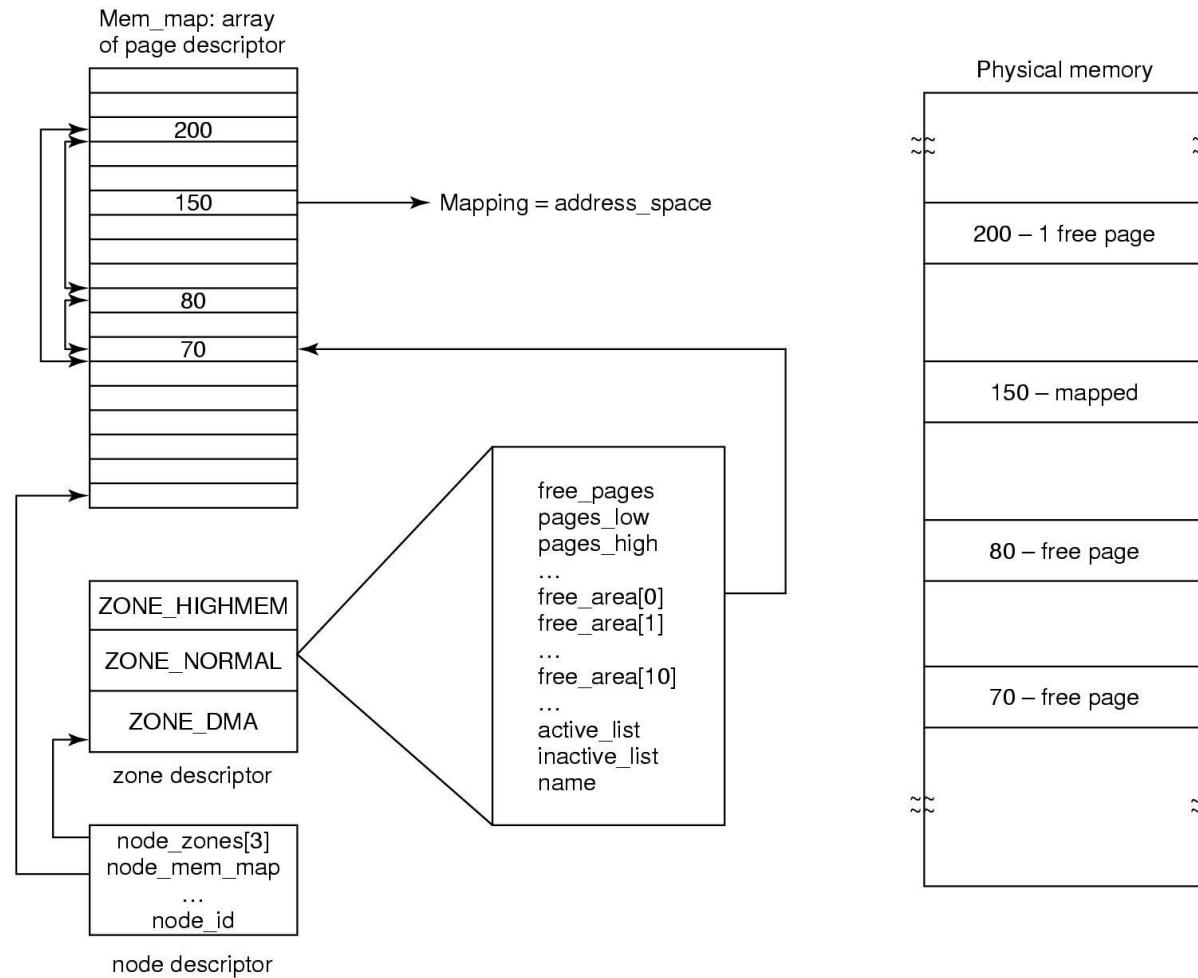


# Bellek Yönetimi Sistem Çağrıları

- brk: programın veri bölümünün sonunu belirleyen program sonu konumunu değiştirir. Bu çağrı, veri bölümünün boyutunu artırmak veya azaltmak için kullanılır.
- mmap: bir dosyayı veya aygıtı belleğe eşler. Bu çağrı, bir programın veri bölümü için bellek ayırmak veya doğrudan erişim için bir dosyayı belleğe eşlemek için kullanılır.
- unmap: daha önce mmap kullanılarak ayrılan belleği serbest bırakır. Bu çağrı, program tarafından artık ihtiyaç duyulmadığında belleği sisteme geri döndürmek için kullanılır.

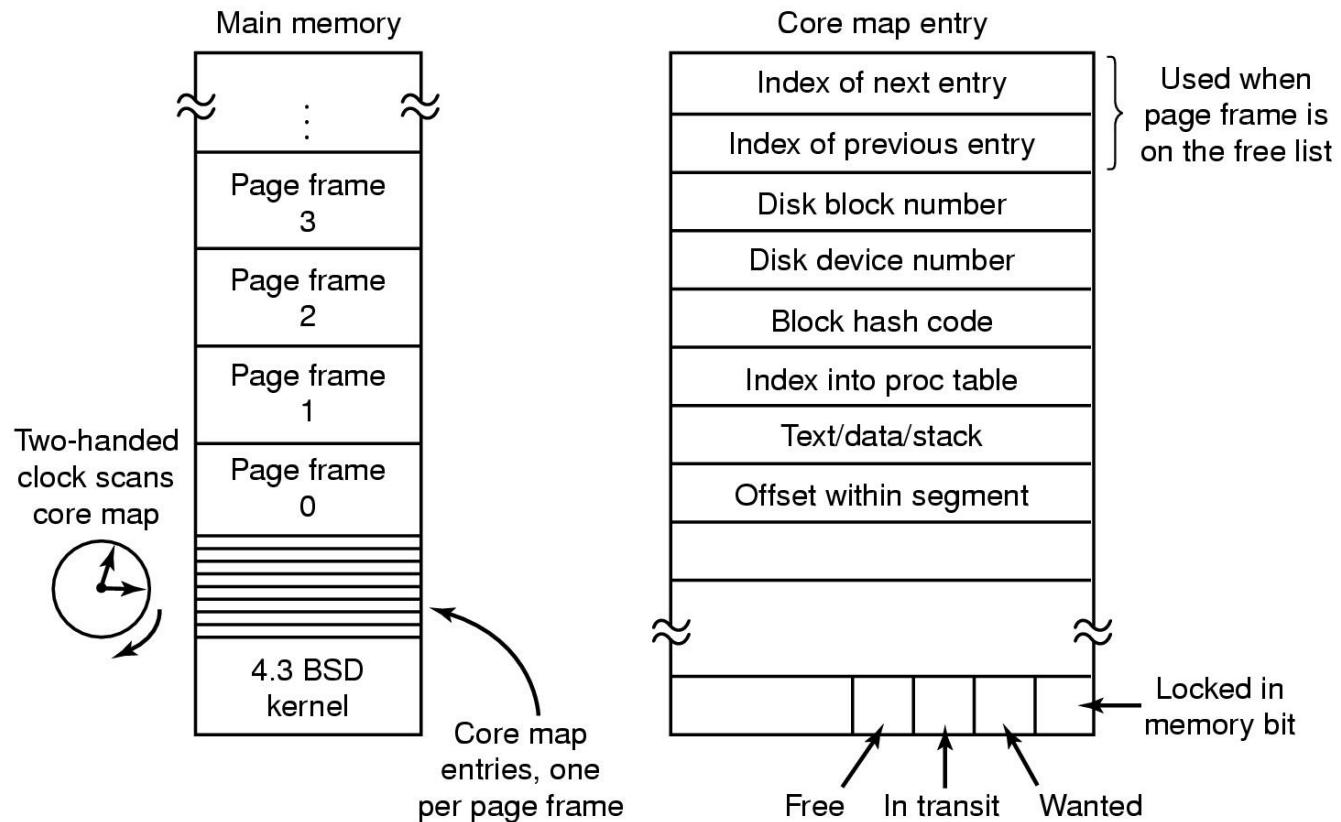
# Linux Ana Bellek Gösterimi

- 
- 



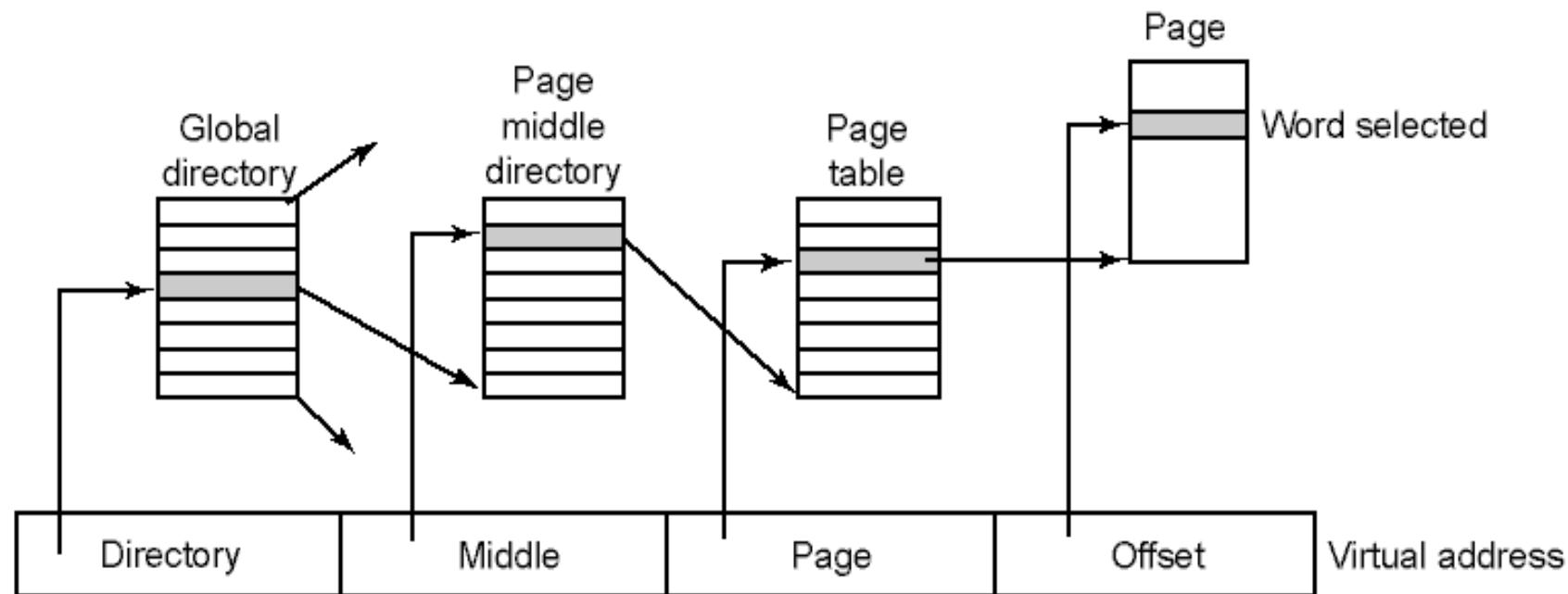
# Sayfalama

- .
- .

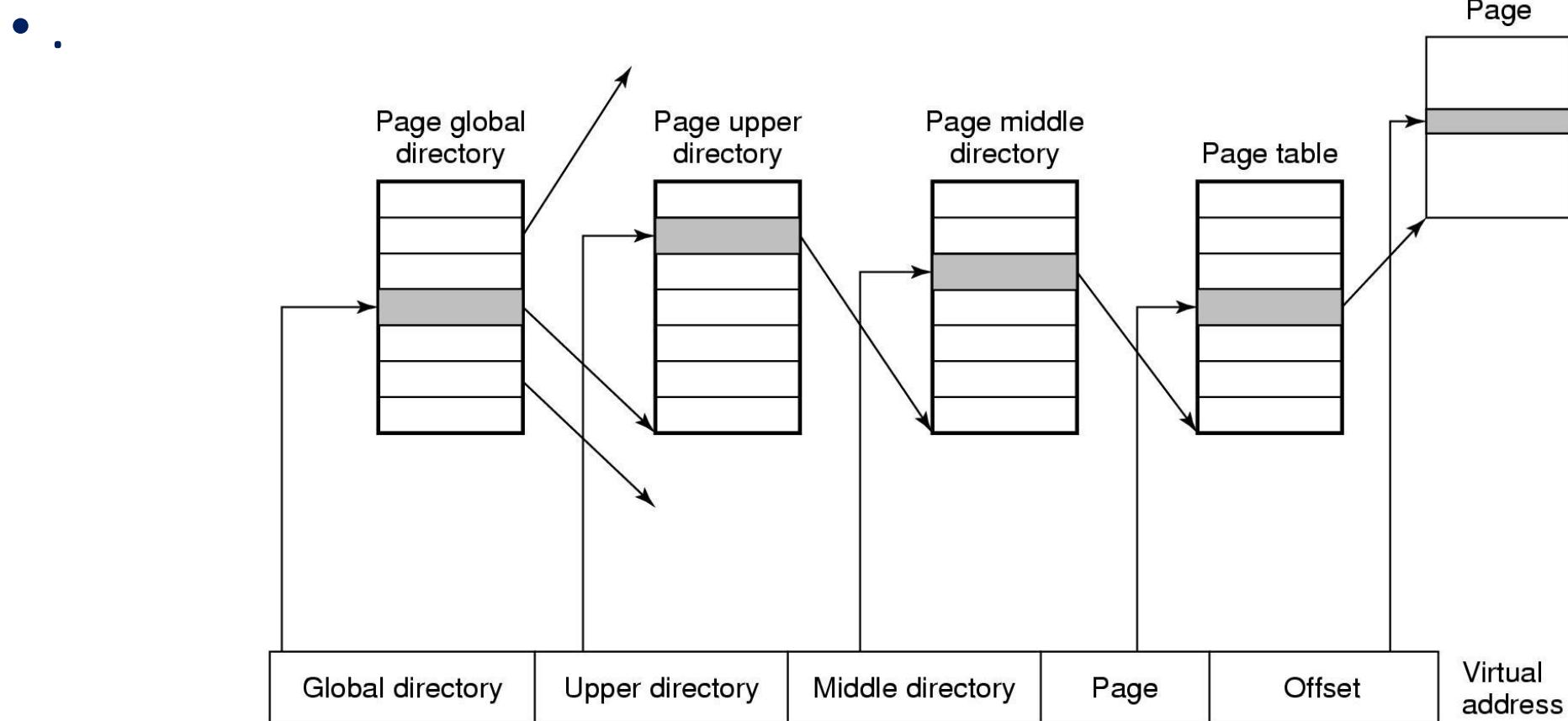


# Üç Düzeyli Sayfa Tablosu

- .
- .



# Dört Düzeyli Sayfa Tablosu

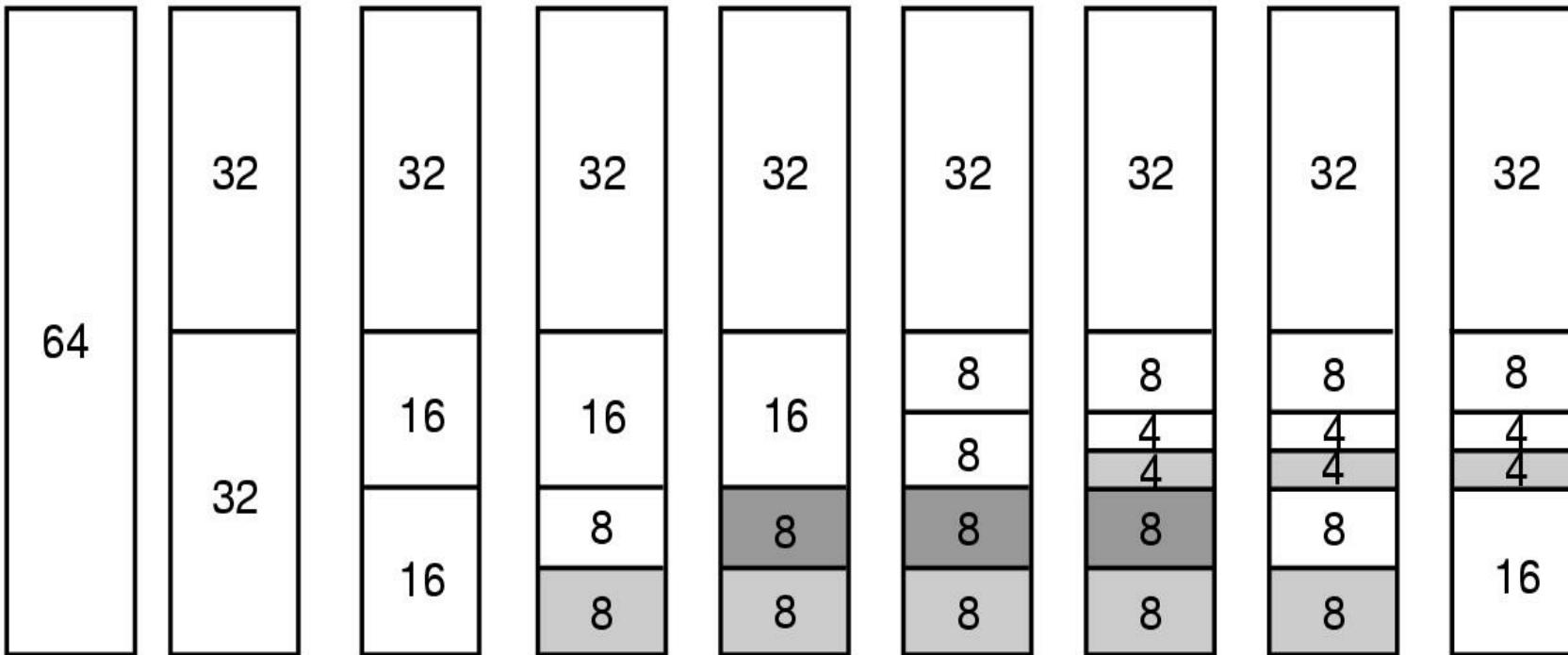


# Buddy Algoritması

- İşletim sistemlerinde belleği verimli bir şekilde yönetmek için kullanılan dinamik bir bellek ayırma stratejisidir.
- Belleği eşit boyutlu bloklara böler ve bunları, her düğümün bir bellek bloğunu temsil ettiği ikili ağaç yapısında düzenler.
- Bir süreç bellek istediğiinde, en küçük bloğu arar ve onu iki küçük bloğa böler.
- Süreç belleği serbest bırakırsa, bitişik blokların daha büyük bir blok oluşturmak için birleştirilip birleştirilemeyeceğini kontrol eder.
- Bloklar birleştirilebildiğinden, harici parçalanmayı azaltmaya yardımcı olur.
- İkili ağaç yapısını kullanarak mevcut en küçük bellek bloğunu hızlı bir şekilde bulabilir.

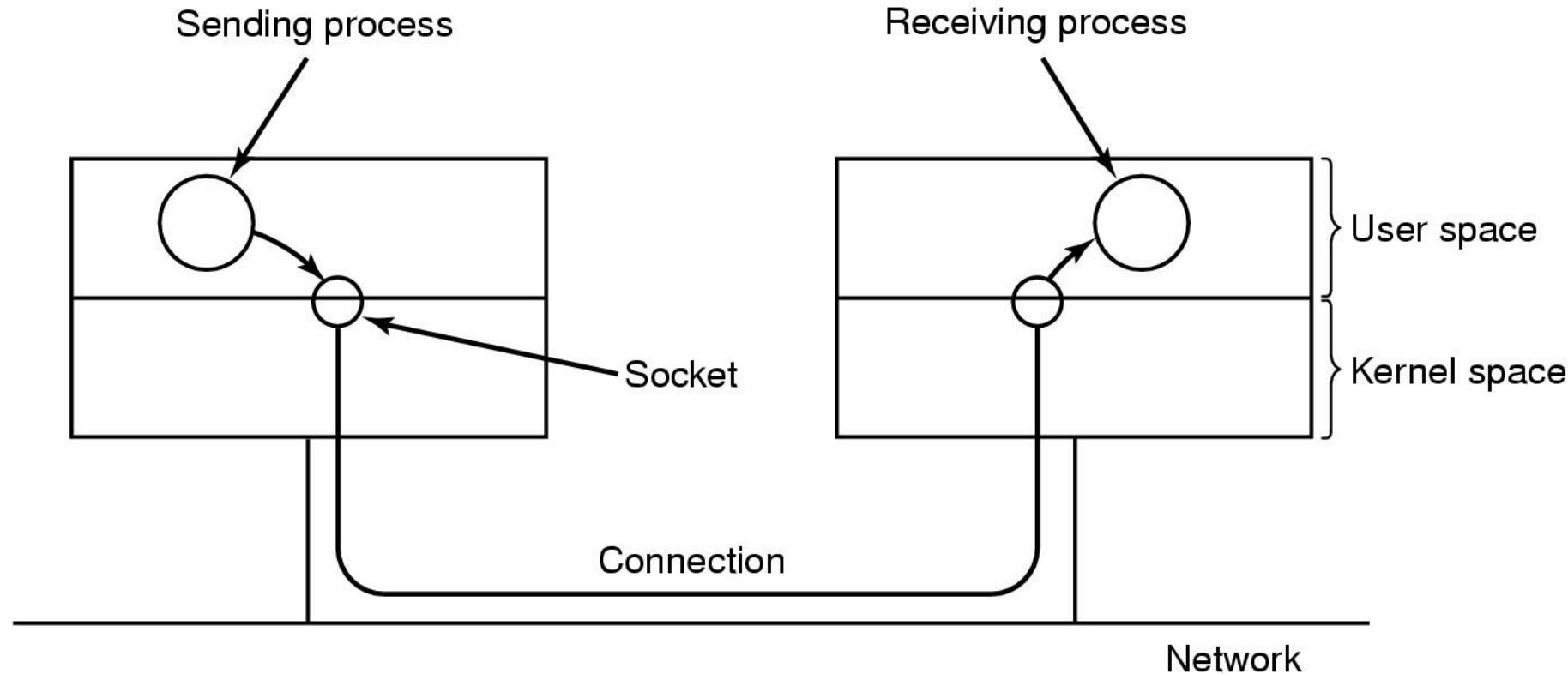
# Buddy Algoritması

- .
- .



# Ağ Haberleşmesi – Socket Kullanımı

- .
- .



# Terminal G/Ç Yönetimi

- cfsetospeed: çıkış baud hızını ayarlar.
- cfsetispeed: giriş baud hızını ayarlar.
- cfgetospeed: çıkış baud hızını alır.
- cfgetispeed: giriş baud hızını alır.
- tcsetattr: Bir terminal ile ilişkili parametreleri değiştirir.
- tcgetattr: Bir terminal ile ilişkili parametreleri alır.

# UNIX G/Ç

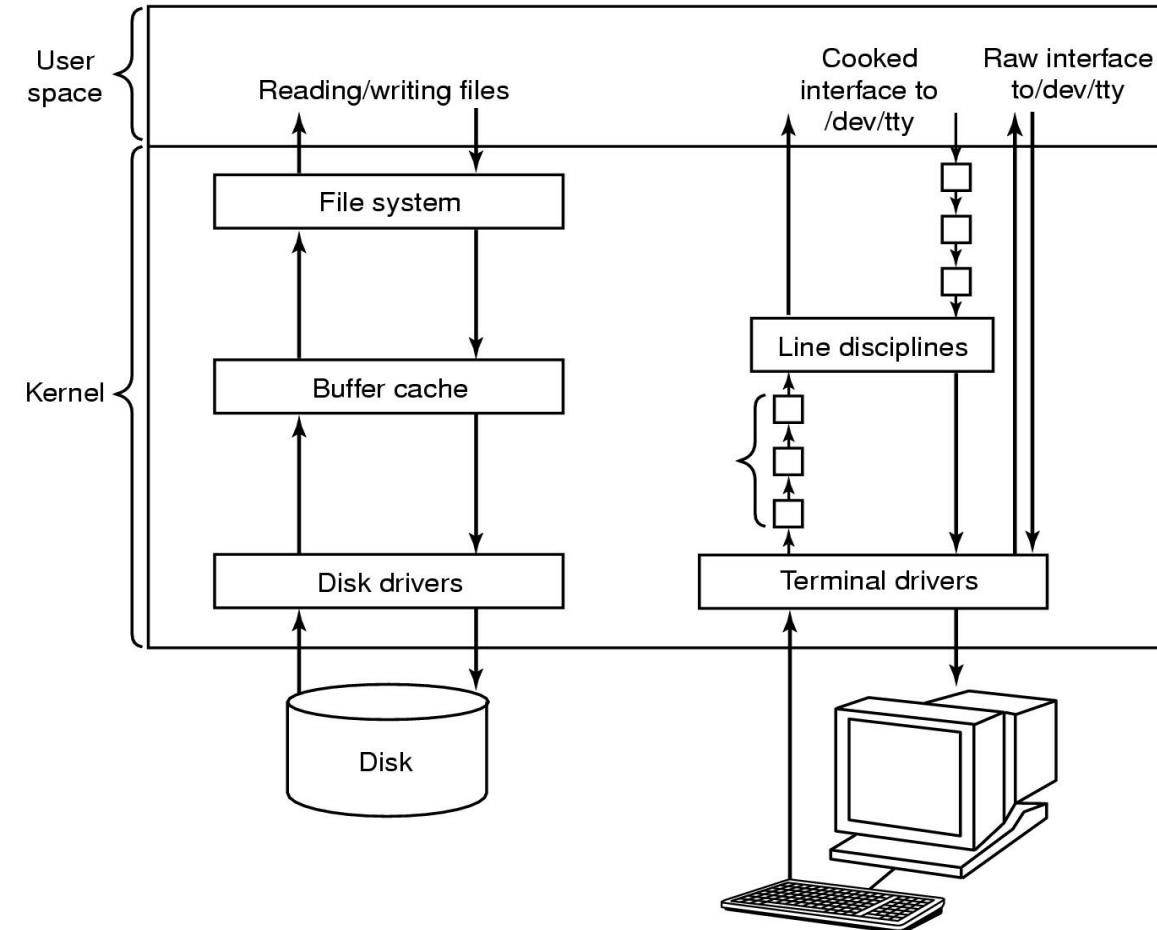
- mem\_read & mem\_write - Bellekten okur ve belleğe yazar
- k\_open - Bir klavye aygıtı açar
- k\_close - Bir klavye cihazını kapatır
- k\_read - Klavyeden veri okur
- k\_ioctl - Klavyenin davranışını kontrol eder
- tty\_open - Bir Tty cihazı açar
- tty\_close - Bir Tty cihazını kapatır
- tty\_read - Tty'den veri okur
- tty\_write - Tty'ye veri yazar.
- tty\_ioctl - Tty'nin davranışını kontrol eder

# UNIX G/Ç

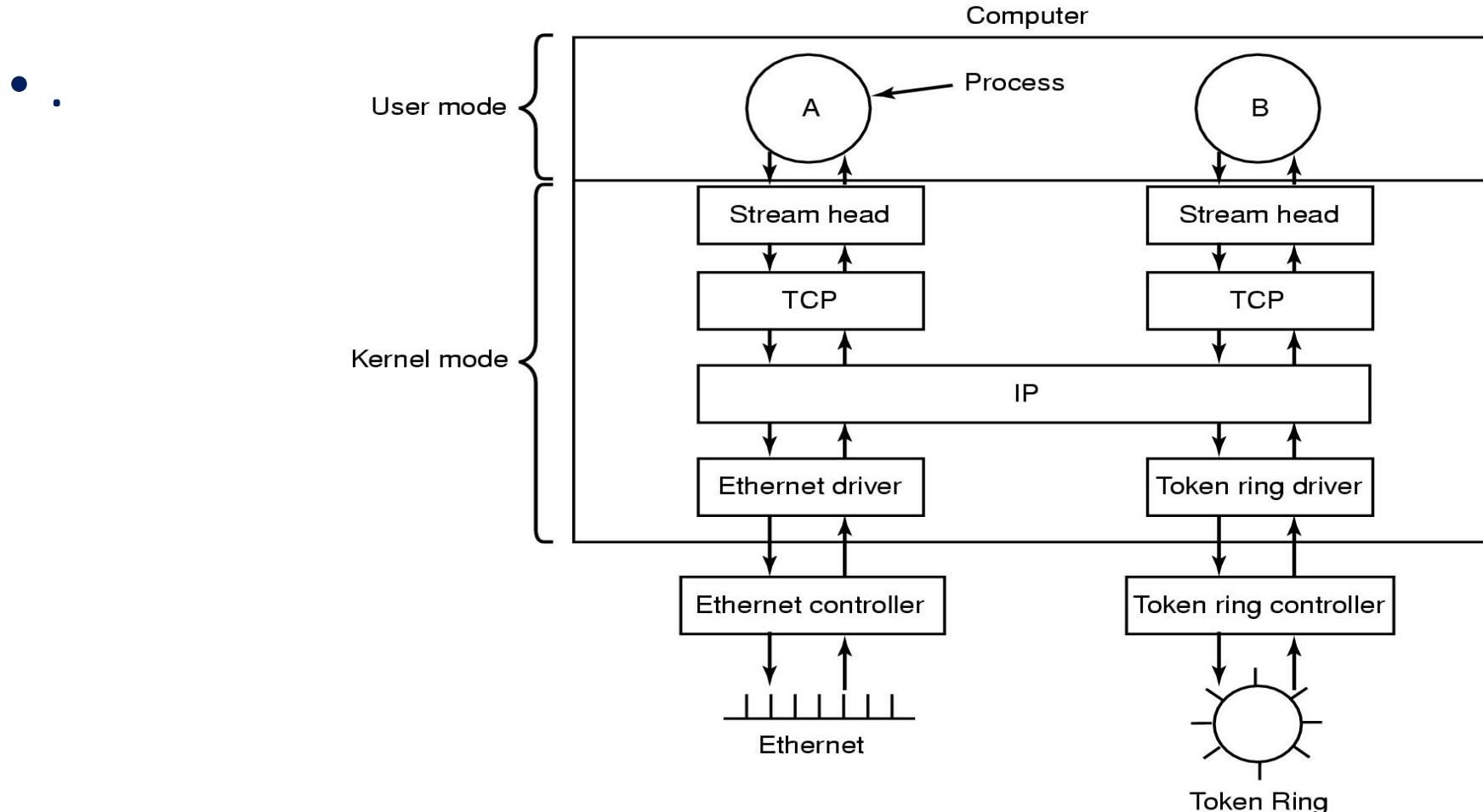
- pr\_open - Bir yazıcı aygıtını açar
- pr\_close - Bir yazıcı aygıtını kapatır
- pr\_write - Yazıcıya veri yazar
- pr\_ioctl - Yazıcının davranışını kontrol eder
- ip\_open - Bir IP cihazı açar
- ip\_close - Bir IP cihazını kapatır
- ip\_write - IP cihazına veri yaz
- ip\_ioctl - IP cihazının davranışını kontrol edin

# UNIX G/Ç Sistemi – BSD

- 
- 

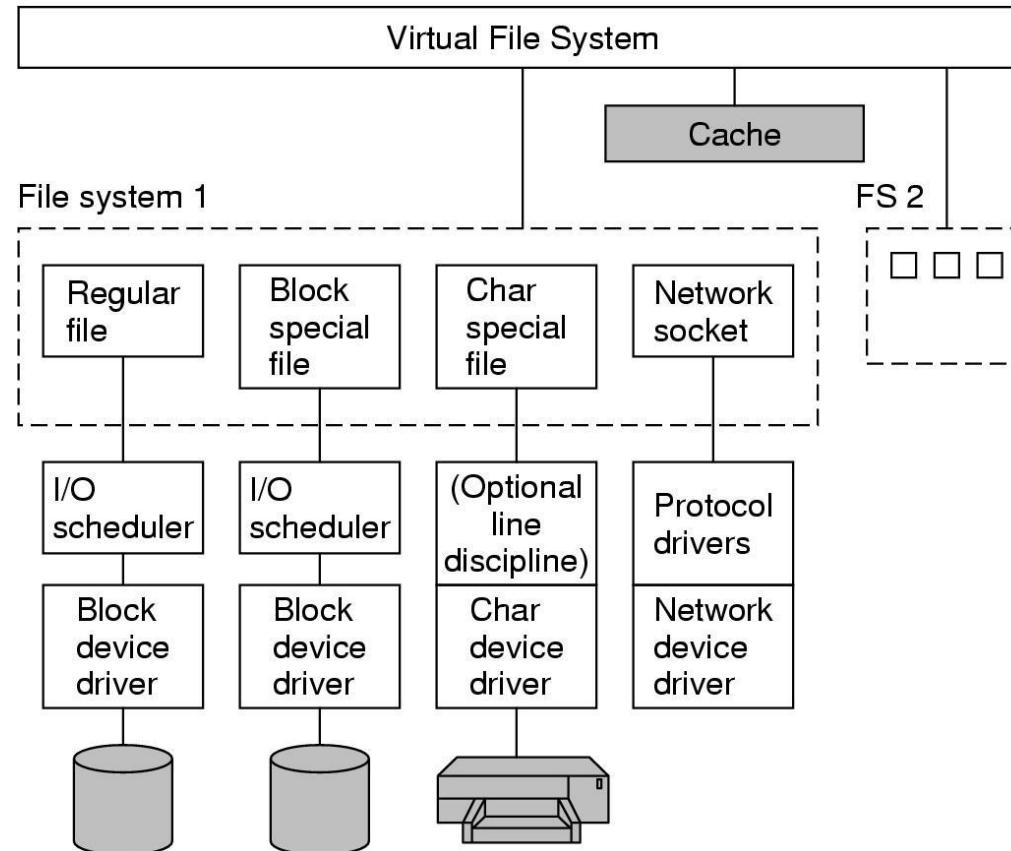


# Akışlar (streams) – System V



# Linux G/Ç Sistemi

- .
- .

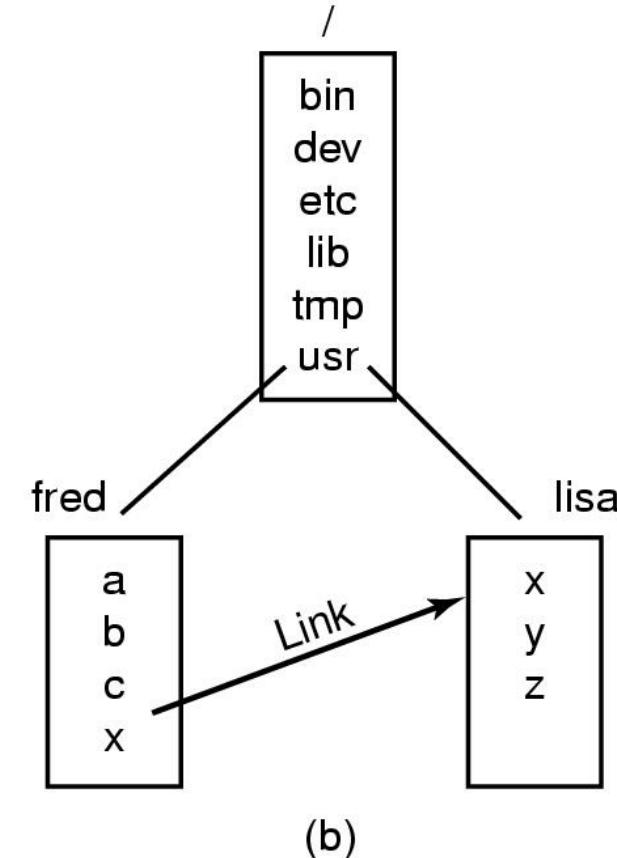
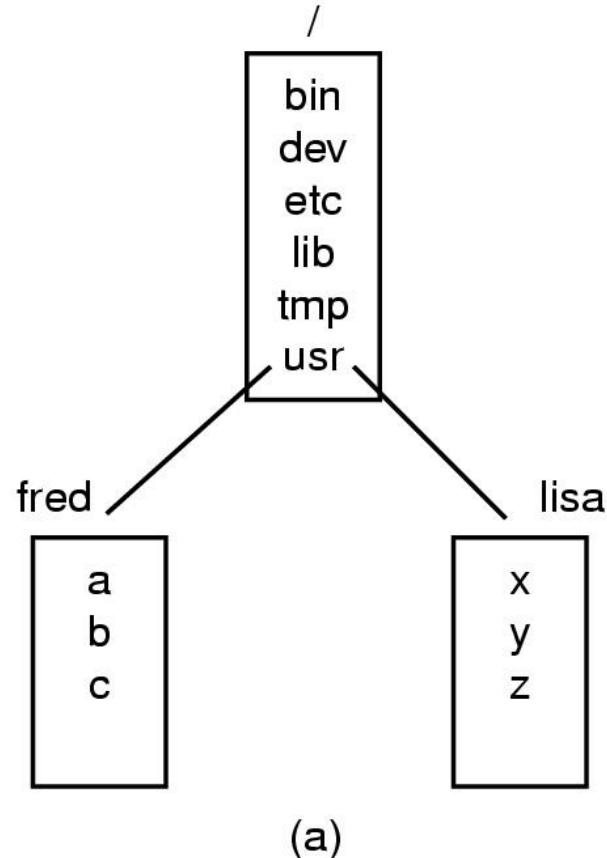


# Önemli Dizinler

- bin: Sistemin çalışması ve kullanıcıların temel görevleri gerçeklestirmesi için gerekli olan yürütülebilir dosyaları ve yardımcı programları içerir.
- dev: Sistemdeki donanım aygıtlarını temsil eden aygit dosyalarını içerir.
- etc: Sistem ve uygulamalar için yapılandırma dosyalarını içerir.
- lib: bin dizinindeki yürütülebilir dosyalar tarafından kullanılan kitaplıkları içerir.
- usr: Sırasıyla yürütülebilir dosyaları, kitaplıkları ve paylaşılan dosyaları içeren bin, lib ve share dahil kullanıcıyla ilgili dosya ve dizinleri içerir.

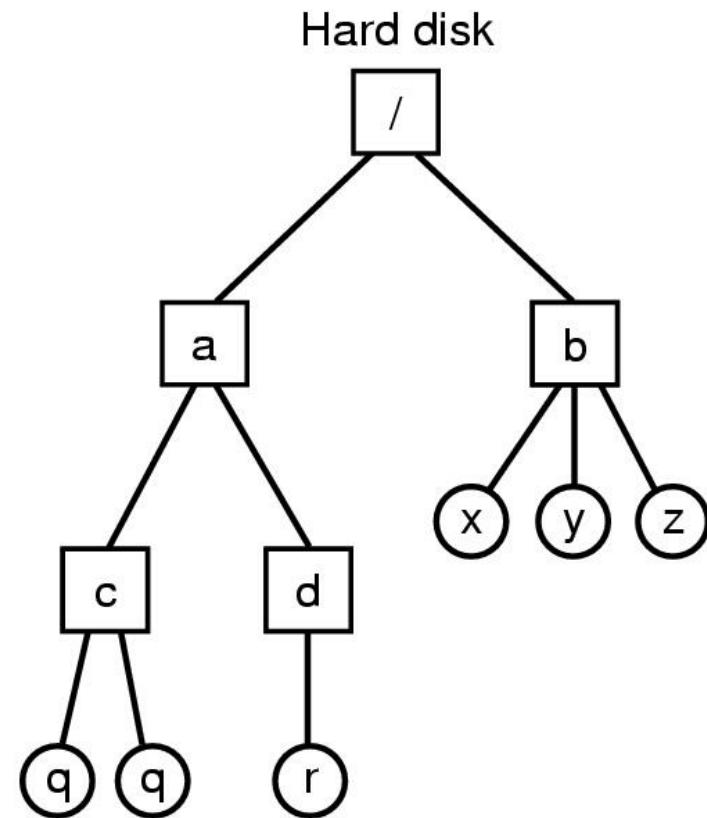
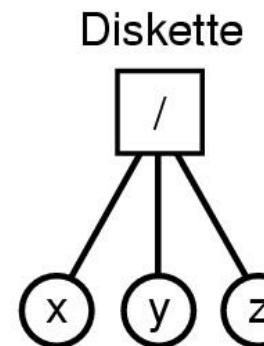
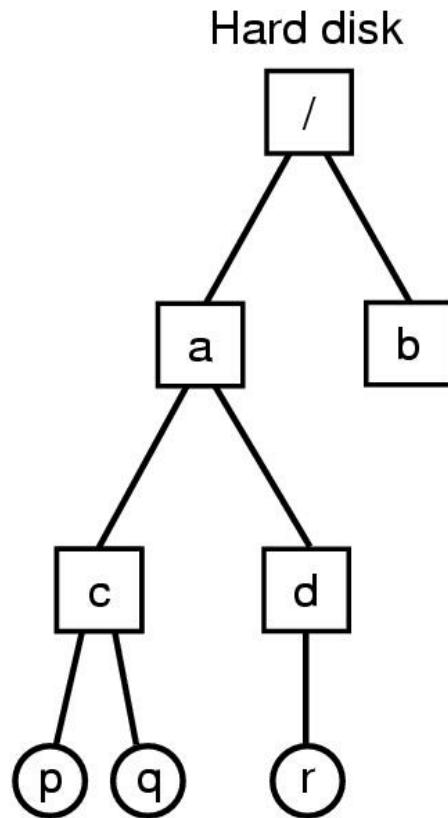
# Dosya Sistemi

- ..



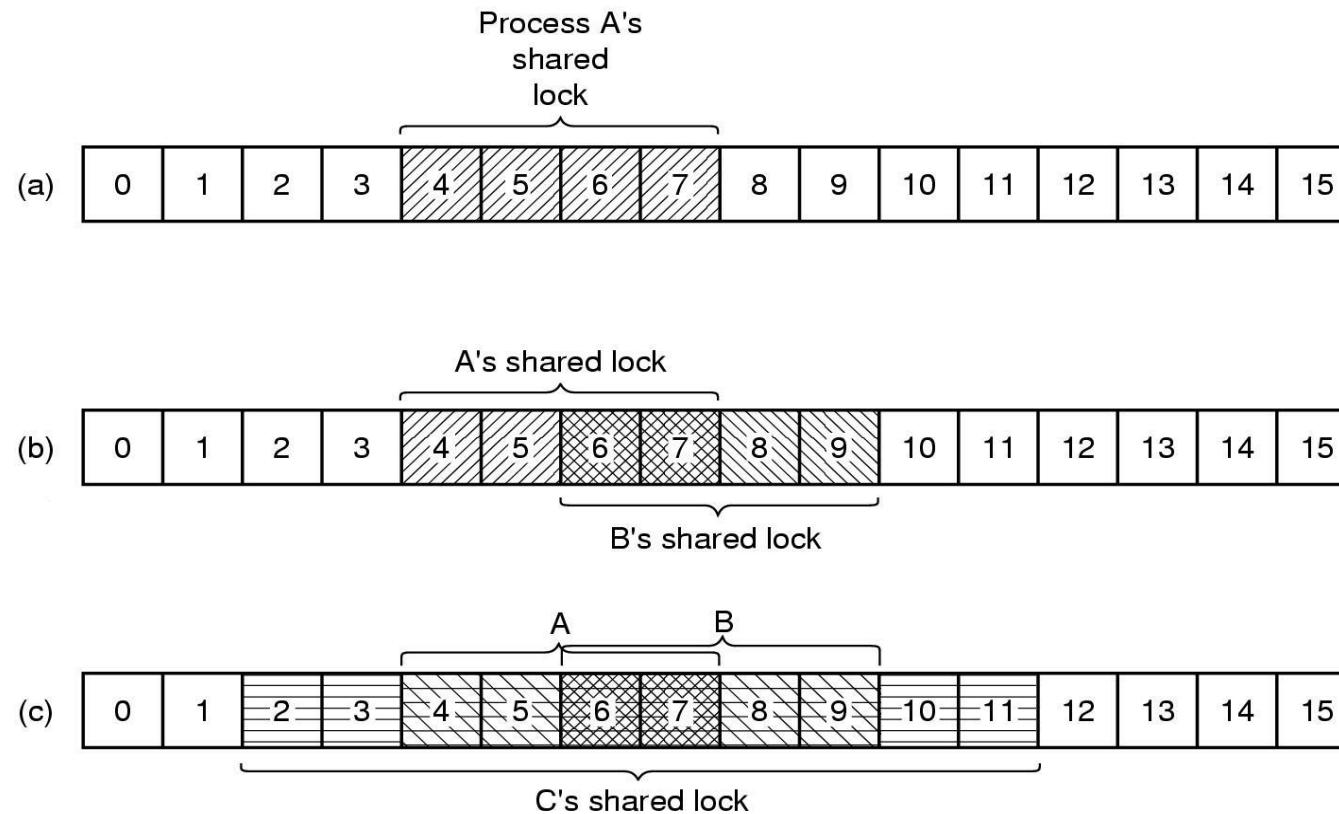
# Dosya Sistemi

• •



# Dosyaların Kitlenmesi (lock)

•



# Dosya Yönetimi Sistem Çağrıları

..

<b>System call</b>	<b>Description</b>
<code>fd = creat(name, mode)</code>	One way to create a new file
<code>fd = open(file, how, ...)</code>	Open a file for reading, writing or both
<code>s = close(fd)</code>	Close an open file
<code>n = read(fd, buffer, nbytes)</code>	Read data from a file into a buffer
<code>n = write(fd, buffer, nbytes)</code>	Write data from a buffer into a file
<code>position = lseek(fd, offset, whence)</code>	Move the file pointer
<code>s = stat(name, &amp;buf)</code>	Get a file's status information
<code>s = fstat(fd, &amp;buf)</code>	Get a file's status information
<code>s = pipe(&amp;fd[0])</code>	Create a pipe
<code>s = fcntl(fd, cmd, ...)</code>	File locking and other operations

# Istat Sistem Çağrısı Dönüş Değerleri

• .

Device the file is on
I-node number (which file on the device)
File mode (includes protection information)
Number of links to the file
Identity of the file's owner
Group the file belongs to
File size (in bytes)
Creation time
Time of last access
Time of last modification

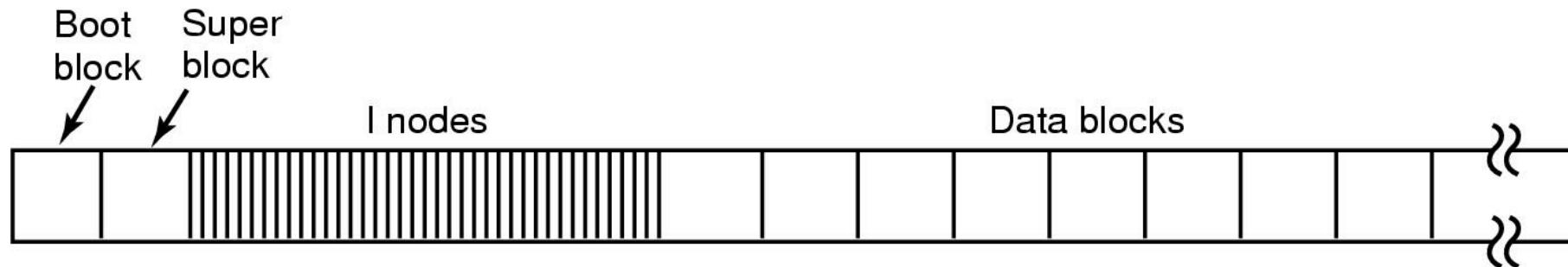
# Dizin Yönetimi Sistem Çağrıları

..

<b>System call</b>	<b>Description</b>
<code>s = mkdir(path, mode)</code>	Create a new directory
<code>s = rmdir(path)</code>	Remove a directory
<code>s = link(oldpath, newpath)</code>	Create a link to an existing file
<code>s = unlink(path)</code>	Unlink a file
<code>s = chdir(path)</code>	Change the working directory
<code>dir = opendir(path)</code>	Open a directory for reading
<code>s = closedir(dir)</code>	Close a directory
<code>dirent = readdir(dir)</code>	Read one directory entry
<code>rewinddir(dir)</code>	Rewind a directory so it can be reread

# UNIX'te Disk Düzeni

- .
- .

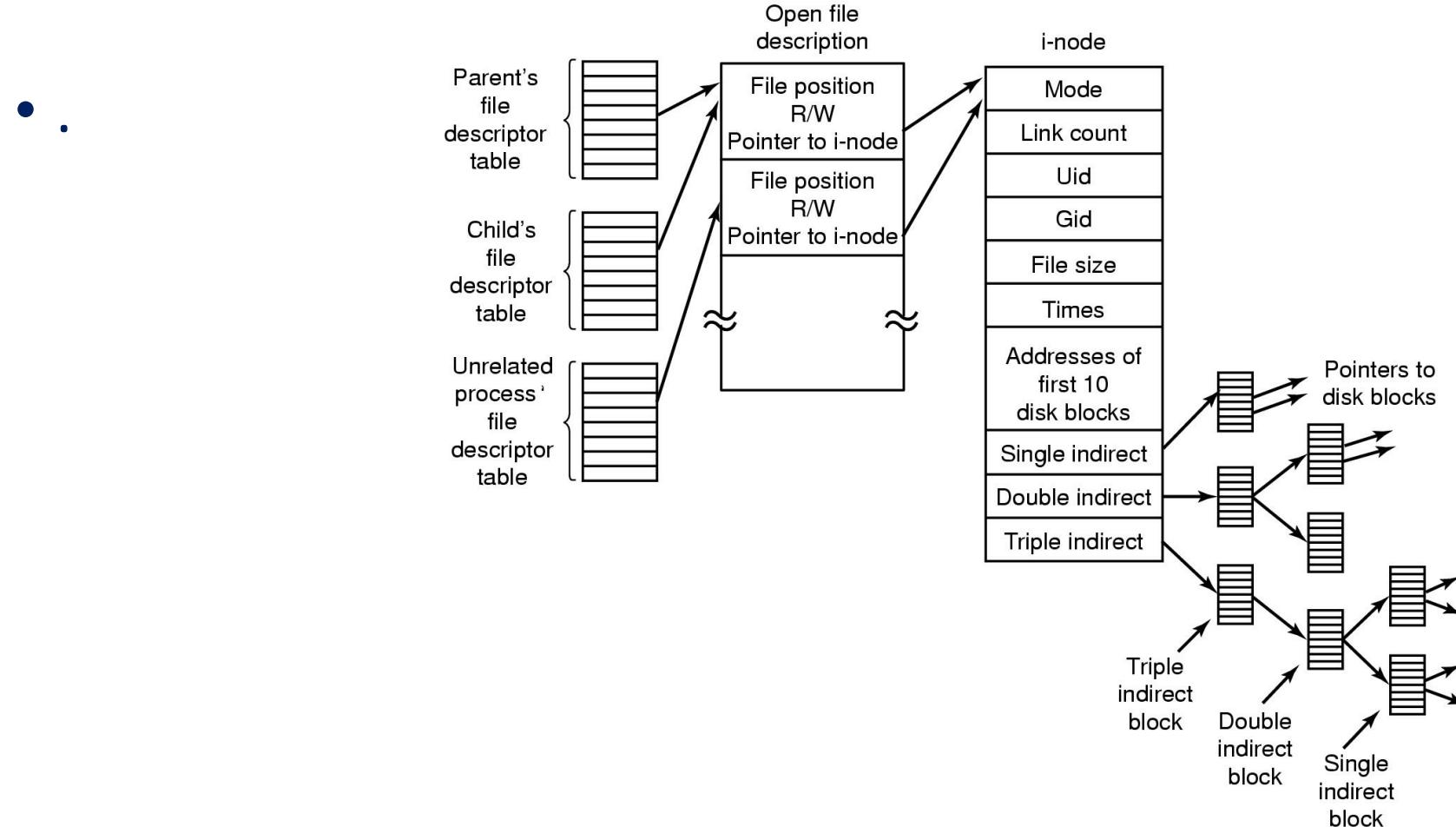


# I-node Yapısı

- .
- .

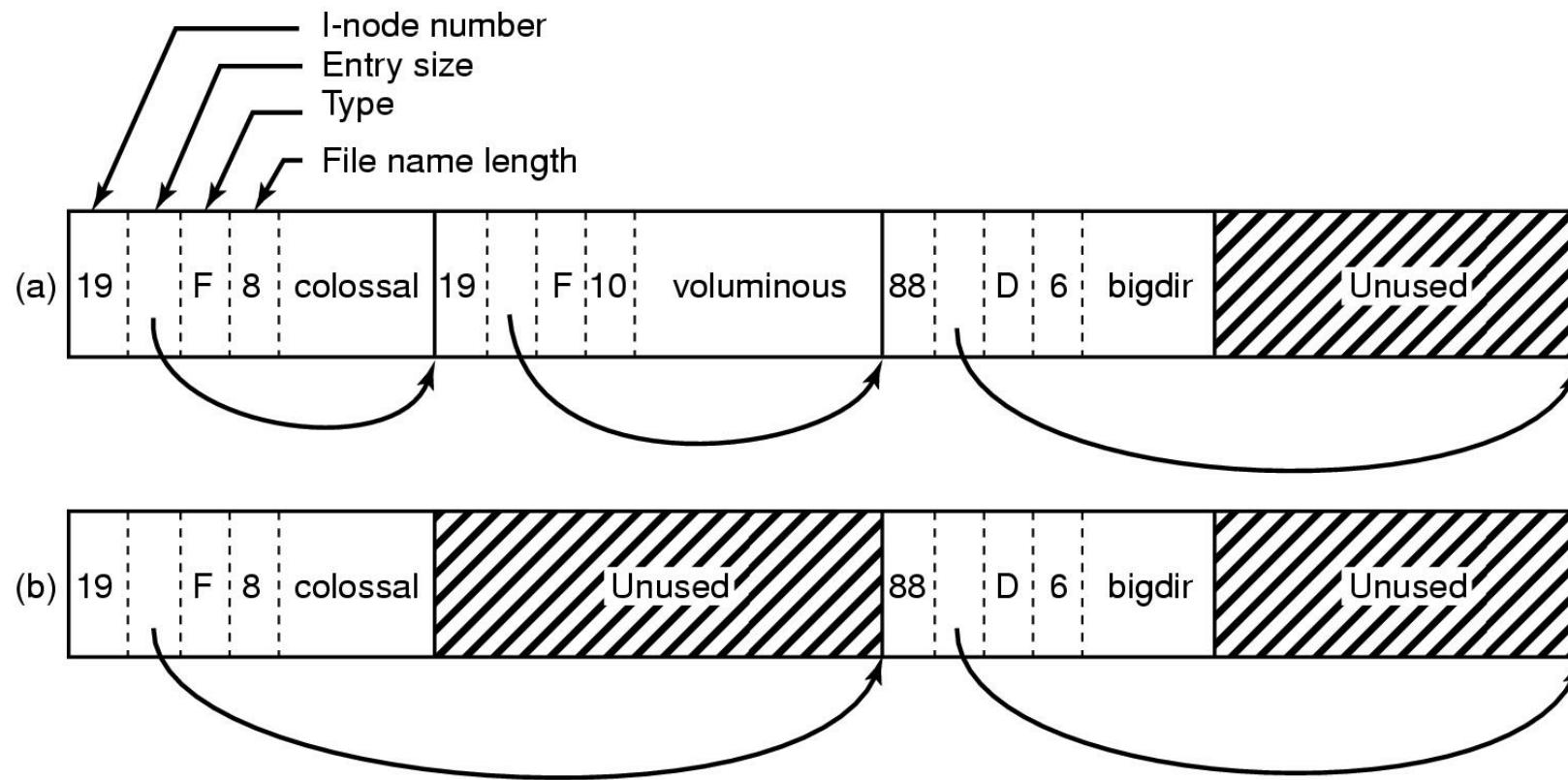
Field	Bytes	Description
Mode	2	File type, protection bits, setuid, setgid bits
Nlinks	2	Number of directory entries pointing to this i-node
Uid	2	UID of the file owner
Gid	2	GID of the file owner
Size	4	File size in bytes
Addr	39	Address of first 10 disk blocks, then 3 indirect blocks
Gen	1	Generation number (incremented every time i-node is reused)
Atime	4	Time the file was last accessed
Mtime	4	Time the file was last modified
Ctime	4	Time the i-node was last changed (except the other times)

# Dosya Tanımlayıcı Tablosu, Açık Dosya Açıklaması



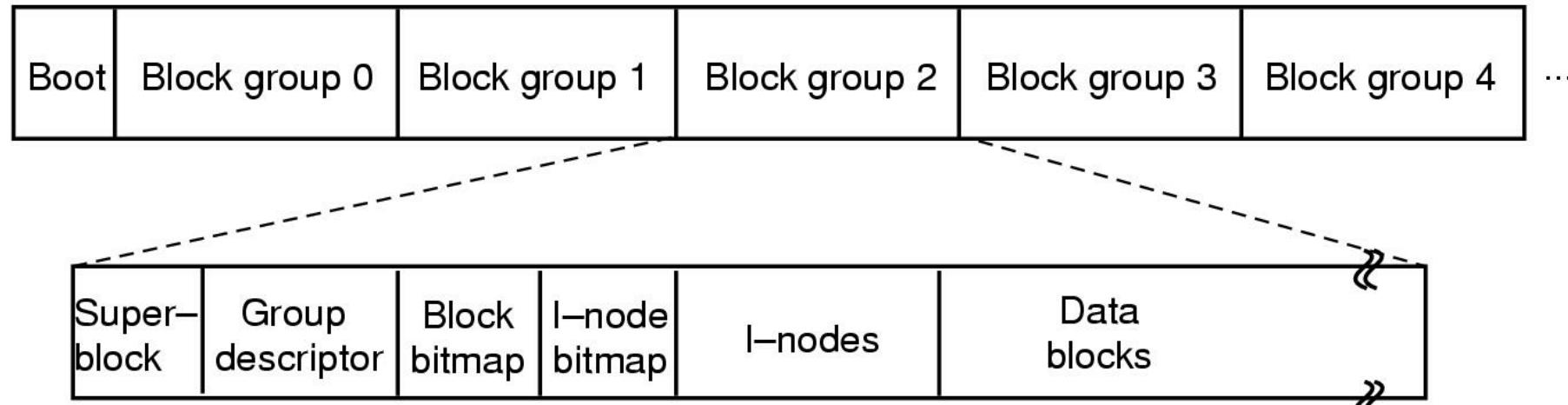
# Üç Dosyalı BSD Dizini

- .
- .

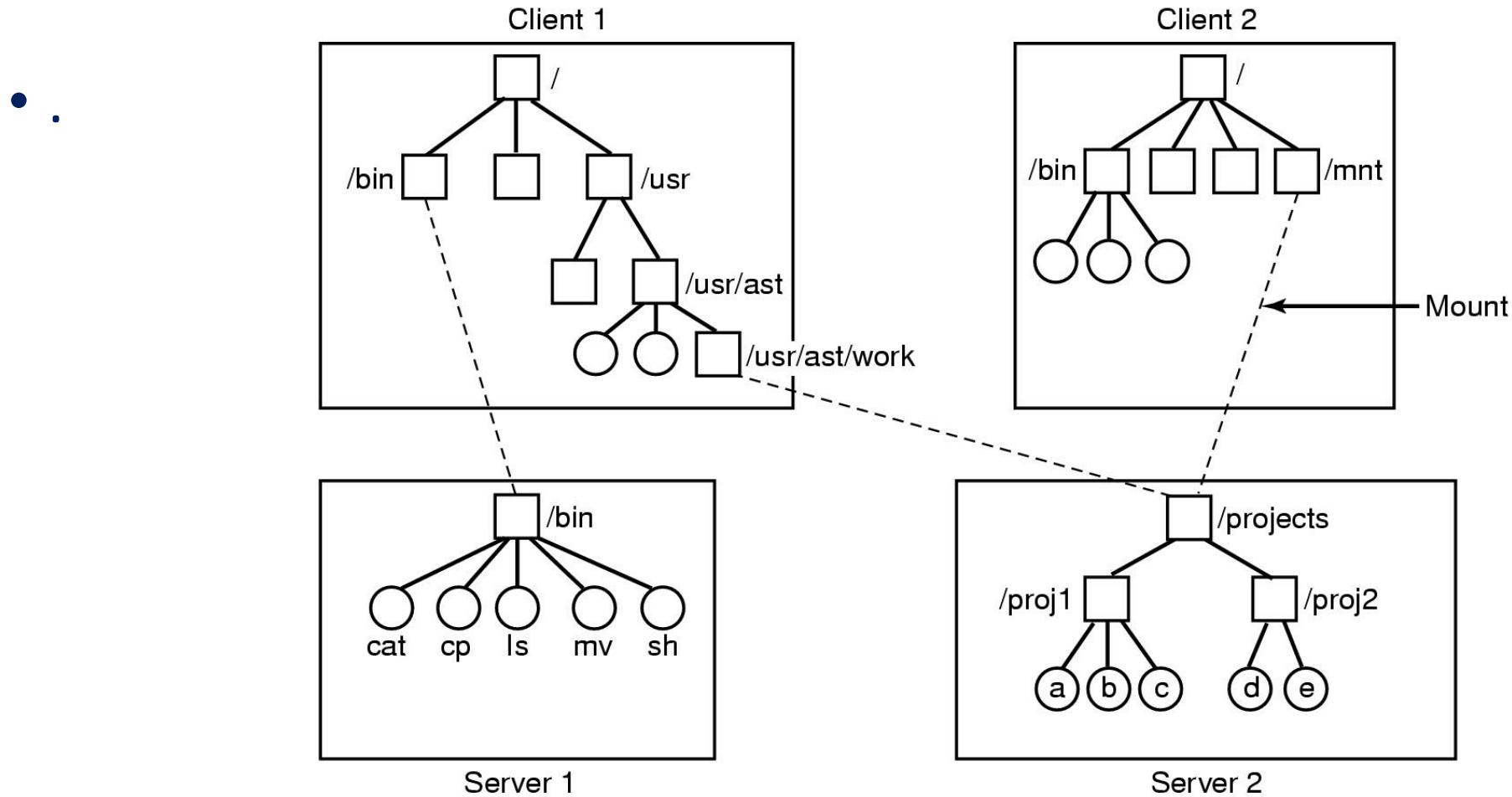


# Linux Ext2 Dosya Sistemi Düzeni

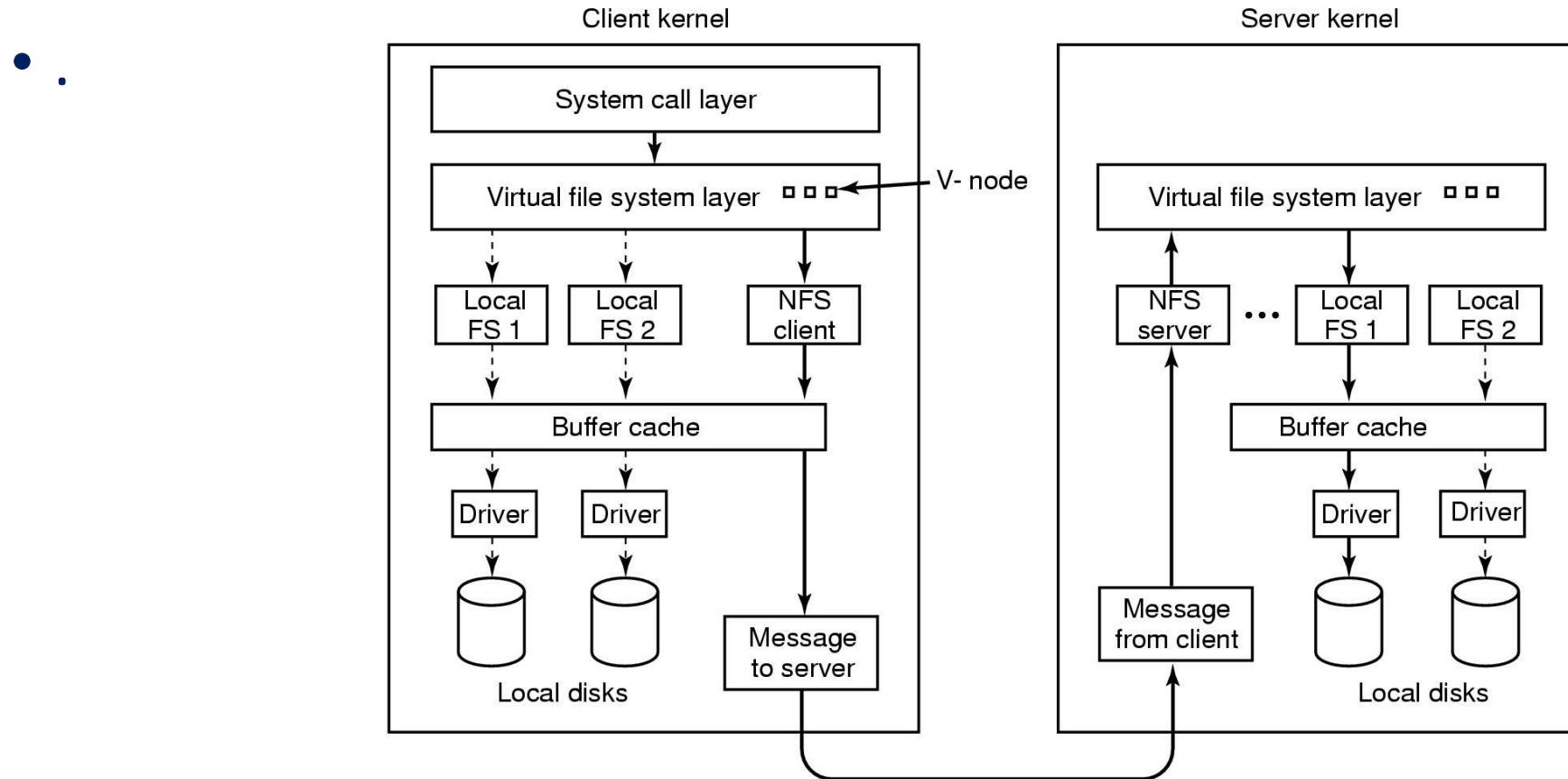
- .
- .



# Ağ Dosya Sistemi (NFS)



# NFS Katman Yapısı



# Dosya Koruma Durumları

..

<b>Binary</b>	<b>Symbolic</b>	<b>Allowed file accesses</b>
111000000	rwx-----	Owner can read, write, and execute
111111000	rwxrwx---	Owner and group can read, write, and execute
110100000	rw-r-----	Owner can read and write; group can read
110100100	rw-r--r--	Owner can read and write; all others can read
111101101	rwxr-xr-x	Owner can do everything, rest can read and execute
000000000	-----	Nobody has any access
000000111	-----rwx	Only outsiders have access (strange, but legal)

# Dosya Koruması Sistem Çağrıları

..

<b>System call</b>	<b>Description</b>
s = chmod(path, mode)	Change a file's protection mode
s = access(path, mode)	Check access using the real UID and GID
uid = getuid( )	Get the real UID
uid = geteuid( )	Get the effective UID
gid = getgid( )	Get the real GID
gid = getegid( )	Get the effective GID
s = chown(path, owner, group)	Change owner and group
s = setuid(uid)	Set the UID
s = setgid(gid)	Set the GID

# SON