
Linux
Kullanıcıları
Derneği

AKADEMİK BİLİŞİM
MERSİN ÜNİVERSİTESİ

# Linux/Unix Sistem Yönetimi Prosedürleri - 1

Emre Eryılmaz

emre.eryilmaz@linux.org.tr

Linux Kullanıcıları Derneği

7 Şubat 2014

# Linux =! Unix

- Linux , Unix değildir. Linux sadece bir Unix klonudur.Linux kernelinde README notu: “*Linux is a Unix clone written from scratch by Linus Torvalds with assistance from a loosely-knit team of hackers across the Net. It aims towards POSIX compliance.*”
- Linux sadece çekirdektir.Bir işletim sistemi değil.Linux dağıtımları Linux çekirdeği + GNU araçları + Grafik Arabirimi+ c/c++ derleyecisi vs.. araçların birleşiminden oluşur.Unix ise komple bir işletim sistemidir.
- Linux çekirdeği tamamen özgür ve ücretsizdir.Yaşasın Özgür Yazılım !!

# Linux =! Unix

- Linux kullanıcı dostudur.Ya Unix , Apple OS X de öyle :)
- Güvenlik duvarı yazılımı: Linux iptables,Solaris vs Unix sistemleri için 3rd party yazılım,ücretli.
- Dosya sistemleri,Linux ext3,ext4.Unix , jfs,gpfs,zfs,ufs vs...
- Farklı gelişim tarihleri.
- Bazı farklı sistem dizaynı.

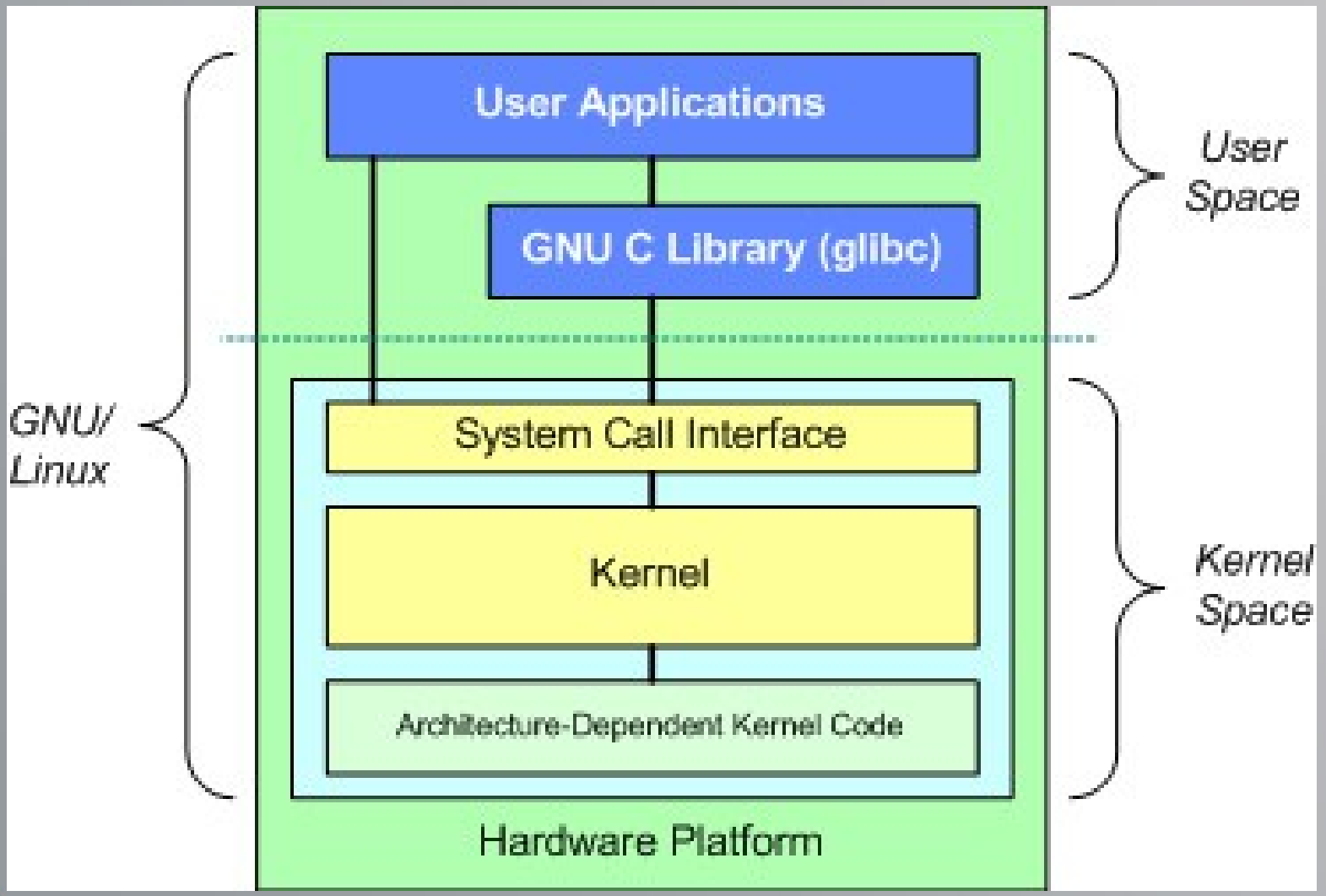
# Linux =! Unix

- Farklı kurulum prosedürleri.
- Farklı donanım cihaz isimlendirmeleri.
- Farklı komut ve araçlar.
- Farklı paket yönetimi.
- Farklı yama yönetimi.
- Farklı geliştirme araçları vs....

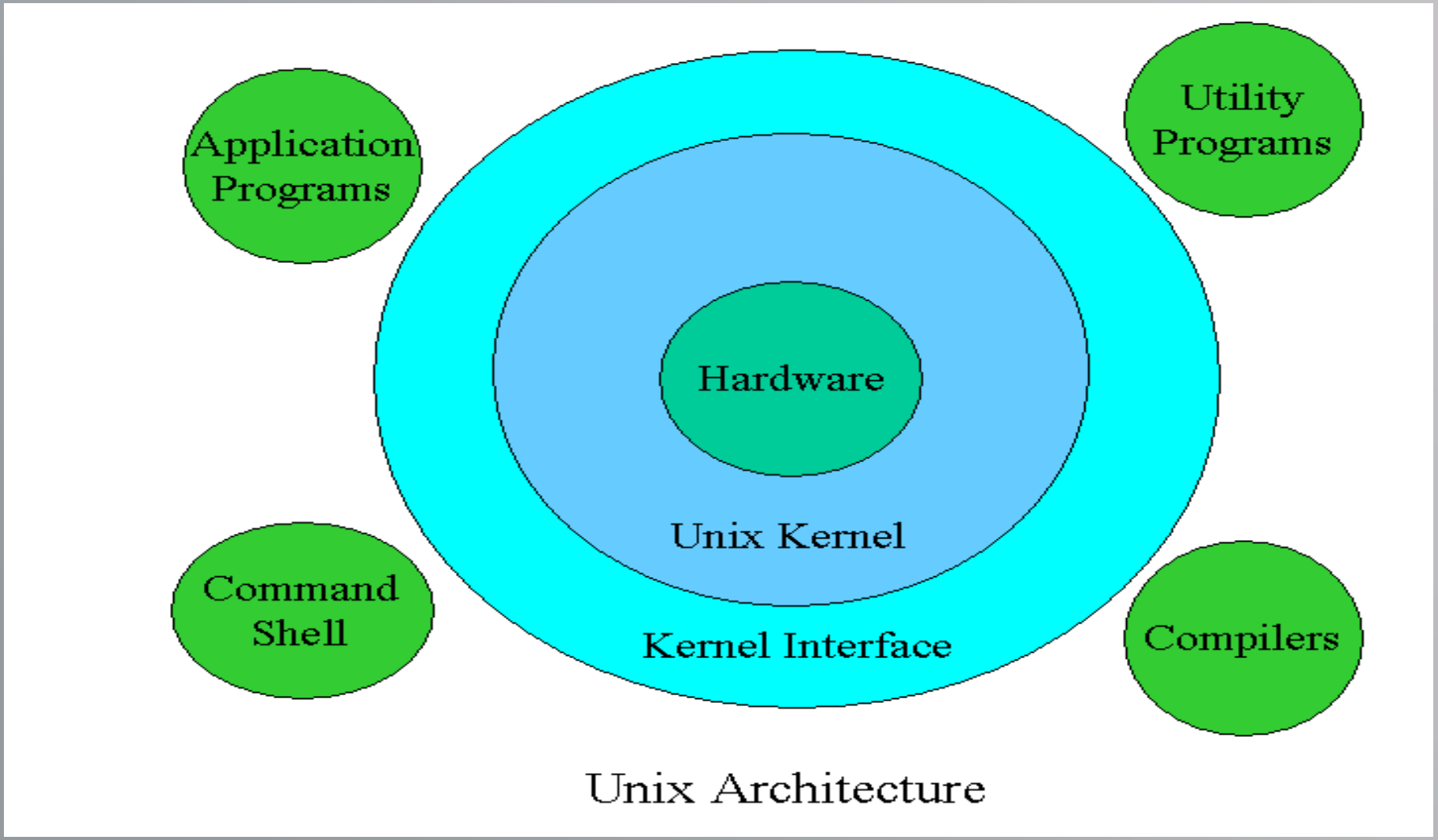
# POSIX

- POSIX = Portable Operating System Interface for Unix, Unix için Taşınabilir İşletim Sistemi Arabirimi
- IEEE tarafından kabul edilmiş bir standart.
- Aslında POSIX, application programming interface(API)'lerini , shell komut satırı uzunluğunu,araçların arayüzlerini tanımlar.
- İsim babası , Richard Stallman'dır.

# Sistem Yapısı



# Sistem Yapısı

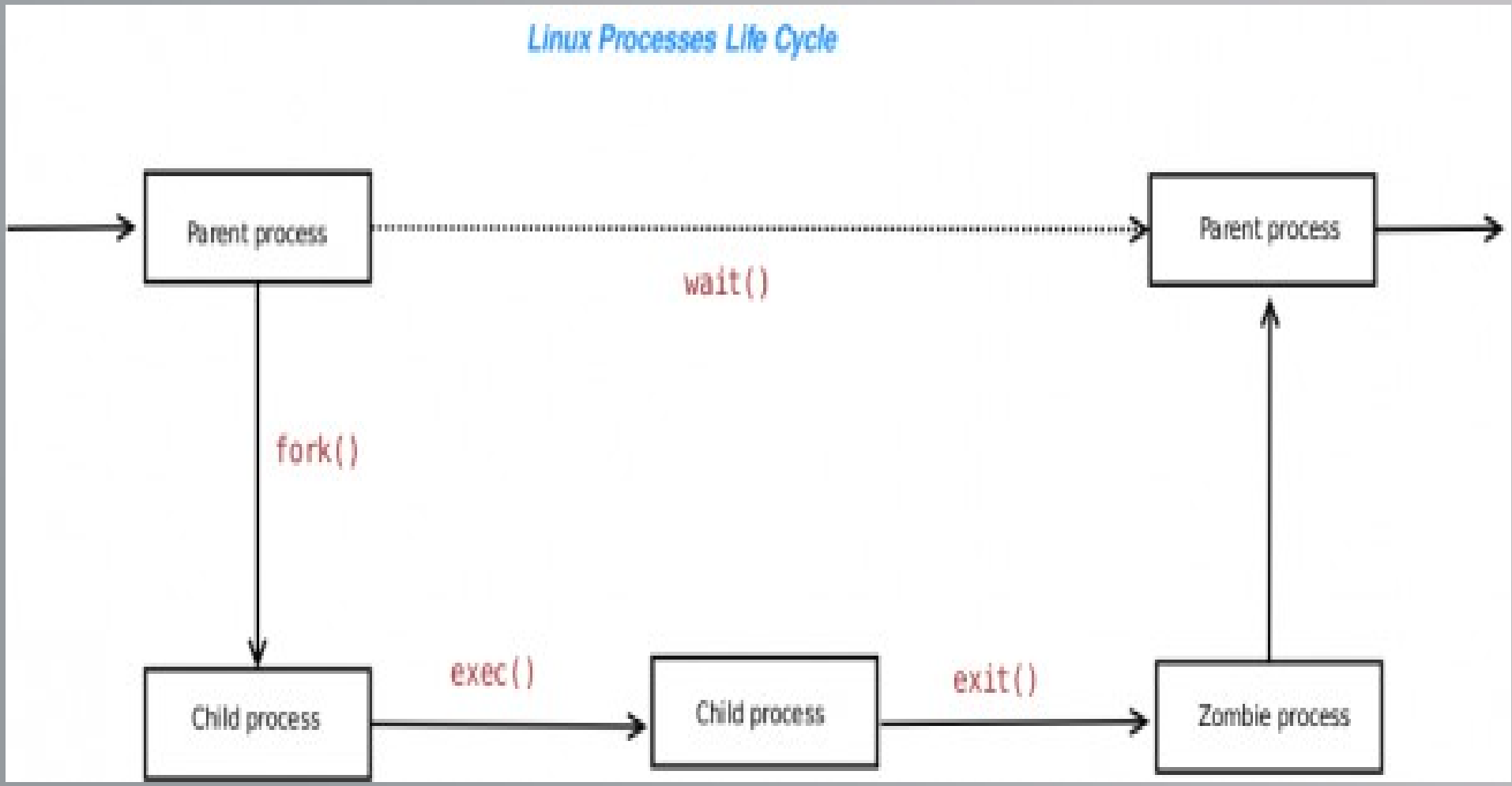



Unix Architecture

# Prosedürler

- Processes Yönetimi
- Disk Yönetimi
- Boot Yönetimi
- Sistem Yönetimi
- Network Yönetimi
- Kernel Yönetimi
- Backup Yönetimi

# Prosedür 1 : Processes Yönetimi



# Prosedür 1: Processes Yönetimi

- **Processes**: Sistem üzerinde çalışan derlenmiş her kod.
- **PID**: Processes ID.Her processes bir ID'ye sahiptir.
- **PPID**: Her processes bir üst(parent) processes'e sahiptir.Her child processes bir parent processes tarafından başlatılır.
- **İnit**: Processes ID'si 1.Kernel tarafından çağırılır ve çalıştırılır.Parent processesi yoktur.

# Prosedür 1: Processes Yönetimi

- **Kill** : Herhangi bir processes çalışmasını durduğunda  ölür.Eğer bir processesi durdurmak istiyorsanız onu öldürmeniz lazım.kill all them !!
- **Daemon**: Bir processes sistem açılışında çalışmaya başlayıp sonsuza kadar çalışmaya devam ediyorsa bu daemon'dur.Daemon öldürülemez!
- **Zombie**: Sistem üzerinde çalışan herhangi bir processesin çalışmasını durduğunuzda ölür.Durdurulan processesler sistemde zombie olarak görünür.Zombie'ler öldürülemez.Zaten ölüdürler :)

# Prosedür 1 : Processes Yönetimi

- $$ => processes(child) ve $PPID => parent processes:
  $ echo $$ $PPID
  2334  2333
- Pidof => herhangi bir processesin ID'sini adıyla bulabiliriz:
  $ pidof firefox
  2446

# Prosedür 1: Processes Yönetimi

- “ps” komutu, processes'lere bakmak için en sık kullanılan araçtır.
  $ ps fx  (sadece komutu veren kullanıcının başlattığı processesler)
  $ ps fax ( sistem üzerindeki tüm processesler)
  $ ps -C bash ( komut adına göre processes arama) psgrep komutuda kullanılabilir.
- “top” komutuda kullanılabilir.

# Prosedür 1 : Processes Yönetimi

- “kill” komutu processesleri durdurmak için ;
  $ kill 1345 => processes ID
- Sinyaller:
  $ kill -l
- SIGHUP(-1) : processesi yeniden konfigürasyon dosyasını okumasını sağlar.
  $ kill -1 1
- SIGTERM(-15): öntanımlı kill sinyali.
  $ kill -15 3456

# Prosedür 1: Processes Yönetimi

- SIGKILL(-9): SIGTERM sinyalinden farklı olarak ,processesi öldürür.kill sinyalini processesin kendisine değil,direk olarak linux çekirdeğine gönderir.Processesi zorla sonlandırır.
  $ kill -9 4567
- “killall” komutu processesin adıyla öldürmemizi sağlar.Öntanımlı 15 sinyalini gönderir.”pkill” komutu da benzer işleve sahiptir.
  $ killall firefox

# Prosedür 1: Processes Yönetimi

- SIGSTOP(-19): Herhangi bir processesi askıya almak için gönderilir.
  $ kill -19 processes_id
- SIGCONT(-18): Askıya alınmış processesi tekrar devam etmesine için bu sinyal gönderilir.
- “top” aracı ile “k” parametresi vererek processes'ler ve belirli sinyaller gönderilerek sonlandırabilir.

# Prosedür 1: Processes Yönetimi

- **Processes önceliği**: Sistem üzerinde processes önceliği 0-20 arasında değişebilir.
  “renice” komutu ile herhangi bir processesin önceliğini değiştirebiliriz.Fakat normal kullanıcılar sadece önceliği '+' olarak verebilir.Sadece root kullanıcısı '-' değeri vererek processes önceliğini değiştirebilir.Negatif öncelik verirken dikkat edin.Sistemi tamamen durdurabilir ya da zarar verebilirsiniz.

  $ renice +8 2936

# Prosedür 1: Processes Yönetimi

- “nice” komutu ile yazdığınız herhangi bir script ya da uygulamanın belirli önceliğe sahip olarak başlamasını sağlayabilirsiniz.
  $ nice -5 ./backup.sh
- Background Processesler: “jobs” komutu ile background çalışan processesleri görebilirsiniz.Örnek:
  $ vi test.sh  ( Ctrl+Z tuş kombinasyonu ile backgrounda processesi yollayın.)
  $ jobs

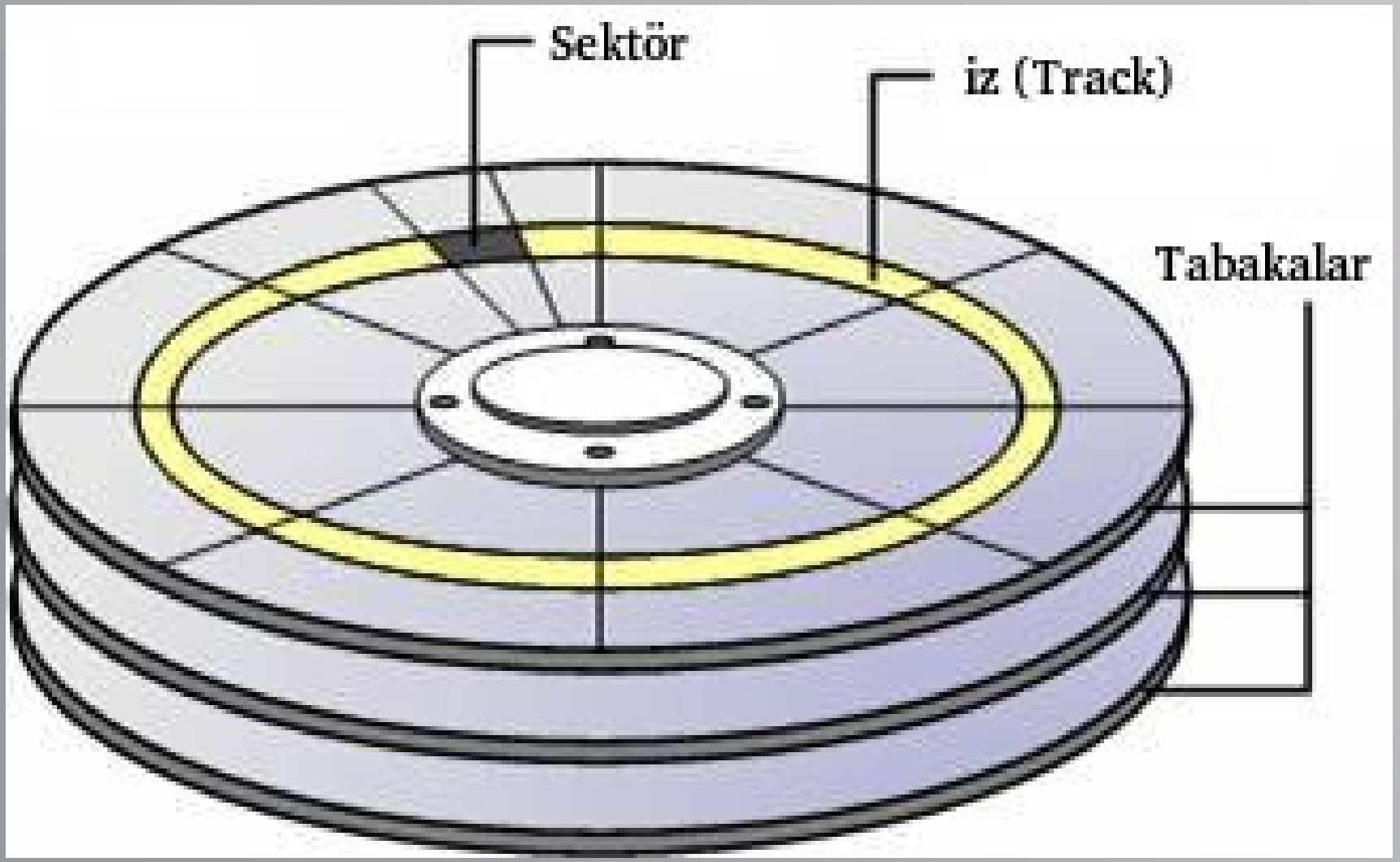
# Prosedür 1 : Processes Yönetimi

- “& ampersand” : processesi başlatırken & karakteri ile işlemi backgrounda alabilirizi.Örnek:
  $ find / > tr.iso 2> /dev/null &
  $ jobs
- “jobs -p” komutu ile background'a atılan processeslerin ID'sini görebilirsiniz.
  $ sleep 500 &
  $ sleep 400 &
  $ jobs -p

# Prosedür 1 : Processes Yönetimi

- “fg” komutu ile background'a atılan processesleri foreground'a çıkartabiliriz.
  $ sleep 200 &
  $ fg 2
- “bg” komutu ile askıya alınan processesleri tekrar çalışmaya başlatabiliriz.Daha önce bahsettiğimiz SIGCONT sinyalini gönderir.
  $ sleep 300 &
  $ jobs
  $ bg 2

# Prosedür 2: Disk Yönetimi



# Prosedür 2 : Disk Yönetimi

- block device ve character device
  $ ls -l /dev/
- IDE ya da SCSI & ATA ya da SCSI
- ATA: her bus üzerinde iki adet cihaz bulunur.Master ve Slave.Jumber ile manual olarak ayarlanır.
- SCSI: “small computer system interface”.SCSI birden fazla aygıtı destekleyebilir.

# Prosedür 2 : Disk Yönetimi

- İde0 => /dev/hda(master) ve /dev/hdb(slave)
  ide1 => /dev/hdc(master) ve /dev/hdd(slave)
- Scsi => /dev/sda-z arasında isimlendirebilir.
- Sistem üzerinde diskleri listelemek için;
  # fdisk -l
  # dmesg | grep -i “scsi disk”
  # dmesg | grep "[hs]d[a-z]"
  # lshw (bazı dağıtımlarda öntanımlı gelebilir.)
  # lsscsi
  # cat /proc/scsi/scsi

# Prosedür 2 : Disk Yönetimi

- Disk bölümlerini görüntüleme:
  # fdisk -l
  # cat /proc/partitions
- MBR ( Master Boot Record) : diskiniz üzerindeki partitions tablosu bilgileri burada tutulur.primary ya da extended bölümler gibi.
  # dd if=/dev/sda of=/SCSIdisk.mbr bs=512 count=1
  (partiton bilgilerini kopyalama)
  # dd if=/dev/zero of=/dev/sda bs=512 count=1
  (mbr silmek için.tehlikeli!)
  # dd if=/dev/zero of=/dev/sda
  (tamamen harddiski silmek için)

# Prosedür 3 : Disk Yönetimi

- Dosya sistemi: disk üzerindeki dosyaların organize edilmesidir. Bir işletim sisteminin bir disk veya bölümleri üzerindeki dosyalarının izlerini bulmak için kullandığı yapı ve yönteme dosya sistemi (filesystem) denir.
- ext2,ext3,ext4
- Sistem tarafından desteklenen dosya sistemleri:
  # cat /proc/filesystems
  # cat /proc/filesystems | grep -v nodev

# Prosedür 2 : Disk Yönetimi

- Dosya sistemi kontrolü:
  # ls /sbin/*fsck* (kontrol komutları listeleme)
- /etc/filesystems ve /proc/filesystems içinde tanımlı olan dosya sistemleri için -t (mount type) belirtmeye gerek yoktur.
- Bağlanmış dosya sistemlerini görünteleme:
  # mount | grep /dev/sda
  # cat /proc/mounts | grep /dev/sda
  # cat /etc/mtab | grep /dev/sda
  # df
  # du

# Prosedür 2 : Disk Yönetimi

- Örnek bir disk bölümleme ve dosya sistemi biçimlendirme.

# Prosedür 2 : Disk Yönetimi

- “hdparm” aracı ile disk üzerinde tuning işlemleri yapabiliriz.
  # hdparm -I /dev/sda (Disk hakkında bilgiler)
  # hdparm -t /dev/sda (Disk hızını ölçmek için, -T cache okuma hızını ölçmek için.)


Linux
Kullanıcıları
Derneği