

Born2beroot detayına kadar öğrenme vol1

|—> lsblk(list block devices) komutu ile disk bölümlediğimiz ve birimlediğimiz alanlar nedir, blok isimlendirmeleri nedir?

NAME	MAJ:MIN	RM	SIZE	RO	TYPE	MOUNTPOINT
sda	8:0	0	30.8G	0	disk	
└─sda1	8:1	0	476M	0	part	/boot
└─sda2	8:2	0	1K	0	part	
└─sda5	8:5	0	30.3G	0	part	
└─sda5_crypt	254:0	0	30.3G	0	crypt	
└─LVMGroup-root	254:1	0	9.3G	0	lvm	/
└─LVMGroup-swap	254:2	0	2.1G	0	lvm	[SWAP]
└─LVMGroup-home	254:3	0	4.7G	0	lvm	/home
└─LVMGroup-var	254:4	0	2.8G	0	lvm	/var
└─LVMGroup-srv	254:5	0	2.8G	0	lvm	/srv
└─LVMGroup-tmp	254:6	0	2.8G	0	lvm	/tmp
└─LVMGroup-var--log	254:7	0	3.7G	0	lvm	/var/log
sr0	11:0	1	1024M	0	rom	

—>lsbk(list block devices) nedir?

lsblk komutunun, /sys/dev/block kullanılarak yapılan her bir blok aygıtına bakmaktadır. Blok aygıtlarla ilgili ayrıntıları görüntülemek için kullanılır ve bu blok aygıtları(ram disk hariç, bu bir hata değil özellikle) temelde bilgisayara bağlı aygıtları temsil eden dosyalardır. Çıktayı temel olarak ağaç benzeri bir yapıda vermektedir, "lsblk -T" (tree)komutu ile aynı işlevdedir.

—>sda(first SCSI disk drive veya first SCSI disk address-wise) nedir?

Disk türleri sda, sdb, sdc şeklinde değişmektektir. Bizim sistemimizde sadece 1 disk olduğundan dolayı sadece sda olarak gözükmektedir.

SD terimi, SCSI(Small Computer System Interface, Küçük Bilgisayar Sistem Arayüzü) diski anlamına gelir.

Linux altındaki SCSI cihazları, genellikle kullanıcının cihazı tanımlamasına yardımcı olmak için adlandırılır. Örneğin, ilk SCSI CD-ROM'u /dev/scd0'dır. SCSI diskleri /dev/sda, /dev/sdb, /dev/sdc vb. olarak etiketlenir. Aygit başlatma tamamlandığında, Linux SCSI disk sürücüsü arabirimleri (sd) yalnızca SCSI OKUMA ve YAZMA komutları gönderir.

Yani sda, ilk SCSI sabit diski anlamına gelir. Aynı şekilde diskteki bireysel bölüm

isimleri sda1, sda2, vb. olarak alır. Yani disk partition işleminde gerçekleştirdiğimiz işlemler sda1, sda2 ve sda5 isimlendirmelerini almıştır.

sda2 bölümünün neden 1Kb olduğunu ve biz bölümlenmede bunun için yer ayırmamamıza rağmen nedenoluştugu konusuna gelirsek; "sudo fdisk -l" komutu ile sda2'nin Extended(genişletilmiş) bir bölüm olduğunu görmüş oluruz.

/dev/sda2 genişletilmiş bir bölümdür, yani herhangi bir veriyi tutmaz, sadece /dev/sda5 gibi mantıksal birimler için bir "kapsayıcı(container)" görevi görür ve biçimlendirilmez.

Kapsayıcılar, uygulama katmanında kod ve bağımlılıkların derlendiği veya birlikte paketlendiği bir soyutlamadır. Kapsayıcılar Linux'un bir özelliğiidir. Linux kapsayıcıları, uygulamaları tüm çalışma zamanı ortamlarıyla (çalıştırmak için gerekli tüm dosyalar) paketlemenize ve yalıtmانına olanak tanıyan teknolojilerdir. Bu, tam işlevselligi korurken, içeren uygulamayı ortamlar (geliştirme, test, üretim vb.) arasında taşımayı kolaylaştırır.

Linux Containers, tek bir Linux çekirdeği kullanarak bir kontrol ana bilgisayarında birden çok yalıtılmış Linux sistemini çalışırmak için işletim sistemi düzeyinde bir sanallaştırma yöntemidir. Kısa çıktı için "sudo fdisk -l /dev/sda" komutu kullanılır.

```
akaraca@akaraca42:~$ sudo fdisk -l
[sudo] password for akaraca:
Disk /dev/sda: 30.8 GiB, 33071247872 bytes, 64592281 sectors
Disk model: VBOX HARDDISK
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x04dde019

Device      Boot   Start     End   Sectors  Size Id Type
/dev/sda1    *      2048   976895   974848  476M 83 Linux
/dev/sda2          978942 64591871 63612930 30.3G  5 Extended
/dev/sda5          978944 64591871 63612928 30.3G 83 Linux
```

> Extended(genişletilmiş) olması kısaca sda5 bölümü için bir sınır oluşturuyor anlamına geliyor.

Kapsayıcılar, bir geliştiricinin bir uygulamayı kitaplıklar ve diğer bağımlılıklar gibi ihtiyaç duyduğu tüm parçalarla paketlemesine ve hepsini tek bir paket olarak göndermesine olanak tanır.

Konteyner neden gereklidir? Kapsayıcılar, uygulamaların daha hızlı dağıtımasına, yamalanmasına veya ölçeklendirilmesine olanak tanır. Kapsayıcılar, geliştirme, test ve üretim döngülerini hızlandırmak için çevik ve DevOps çabalarını destekler.

<https://embeddedbits.org/introduction-linux-containers/>

—> sda5_crypt neden sonunda crypt vardır?

Disk bölümlerinde kullanıcı dosyaları olarak karar kıldığımiz disk bölümünü şifreleme işlemine dahil ettiğimizden dolayı yazmaktadır. Sunucunun reboot sonrasında sda5 diskine erişmek için sda5_crpyt kısmının şifre istemeside bu yüzdedir.

```
Volume group "LVMGroup" not found
Cannot process volume group LVMGroup
Volume group "LVMGroup" not found
Cannot process volume group LVMGroup
Please unlock disk sda5_crypt: _
```

---> Şifreli LVM nedir?

Şifreli bir LVM bölümü kullanıldığında, şifreleme anahtarı bellekte (RAM) depolanır. Bu anahtarın alınması verilerin şifresinin çözülmesine izin verdiginden, bilgisayarın veya birimin olası hırsızı veya bir bakım teknisyeninin erişebilecegi bu anahtarın bir kopyasının bırakılmasından kaçınılması önerilir.

LVM şifreli bölümleri kullandığınızda, takas bölümünü de şifrelemeniz önerilir.

---> Diskimin şifreli Linux olup olmadığını nasıl anlarım?

Şifreleme durumunu doğrulamanın başka bir yolu da Disk ayarları bölümünde bakmaktadır. Bu durum, disklerin aslında işletim sistemi düzeyinde şifrelendiklerini değil, şifreleme ayarlarının damgalı olduğu anlamına gelir. Tasarım gereği, diskler önce damgalanır ve daha sonra şifrelenir.

--->Disk şifreleme anahtarları nerede saklanır?

Şifreleme anahtarı, anahtar yönetim sunucusunda oluşturulur ve saklanır. Anahtar yöneticisi, kriptografik olarak güvenli bir rastgele bit oluşturucu kullanarak şifreleme anahtarını oluşturur ve anahtarı, tüm özellikleriyle birlikte anahtar depolama veritabanında saklar.

--->Linux şifreleme anahtarlarını nerede saklar?

Çoğu Unix (ve Linux bir istisna değildir) parolalarınızı şifrelemek için öncelikle

DES (Veri Şifreleme Standardı) adı verilen tek yönlü bir şifreleme algoritması kullanır. Bu şifreli parola daha sonra (tipik olarak) /etc/passwd (veya daha az yaygın olarak) /etc/shadow dizininde saklanır.

/etc/shadow olarak da bilinen bir gölge parola dosyası, Linux'ta şifrelenmiş kullanıcı parolalarını saklayan ve yalnızca kök kullanıcı tarafından erişilebilen, yetkisiz kullanıcıların veya kötü niyetli kişilerin sisteme girmesini engelleyen bir sistem dosyasıdır.

"chage" komutu ile güncelleme yapmak yerine bu dosya üzerinden de güncelleme yapabilirsiniz.

<https://www.cyberciti.biz/faq/understanding-etcshadow-file/>

—> Neden isimlendirmede sda1 sda2 şeklinde devam ederken sda5'e geçiş yapmış?

Linux'taki geleneksel DOS bölümleri şu şekilde gözükmektedir;

* 1'den 4'e kadar olan bölümler birincil(Primary) bölümlerdir.

* 5'in üzerindeki bölümler mantıksal(Logical) bölümlerdir.

DOS bölümleme şemasında (bu Linux'a özgü değildir), mantıksal bölümleri kullanmak istiyorsanız, bunlar için birincil bölümlerden birinde bir işaretçi tanımlamanız gereklidir. Bu işaretçide BIOS daha fazla bilgi bulacaktır.

Bu işaretçi (makinenizdeki sda2) fdisk'te id 5 "Genişletilmiş" olarak gösterilir - bölümleme şemasını normalde mümkün olan varsayılan 4 bölümden daha fazlasına genişletir.

Bundan dolayı artık sistemimiz iki bölümden oluşuyor; Bir birincil(primary), önyüklenen bir bölüm: sda1 (bir linux-raid-dizisinin parçasıydı ya da parçasıydı) ve bir mantıksal(logical) bölüm: sda5 (bir linux-raid-dizisinin parçasıydı ya da onun parçası).

Teknik olarak 2 birincil bölüm ve bir mantıksal bölüm içerir.

—> LVM(Logical Volume Manager, Mantıksal birim yönetimi) nedir?

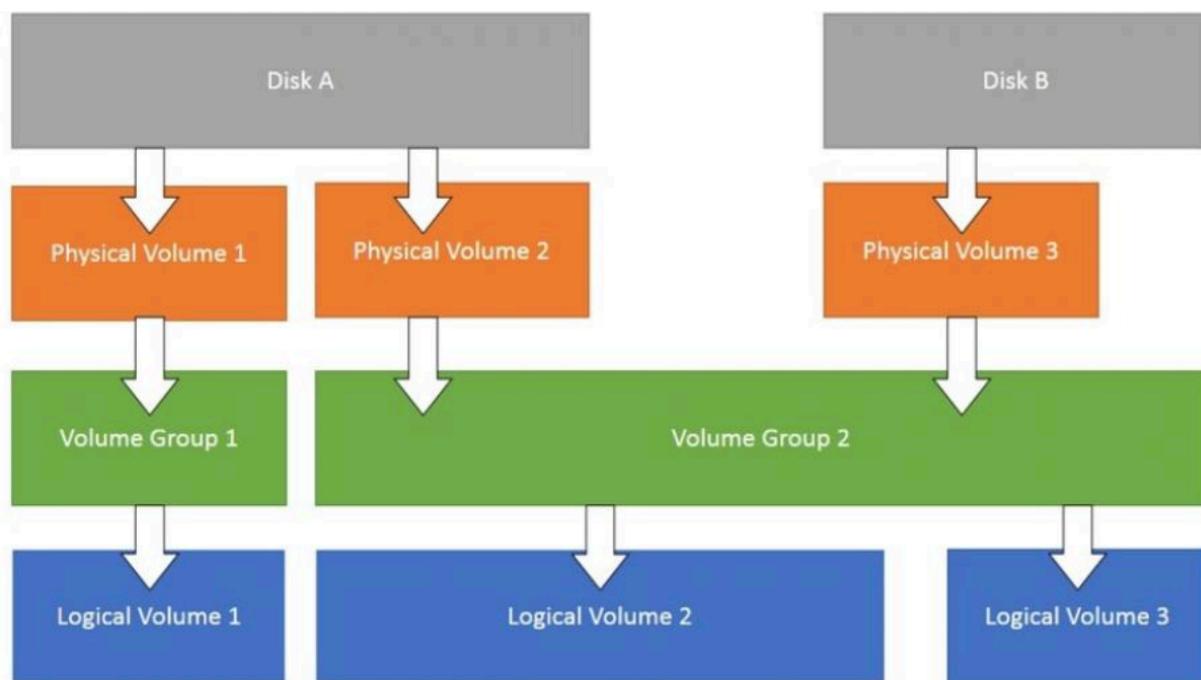
LVM, disklerin ve bölümlerin sıkılıkla taşındığı veya yeniden boyutlandırıldığı dinamik ortamlarda son derece yardımcı olabilir. Normal bölümler de yeniden boyutlandırılabilirken, LVM çok daha esnek ve genişletilmiş işlevsellik sağlar. Olgun bir sistem olarak, LVM de çok kararlıdır ve her Linux dağıtımını varsayılan olarak onu destekler.

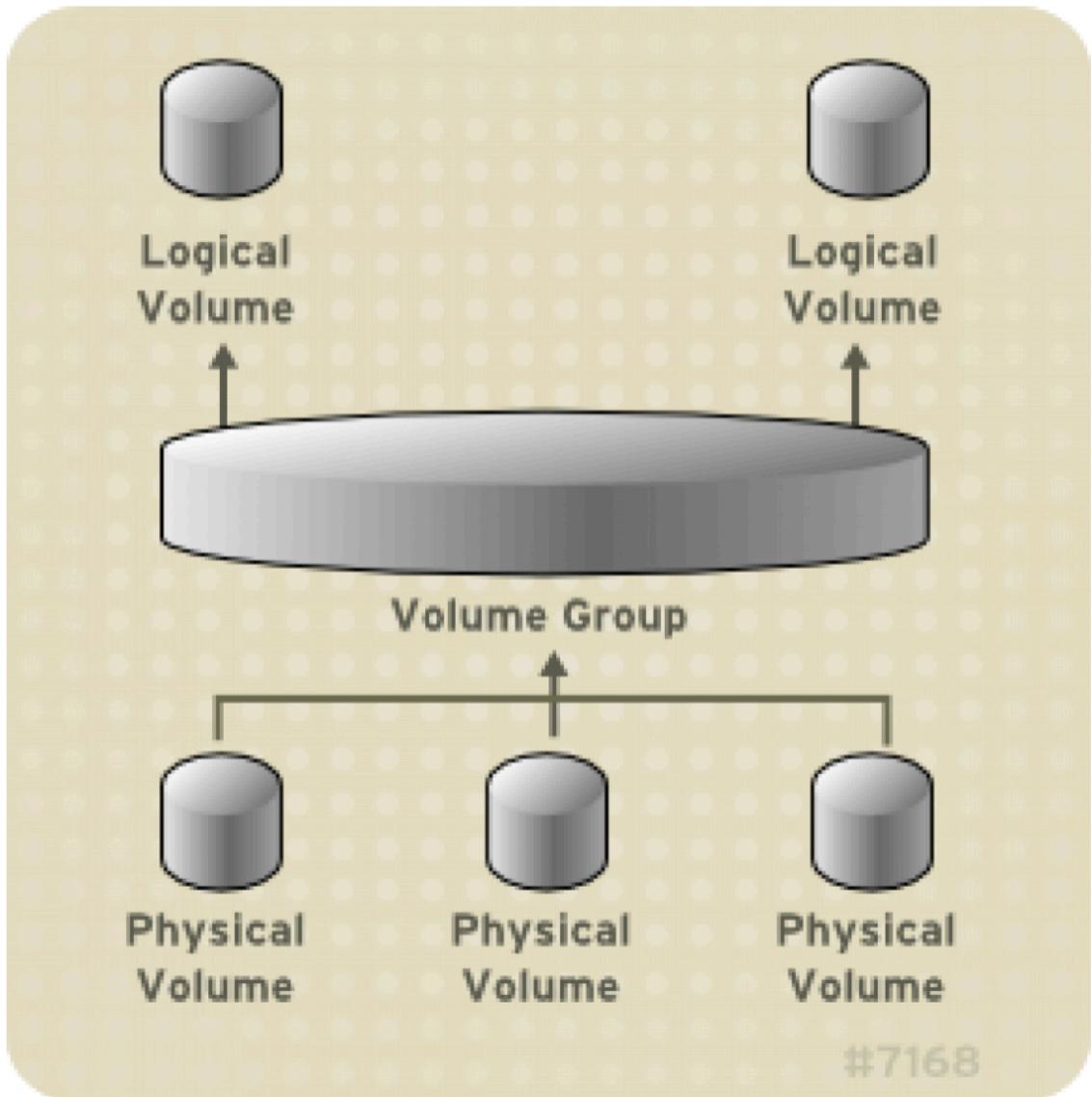
#LVM'nin ana avantajları artan soyutlama, esneklik ve kontroldür. Mantıksal birimler, "veritabanları" veya "kök yedekleme" gibi anlamlı adlara sahip olabilir. Alan gereksinimleri değişikçe ve çalışan bir sistemdeki havuz içindeki fiziksel cihazlar arasında geçiş yaptıkça veya kolayca dışa aktarıldıkça birimler dinamik olarak yeniden boyutlandırılabilir.

—> LVMGroup nedir?

Bir birim grubu(volume Group) (VG), Mantıksal Hacim Yöneticisi(Logical Volume Manager) (LVM) mimarisinin merkezi birimidir. Birleştirilmiş fiziksel cihazların depolama kapasitesine eşit, tek bir depolama yapısı oluşturmak için birden çok fiziksel birimi birleştirdiğimizde yarattığımız şey budur. Kısaca birden fazla disk bulunuyor ise bunu tek bir disk haline getirip dosya dizinlerini tek bir disk üzerinden yönlendiriyor. LVM'nin dezavantajı ise birleşimi gerçekleştirilen disklerden biri arızalanırsa oluşturduğumuz grup içerisindeki veri kaybı yaşarız, bu veri kaydı tüm diskler içerisindeki verilerde olabilir. Kısacası kullanımı oldukça güzel ama bir o kadar da riskli bir yapıdır.

<https://www.tiger-computing.co.uk/linux-tips-list-block-devices/>





#7168

LVM Architecture model

LVMGroup /dev dizini altındadır, birimler ise grubun altında yer almaktadır.

```

akaraca@akaraca42:/dev$ ls
LVMGroup      disk  fd      mem   rtc      stderr  tty16  tty27  tty38  tty49  tty6   uhid     vcsa   vcsu5
autofs        dm-0  full    inqueue  rtc0    stdin   tty17  tty28  tty39  tty5   tty60  uinput   vcsa1  vcsu6
block         dm-1  fuse    net     sda    stdout  tty18  tty29  tty4   tty50  tty61  urandom  vcsa2  vfio
bsg          dm-2  hidraw0 null    sda1   tty     tty19  tty3   tty40  tty51  tty62  vboxguest  vcsa3  vga_arbiter
btrfs-control dm-3  hpet    nvram   sda2   tty0    tty2   tty30  tty41  tty52  tty63  vboxuser   vcsa4  vhci
bus          dm-4  hugepages port    sda5   tty1    tty20  tty31  tty42  tty53  tty7   vcs     vcsa5  vhost-net
cdrom        dm-5  initctl  ppp    sg0    tty10   tty21  tty32  tty43  tty54  tty8   vcs1    vcsa6  vhost-vsock
char         dm-6  input    psaux  sg1    tty11   tty22  tty33  tty44  tty55  tty9   vcs2    vcsu   zero
console      dm-7  kmsg    ptmx   shm    tty12   tty23  tty34  tty45  tty56  tty50  vcs3    vcsu1
core         dri   log     pts    snapshot  tty13   tty24  tty35  tty46  tty57  tty51  vcs4    vcsu2
cpu_dma_latency dvd  loop-control random  snd    tty14   tty25  tty36  tty47  tty58  tty52  vcs5    vcsu3
cuse         fb0  mapper   rfkill sr0    tty15   tty26  tty37  tty48  tty59  tty53  vcs6    vcsu4
akaraca@akaraca42:/dev$ cd LVMGroup/
akaraca@akaraca42:/dev/LVMGroup$ ls
home  root  srv  swap  tmp  var  var-log

```

—> /boot nedir?

Bilgisayar başlatıldığından ilk olarak yüklenmesi gereken programların bulunduğu yerdir.

Boot, bir kaynaktan işletim sistemini çalışmaya, başlatmaya yarayan işlevdir.

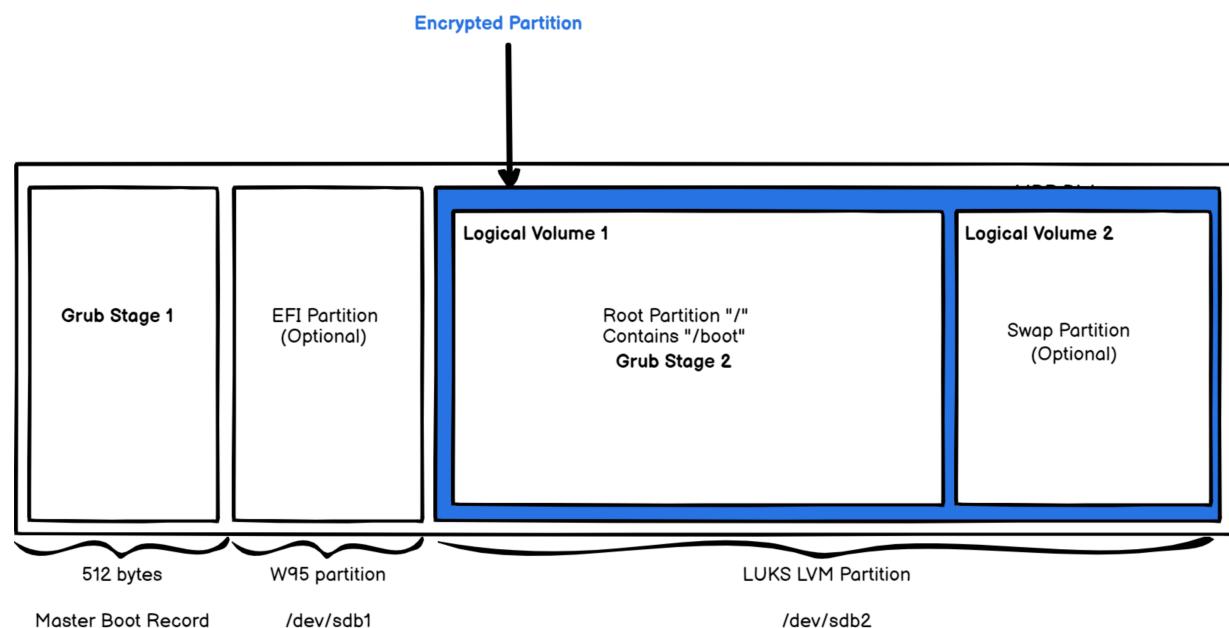
CPU'ya ilk elektrik verildiğinde bilgisayarın çalışabilmesi için uygulamaya konulan işlemler bütünü ve çalışan uygulamalardır. İlk elektrik verildiğinde başlar ve bilgisayar normal görevlerini yapmaya hazır hale geldiğinde sonlanır.

Bu dizin ne işe yarar; Sistemin boot etmesi ve boot için herhangi bir engel çıktığında kurtarmak için bulunan bölüntüdür.

Şifrelenmeyi (crypt altında) ve LVM ile ayrılmayı desteklemiyor bu yüzden disk iki bölüme ayııyoruz. Biri, ana dosya (şifreli) kurtarma için yedek dizin ve diğer (/boot) start vermek için kullanılıyor.

Önyükleme bölümü şifrelenebilir mi? Şifrelenenler, işletim sistemi bölümü ve Linux çekirdeğini ve ilk RAM diskini içeren önyükleyici ikinci aşama dosya sistemidir. ... Her iki durumda da, birinci aşama GRUB önyükleyici dosyaları, BIOS önyükleme modunda kriptografik aracılığıyla şifrelenemez (ve korunamaz).

MBR Disk Design



—> root (/), diğer adıyla kök dizin nedir?

Kök dosya sistemi, dosya sisteminin en üst düzey dizinidir. Diğer dosya sistemleri monte edilmeden önce Linux sistemini başlatmak için gereken tüm dosyaları içermelidir. Kalan dosya sistemlerini başlatmak için gereken tüm yürütülebilir dosyaları ve kitaplıklarını içermelidir.

En üst dizindir.

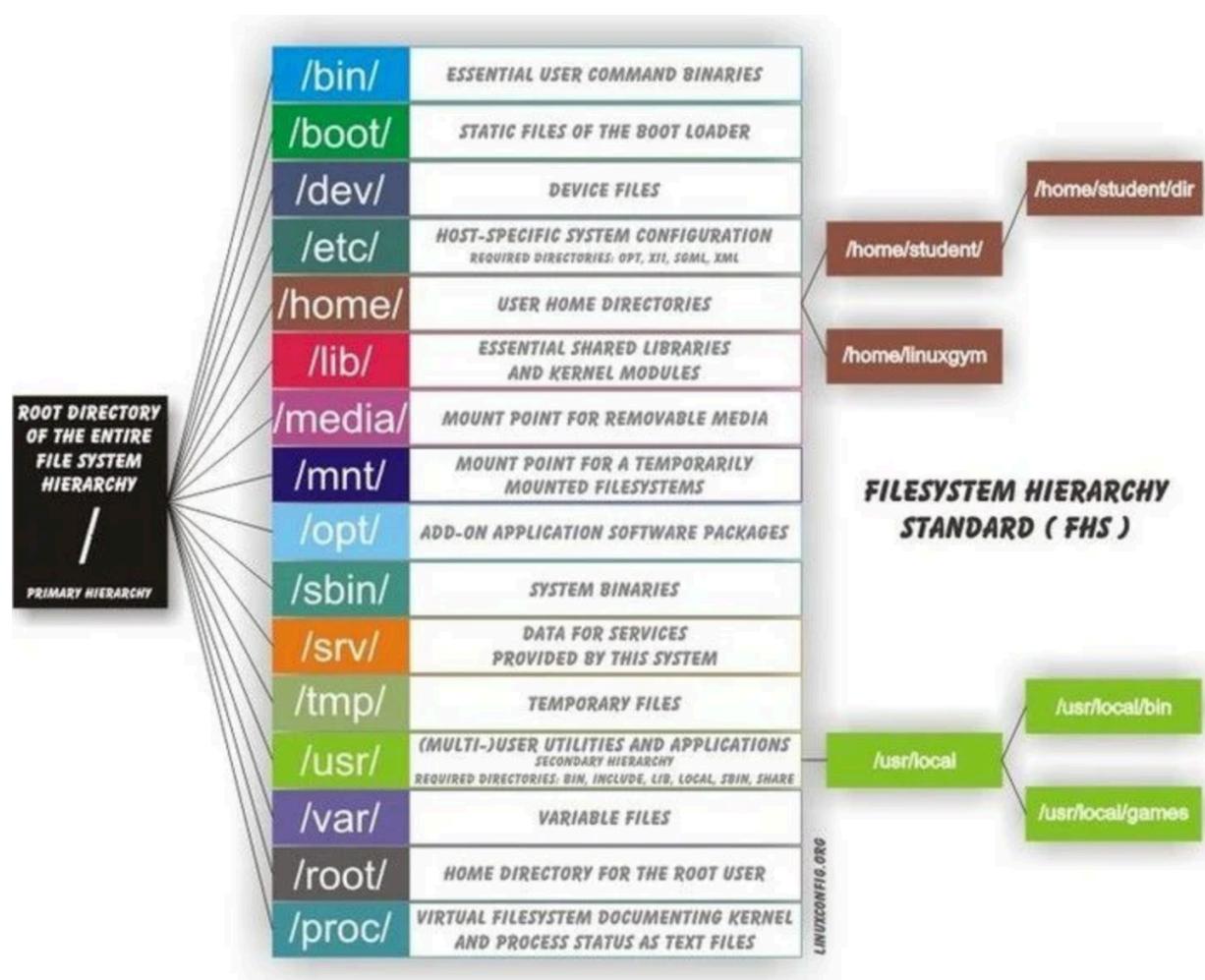
Bir modüler disk veri kümesidir. LVM ve crypt altında şifrelenerek korunmaktadır.

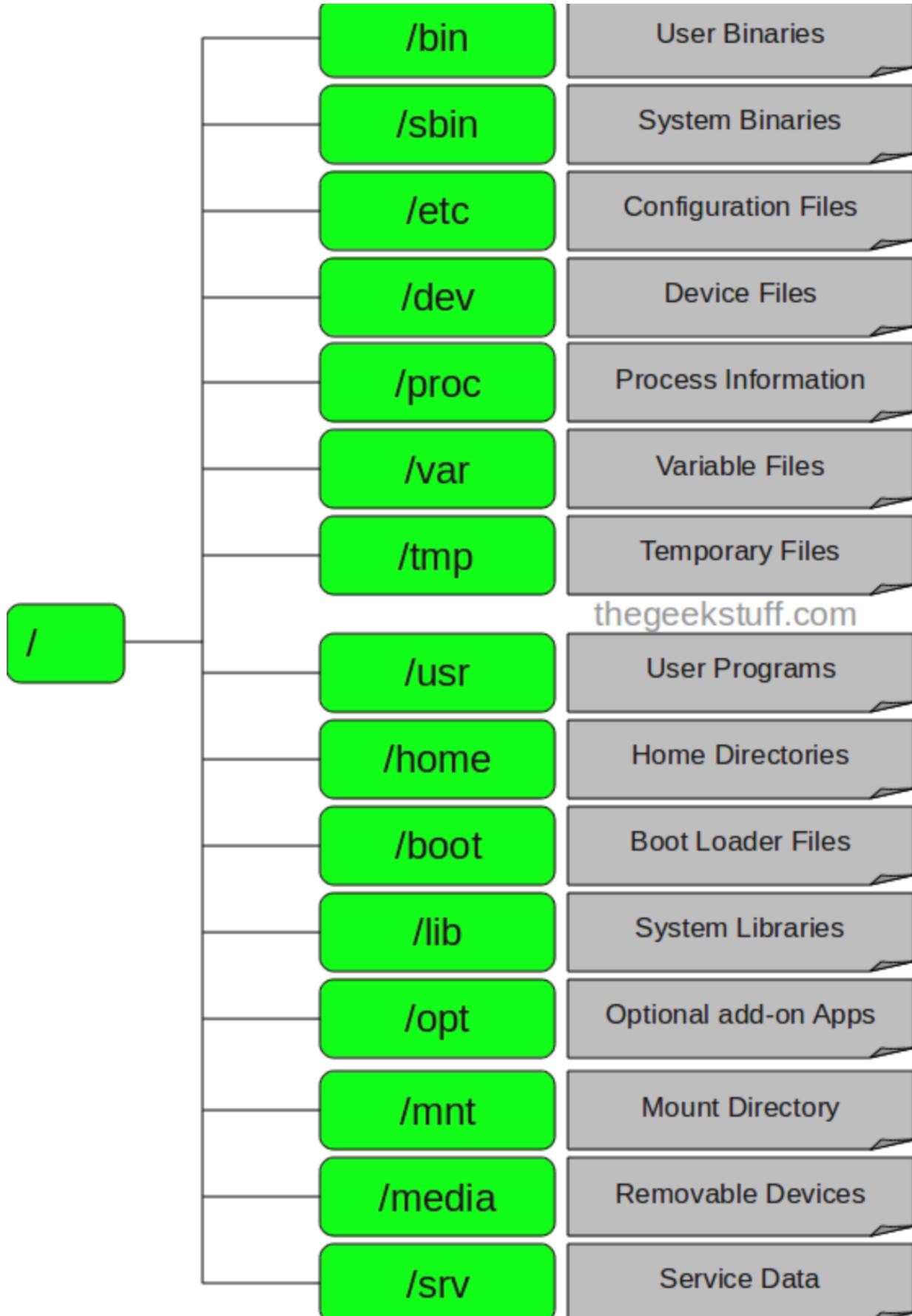
Bilgisayar dosya dizgelerinde hiyerarşideki en üstteki veya ilk gelen dizine denir.

Bir ağacın kökü gibi düşünülebilir.

Linux - Unix sistemlerde, işletim sistemine her türlü müdahale bulunabilme yetkisine sahip, "root" adıyla tanımlanmış, süper yetkili(super user, su) özel bir kullanıcı hesabı vardır. /root dizini , bu özel kullanıcı hesabının ev dizinidir.

Önyükleme esnasında hafızaya yüklenen ilk bölümdür.





—> swap ([SWAP])nedir?

Swap memory (takas belleği), bir VPS veya Bulut Sunucusu ortamında açıkça çok önemlidir. Özellikle düşük belleğe (RAM) sahip bir sanal özel sunucunuz varsa, örneğin 1024MB. Ancak, yalnızca VPS ve bulut sunucuları için değil, tahsis edilmiş sunucular için takas belleği de aynı derecede önemlidir. Takas belleğinin rolü çok önemli hale gelir ve bir sunucuya RAM'i bittiğinde yardımcı olabilir.

Tipik bir bilgisayarda iki temel bellek türü vardır. Birinci tip, rastgele erişimli bellek (RAM), bilgisayar tarafından aktif olarak kullanılırken veri ve programları depolamak için kullanılır. Programlar ve veriler, RAM'de depolanmadıkça bilgisayar tarafından kullanılamaz. RAM geçici bellektir; yani, bilgisayar kapatılırsa RAM'de depolanan veriler kaybolur.

Sabit sürücüler, verilerin ve programların uzun süreli depolanması için kullanılan manyetik ortamlardır. Manyetik ortam uçucu değildir; bir diskte depolanan veriler, bilgisayardan güç kesildiğinde bile kalır. CPU (merkezi işlem birimi), sabit sürücüdeki programlara ve verilere doğrudan erişemez; önce RAM'e kopyalanması gereklidir ve bu, CPU'nun programlama talimatlarına ve bu talimatlar tarafından çalıştırılacak verilere erişebileceği yerdır. Önyükleme işlemi sırasında, bir bilgisayar, çekirdek ve init veya systemd gibi belirli işletim sistemi programlarını ve sabit sürücüdeki verileri, doğrudan bilgisayarın işlemcisi olan CPU tarafından erişilen RAM'e kopyalar.

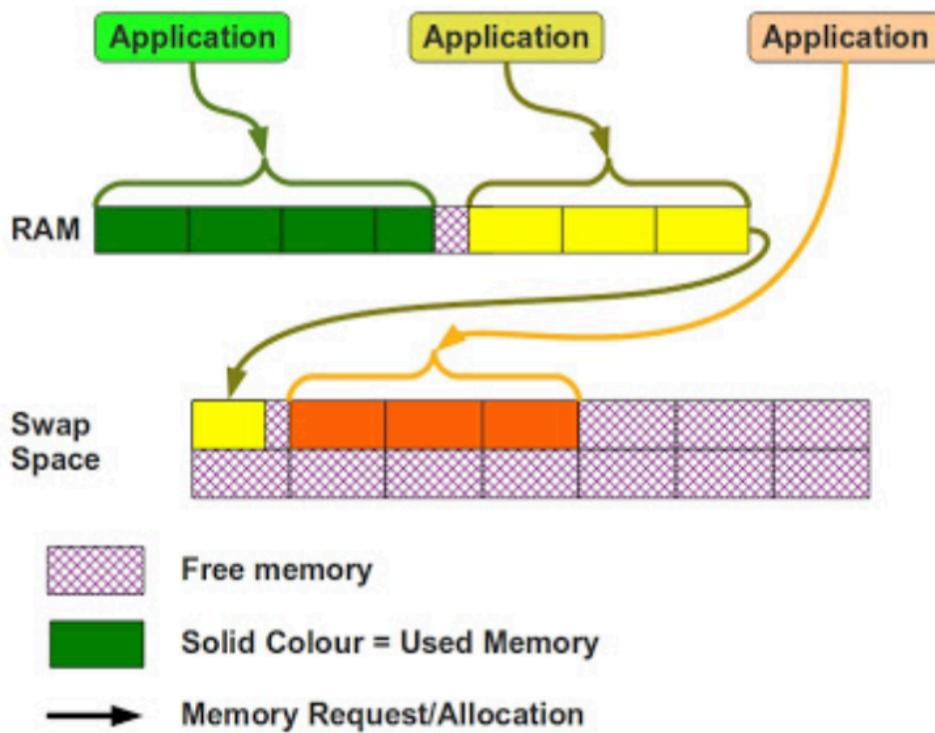
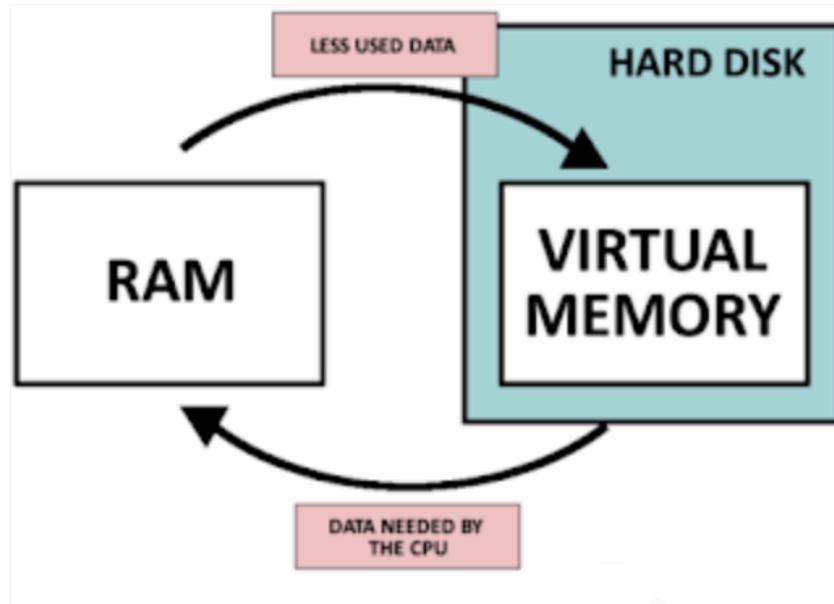
Modern Linux sistemlerindeki ikinci bellek türü takas alanıdır.

Takas alanının birincil işlevi, gerçek RAM dolduğunda ve daha fazla alana ihtiyaç duyulduğunda RAM belleği için disk alanını değiştirmektir.

Örneğin, 8 GB RAM'e sahip bir bilgisayar sisteminiz olduğunu varsayılmı. Bu RAM'i doldurmayan programları başlatırsanız, her şey yolundadır ve takas gerekmeyecektir. Ancak, üzerinde çalıştığınız elektronik tablonun daha fazla satır eklediğinizde büyüğünü ve bunun ve çalışan diğer her şeyin şimdi tüm RAM'i doldurduğunu varsayılmı. Kullanılabilir takas alanı olmadan, diğer programları kapatarak sınırlı RAM'inizin bir kısmını boşaltana kadar elektronik tablo üzerinde çalışmaya bırakmanız gereklidir.

Çekirdek, içeriğin yakın zamanda kullanılmadığı bellek sayfalarını, yani blokları algılayan bir bellek yönetim programı kullanır. Bellek yönetim programı, nispeten seyrek olarak kullanılan bu bellek sayfalarından yeteri kadarını sabit sürücüde "paging(sayfalama)" veya takas için özel olarak belirlenmiş özel bir bölüme değiştirir. Bu, RAM'de yer açar ve elektronik tablonuza daha fazla veri girilmesi için yer açar. Sabit sürücüye aktarılan bu bellek sayfaları, çekirdeğin bellek yönetim kodu tarafından izlenir ve gerektiğinde RAM'e geri çağrılabılır.

Bir Linux bilgisayarındaki toplam bellek miktarı, RAM artı takas alanıdır ve sanal bellek olarak adlandırılır.



--> home (/home) nedir?

Ev dizini, bir ağda veya Unix veya Linux türevi işletim sisteminde bir kullanıcıya yaygın olarak verilen dizin veya klasördür. Giriş dizini ile kullanıcı tüm kişisel bilgilerini, dosyalarını, oturum açma komut dosyalarını ve kullanıcı bilgilerini

saklayabilir.

Linux'ta ev dizini nerede? Ana dizininize gitmek için "cd" veya "cd ~" kullanın Bir dizin düzeyinde yukarı gitmek için "cd .." kullanın Bir önceki dizine (veya geri) gitmek için "cd -" kullanın.

---> var (/var) (variable, değişken) nedir?

/var, Linux ve diğer Unix benzeri işletim sistemlerinde, sistemin çalışması sırasında veri yazdığı dosyaları içeren kök dizinin standart bir alt dizinidir.

var local ne için kullanılır? "/var" genellikle günlük dosyaları, 'geçici' dosyalar (posta biriktirme, yazıcı biriktirme vb.), veritabanları ve belirli bir kullanıcıya bağlı olmayan diğer tüm veriler için kullanılır.

#Bu dizin, biriktirme(spool, daha sonra işlenmeyi bekleyen veriler) ve günlük dosyaları gibi boyutu değiştirebilen dosyaları içerir.

/var/tmp, /tmp gibi, bu dizin de belirsiz bir süre boyunca saklanan geçici dosyaları tutar.

---> var-log (/var/log) nedir?

/var/log/messages - Bu dosya, sistem başlangıcında günlüğe kaydedilen mesajlar da dahil olmak üzere, içinde bulunan tüm genel sistem mesajlarını içerir. Sistem günlüğü yapılandırma dosyasının nasıl gönderildiğine bağlı olarak, posta, cron, arka plan programı, çekirdek, yetkilendirme vb. dahil olmak üzere bu dosyada günlüğe kaydedilen birkaç şey vardır.

var günlük dosyalarını kaldırabilirim miyim? Sisteminiz bir test sistemi ise veya günlükte ne olduğunu gerçekten umursamıyorsanız, günlüğü temizleyebilirsiniz. Ancak herhangi bir uygulamanız hata veriyorsa, tam açıklama bulabileceğiniz tek yer günlüklerdir. Günlüklerin hiçbirinin sizin için yararlı olmadığından eminseniz, bunları her zaman temizleyebilirsiniz.

---> srv (/srv) (service, hizmetler) nedir?

Web sunucuları için veriler ve komut dosyaları, FTP sunucuları tarafından sunulan veriler ve sürüm kontrol sistemleri için depolar gibi bu sistem tarafından sunulan siteye özgü veriler. srv, hizmet anlamına gelir. Sunucuya özel hizmetlerle ilgili verileri içerir. Örnek, /srv/cvs, CVS ile ilgili verileri içerir.

Bu, sistem tarafından sağlanan hizmetler için veri olan sunucu verileridir.

Bu dizin, bu sistem tarafından sunulan siteye özel verileri içerir.

---> tmp (/tmp) (temp, temporary files, geçici dosyalar) nedir?

Bu dizin çoğunlukla geçici olarak gerekli olan dosyaları içerir. Birçok program bunu kilit dosyaları oluşturmak ve verilerin geçici olarak depolanması için kullanır. Ne yaptığınızı tam olarak bilmeyorsanız bu dizinden dosyaları kaldırın! Bu dosyaların çoğu, şu anda çalışan programlar için önemlidir ve bunların silinmesi sistemin çökmesine neden olabilir. Genellikle zaten birkaç KB'den fazlasını içermez. Çoğu sistemde, bu dizin, yerel sistem tarafından önyükleme veya kapatma sırasında temizlenir.

Birçok program bu /tmp dizinini geçici veri yazmak için kullanır ve genellikle artık gerekmeliğinde verileri kaldırır. Aksi takdirde, sunucu yeniden başlatıldığında /tmp dizini temizlenir.

—> rom nedir?

#ROM (read-only memory, salt okunur bellek), kalıcı bir bellek türüdür. Bu, verileri aldığı ve bir çip üzerine kalıcı olarak yazdığı ve bilgisayarınızı kapattıktan sonra bile sürdüğü anlamına gelir.

Açıklaması Sadece okunabilir bellek. ROM, bilgisayarlarda ve diğer elektronik aletlerde kullanılan bir depolama birimidir. RAM gibi yazılıp silinebilen bir depolama birimi değildir. ROM içeriği sadece üretim anında yazılır. Kullanıcının kendi isteği doğrultusunda programlanamaz.

—>sr0 nedir?

Burada anlaşılması gereken gerçek şu ki /dev/sr0 scsi controller (hypervisor) üzerinde bulunan bir cihazdır. /dev/sr0 ayrıca bir DVD- /CD-ROM veya benzeri olabilir. ???Oradaki medya salt okunur olduğu için her zaman %100 kullanılır.???

<https://linuxhint.com/linux-lsblk-command-tutorial-for-beginners/>

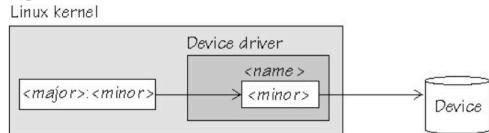
CD-ROM: Esas olarak sr0 ile gösterilirler ve RM değeri 1'dir.

sr0	11:0	1	1024M	0	rom
-----	------	---	-------	---	-----

---> MAJ:MIN (major and minor device number) nedir?

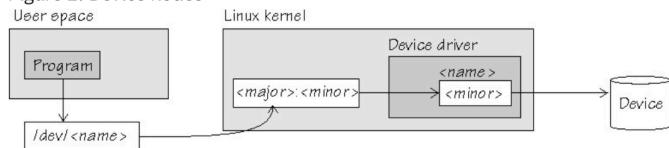
Bunlar çekirdek tarafından aygıtları dahili olarak tanımlamak için kullanılan sayılardır, ilk sayı aygit türünü belirtir (örneğin 8, SCSI diskleri için kullanılır).

Figure 1. Minor numbers and device names



User space programs access character and block devices through *device nodes* also referred to as *device special files*. When a device node is created, it is associated with a major and minor number.

Figure 2. Device nodes



<https://www.ibm.com/docs/en/linux-on-systems?topic=hdaa-names-nodes-numbers>

https://wiki.st.com/stm32mpu/wiki/How_to_find_Linux_kernel_driver_associated_to_a_device

Major number 254 ve minor number 0 olan bir karakter aygıtı (c) oluşturur. Küçük sayılar 0 ila 255 aralığında olmalıdır çünkü tarihsel nedenlerden dolayı bazen tek bir bayt içinde depolanırlar.

Her aygit dosyasının bir ana(major) kimlik numarası ve bir ikincil(minor) kimlik numarası vardır. Ana kimlik, genel aygit sınıfını tanımlar ve çekirdek tarafından bu tür aygit için uygun sürücüyü aramak için kullanılır. Küçük kimlik, genel bir sınıf içindeki belirli bir cihazı benzersiz şekilde tanımlar. Bir aygit dosyasının ana ve küçük kimlikleri ls -l komutuyla görüntülenir. Her aygit sürücüsü, ilişkisini belirli bir ana aygit kimliğiyle kaydeder ve bu ilişkilendirme, aygit özel dosyası ile aygit arasındaki bağlantıyı sağlar. Çekirdek, aygit sürücüsünü aradığında aygit dosyasının adının hiçbir ilgisi yoktur.

```

# Major numbers;
Character devices:
 1 mem
 2 pty
 3 ttyp
 4 /dev/vc/0
 4 tty
 5 /dev/tty
 5 /dev/console
 5 /dev/ptmx
 5 ttyRPMMSG
 7 vcs
10 misc
13 input
21 sg
29 fb
81 video4linux
89 i2c
90 mtd
116 alsa
128 ptm
136 pts
153 spi
166 ttyACM
180 usb
189 usb_device
199 galcore
226 drm
245 cec
246 media
247 ttySTM
248 bsg
249 watchdog
250 iio
251 ptp
252 pps
253 rtc
254 gpiochip

```

Block devices:
1 ramdisk
7 loop
8 sd
11 sr
31 mtdblock
65 sd
66 sd
67 sd
68 sd
69 sd
70 sd
71 sd
128 sd
129 sd
130 sd
131 sd
132 sd
133 sd
134 sd
135 sd
179 mmc
254 virtblk
259 blkext

---> RM (removable, çıkarılabilir) nedir?

Bu sütun, cihazın çıkarılabilir olup olmadığını gösterir. sr0 cihazının RM değerlerinin 1'e eşit ve bunların çıkarılabilir olduğu anlamına gelir.

---> RO (read-only, salt okunur) nedir?

Bu sütun, bir cihazın salt okunur durumunu gösterir. 1 aygıtın salt okunur

olduğunu ve 0 salt okunur olmadığını belirtir.

—> SSH (secure shield) yani güvenli kabuk nedir?

SSH (Secure Shell), iki sistem arasında güvenli uzak bağlantılar sağlayan bir ağ protokolüdür. Sistem yöneticileri, makineleri yönetmek, dosyaları kopyalamak veya sistemler arasında taşımak için SSH yardımcı programlarını kullanır. SSH, verileri şifreli kanallar üzerinden ilettiği için güvenlik üst düzeydedir.

Bağlantı noktası(port), Güvenli Kabuk (SSH) iletişimini için kullanılır ve VM(sanal makine)'ye uzaktan yönetim erişimine izin verir. Genel olarak trafik, parola doğrulaması kullanılarak şifrelenir.

SSH istemci bağlantıları için varsayılan bağlantı noktası 22'dir.

—> UFW (Uncomplicated Firewall) yani Karmaşık olmayan Güvenlik Duvarı nedir?

Kullanımı kolay olacak şekilde tasarlanmış bir netfilter(internet filtresi) güvenlik duvarını yönetmek için kullanılan bir programdır. Az sayıda basit komuttan oluşan bir komut satırı arabirimini kullanır ve yapılandırma için iptables kullanır.

iptables, Linux için bir güvenlik duvarı programıdır. Tabloları kullanarak sunucunuza gelen ve sunucunuza gelen trafiği izleyecektir. Bu tablolar, gelen ve giden veri paketlerini filtreleyecek, zincir adı verilen kural kümelerini içerir.

—> sudo (super user do), nedir?

Yönetici izinlerine sahip olmayan bir kullanıcıyı, root(kök) kullanıcısı gibi sınırlı yetkiler içerisinde özgür bırakmak için sudo argümanı kullanılır. Bu argümanın erişilebilir olması için öncelikle root ile kullanıcıya bu komutu kullanma izni verilmelidir.

Sudo komutunun diğer bir işlevi ise terminal üzerinde kullandığımız komut satırlarının komut olduğunu belirtmek için kullanılır. Örn: "sudo visudo" komutunu girdiğim zaman visudo'yu bir komut olarak algılar. Lakin sadece "visudo" şeklinde komut girersem bunu komut olarak algılamıyor.

```
root@akaraca42:/home/akaraca# visudo  
bash: visudo: command not found
```

Sudo komutu, programları başka bir kullanıcının güvenlik ayrıcalıklarıyla (varsayılan olarak süper kullanıcı olarak) çalıştırmanıza izin verir. Sizden kişisel parolanızı ister ve sistem yöneticisinin yapılandırdığı sudoers adlı bir dosyayı kontrol ederek bir komut yürütme isteğiniz onaylar.

—> hostname nedir?

Sunucu adı, bilgisayar ağında bağlı cihazlara atanmış bir etikettir ve bu cihazı dünya çapında ağ gibi çeşitli elektronik iletişim formlarında tanımlamak için kullanılır. Sunucu adları, tek bir sözcük veya deyimden oluşan basit adlardan oluşturulabilir.

```
root@akaraca42:/home/akaraca# hostname  
akaraca42
```

—> Virtual Machine (VM) yani Sanal makine nedir?

Programları çalıştırma ve dağıtmak için fiziksel bir bilgisayar yerine yazılım kullanan bir bilgi işlem kaynağıdır. uygulamalar. Her sanal makine, kendi işletim sistemini çalıştırır ve diğer VM'lerden ayrı olarak çalışır. aynı ana bilgisayarda çalışıyor. Örneğin, fiziksel bir PC'de sanal bir MacOS makinesi çalıştırabilirsiniz.

—> Sanal makine nasıl çalışır?

Sanal makineler, sanallaştırma teknolojisi ile mümkün kılınmaktadır. Sanallaştırma, birden çok VM'nin tek bir makinede çalışmasına izin veren sanal donanımı simüle etmek için yazılım kullanır. Fiziksel makine ana bilgisayar olarak bilinirken, üzerinde çalışan VM'lere misafir denir.

—> Neden sanal makine kullanırız?

VM'ler, farklı seviyelerde işlem gücü ihtiyaçlarını karşılamak, farklı bir işlem gücü gerektiren yazılımı çalıştırma için kullanılabilir. İşletim sistemi veya güvenli, korumalı bir ortamda uygulamaları test etmek için cm'ler tercih edilir.

—> Debian ve CentOS işletim sistemleri arasında farklar?

CentOS, Debian'in yaptığı kadar çok mimariyi desteklemez.

CentOS, destek sağlayan daha büyük bir topluluğa sahiptir ve dolayısıyla Red Hat Limited tarafından sağlanan ek destek sayesinde Debian'dan daha kararlıdır.

Hem Debian hem de CentOS düzenli olarak daha yeni sürümlere yükseltilir, ancak fark yaşam döngülerindedir. CentOS, düzenli büyük güncellemeleri desteklemez, ancak zaman zaman bazı küçük ince ayarlar ve hata düzeltmeleri olabilir. CentOS sürümleri yaklaşık on yıl sürer ve daha yeni sürümle geçiş sorunsuz

değildir. Buna karşılık Debian, her 2 ila 3 yılda bir düzenli sürüm güncellemelerini destekler ve Debian'ın geçiş süreci de çok sorunsuz ve çoğunlukla hatasızdır.

Destek açısından, hem Debian hem de CentOS ve her ikisi de açık kaynaklı Linux çekirdeğine dayandığından topluluk tarafından desteklenir. Ancak CentOS kullanıcıları, kilitlenme ve hata raporlarını, bunları küçük bir yukarı akış sürümünde değiştirmek için Red Hat'e gönderebilir.

Paket yönetimi için, paket formatı olarak RPM(Red Hat Package Manager) ve paket yöneticisi olarak YUM/DNF, CentOS tarafından desteklenirken, Debian, paket formatı olarak DEB(Debian Software Package file)'ye ve paket yöneticisi olarak dpkg/APT'ye sahiptir.

**** DNF: sudo yum install sudo. // yum, Yellowdog updater modifier

**** APT: sudo apt install sudo. // apt, Advanced(gelişmiş) Packaging Tool // dpkg, Debian Package

İşletim sistemine göre özel komut olarak geçmektedir.

Operating System	Format	Tool(s)
Debian	.deb	apt , apt-cache , apt-get , dpkg
Ubuntu	.deb	apt , apt-cache , apt-get , dpkg
CentOS	.rpm	yum
Fedora	.rpm	dnf

#apt komutunun farklılıklarını şu şekildedir:

apt command	the command it replaces	function of the command
apt install	apt-get install	Installs a package
apt remove	apt-get remove	Removes a package
apt purge	apt-get purge	Removes package with configuration
apt update	apt-get update	Refreshes repository index
apt upgrade	apt-get upgrade	Upgrades all upgradable packages
apt autoremove	apt-get autoremove	Removes unwanted packages
apt full-upgrade	apt-get dist-upgrade	Upgrades packages with auto-handling of dependencies
apt search	apt-cache search	Searches for the program
apt show	apt-cache show	Shows package details

- **list:** which is similar to `dpkg list` and can be used with flags like `--installed` or `--upgradable`.
- **search:** works just like `apt-cache search` but sorted alphabetically.
- **show:** works like `apt-cache show` but hide some details that people are less likely to care about (like the hashes). The full record is still available via `apt-cache show` of course.
- **update:** like the regular `apt-get update` with color output enabled, but `apt update` also shows the number of upgradeable packages (if any).
- **install,remove:** adds progress output during the dpkg run.
- **upgrade:** the same as `apt-get upgrade --with-new-pkgs .*`
- **full-upgrade:** a more meaningful name for `dist-upgrade`.
- **edit-sources:** edit `sources.list` using `$EDITOR`.
- **policy:** works just like `apt-cache policy`

Yukarıdaki tabloda, apt upgrade komutu dışında apt-get'i apt ile değiştirirseniz

tüm komutlar aynıdır. Eski apt-get upgrade komutu, sisteminizde mevcut olan tüm paketleri günceller. Sisteminizde var olan paketi yüklemez veya kaldırırmaz. Ancak, yeni apt upgrade komutu, yükseltilebilir paketlerin bağımlılıkları olarak eklenen paketleri yükler. apt-get upgrade'e benzemesine rağmen, daha önce kurulmuş olan paketleri de kaldırırmaz.

apt-get, Paketleri sisteminizden kurmak, güncellemek, listelemek ve kaldırmak için bu aracı kullanırsınız.

apt-get, APT (Gelişmiş Paket Aracı) paket yönetim sistemi ile etkileşim kurmak için kullanılan bir komut satırı programıdır.

apt aracı, apt-get ve apt-cache işlevlerini birleştirir.

Ek çıktı ve geliştirilmiş tasarım

genel olarak bir alışkanlık meselesidir. Tercihe göre kullanım vardır.

—> aptitude ve apt arasındaki farklar nelerdir?

Aptitude APT'ın kullanıcı arabirimidir. Yazılım paketlerini listelemeye, onları seçip kurmaya ve kaldırırmaya yarar. APT debian tabanlı sistemlerin paket yöneticisidir APT ile yazılım kurma, yazılım kaldırma, sistemi güncelleme, çekirdeği derleme gibi işlemleri terminal üzerinden gerçekleştirebilirsiniz.

APT'nin alt düzey bir paket yöneticisi ve Aptitude'un üst düzey bir paket yöneticisi olmasıdır. Bu, APT'nin diğer üst düzey paket yöneticilerinde kullanılabileceği anlamına gelir.

Diğer bir büyük fark, her iki aracın da sunduğu işlevselliktir. Aptitude, apt-get'e kıyasla daha iyi işlevsellik sunar. Aslında, apt-get, apt-mark ve apt-cache işlevlerini içerir.

Örneğin, apt-get paket yükseltme, yükleme, bağımlılıkları çözme, sistem yükseltme yükseltme vb. için etkin bir şekilde kullanılabilir. Ancak Aptitude, apt-cache ve apt-mark işlevlerinin dahil edilmesi sayesinde daha fazla özellik sunar. Bu, Aptitude'un paket arama, paket kurulumunu otomasyon veya manuel olarak ayarlama ve paketler üzerinde daha iyileştirilmiş eylemler dahil olmak üzere daha fazla işlevsellik için kullanılabileceği anlamına gelir.

Yüklü paketleri ve kullanılmayan paketleri kaldırmak istiyorsanız, Aptitude kullanmanız gereklidir. apt-get durumunda, “-auto-remove” seçeneğini kullanarak kullanılmayan paketlerden kurtulmak istediğiniz açıkça belirtmeniz gereklidir.

Eylemlerin davranışları hakkında daha fazla bilgi edinmek istiyorsanız, o zaman Aptitude, bilgiye daha iyi erişim için neden ve neden olmasın komutlarını sunduğu

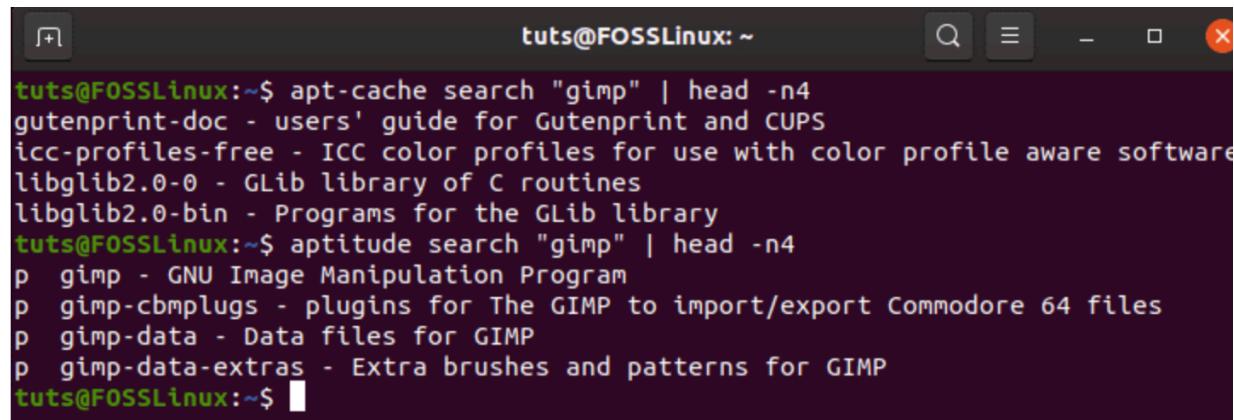
icin arkadaşinizdir. Bu nedenle, kafanız karıştıysa ve belirli bir pakete veya bağımlılıklara ihtiyacınız olup olmadığını bilmek istiyorsanız, Aptitude, yüklenmesi gereken paketler hakkında önerilerde bulunarak sizi bilgilendirebilir.

```
tuts@FOSSLinux:~$ aptitude why gimp
i  ubuntu-desktop-minimal Recommends cups-filters
i A cups-filters           Suggests  imagemagick
p  imagemagick              Depends   imagemagick-6.q16 (>= 8:6.9.2.10+dfsg-2~)
p  imagemagick-6.q16        Suggests  gimp
tuts@FOSSLinux:~$
```

Yukarıdaki örnekte, gimp hakkında bilgi edinmek için aptitude kullandık. Bize tavsiyeler ve öneriler gösterdi. Ayrıca, bize bazı paketlere bağlılığı da gösterdi. Bu şekilde, GIMP'yi kurmayı seçerseniz neyin kurulacağını tamamen bilebilirsiniz.

Paketleri kurarken çakışmaların olması yaygındır. apt-get durumunda, bir çakışma olursa, sorunu çözmez. Temelde bir hata verir ve paket kaldırma veya paket yükleme çakışması sırasında çalışmayı durdurur. Buna karşılık, Aptitude, güçlü arama özelliğiyle paketler üzerinde çalışma olanağı sunarak, tüm depoyu sınırlama olmaksızın incelemenize olanak tanır. Bununla birlikte, bir paketi aramak için apt-cache'i kullanabilirsiniz, ancak bu, apt'nin farklı bir çeşididir.

Gördüğünüz gibi, her iki yöntem de gereklirse paketi aramanıza izin verebilir. Ancak, ikisi arasındaki temel fark şudur. Yetenek, p bayrağını kullanarak her paketin paket kurulum durumunu gösterir; bu, paketin mevcut olduğu ancak kurulu olmadığı anlamına gelir. Paket kuruluysa, i bayrağını gösterecektir.



```
tuts@FOSSLinux:~$ apt-cache search "gimp" | head -n4
gutenprint-doc - users' guide for Gutenprint and CUPS
icc-profiles-free - ICC color profiles for use with color profile aware software
libglib2.0-0 - GLib library of C routines
libglib2.0-bin - Programs for the GLib library
tuts@FOSSLinux:~$ aptitude search "gimp" | head -n4
p  gimp - GNU Image Manipulation Program
p  gimp-cbmplugs - plugins for The GIMP to import/export Commodore 64 files
p  gimp-data - Data files for GIMP
p  gimp-data-extras - Extra brushes and patterns for GIMP
tuts@FOSSLinux:~$
```

aptitude kurulu olarak bulunmamaktadır, kurulması gerekmektedir. "apt-get install aptitude"

<https://www.fosslinux.com/43884/apt-vs-aptitude.htm>

—> APParmor nedir?

AppArmor ("Uygulama Zırhı"), sistem yöneticisinin programların özelliklerini program başına profillerle kısıtlamasına izin veren bir Linux çekirdeği güvenlik

modülüdür. Profiller, ağ erişimi, ham soket erişimi ve eşleşen yollardaki dosyaları okuma, yazma veya yürütme izni gibi yeteneklere izin verebilir.

AppArmor, savunmasız süreçleri kilitler, bu süreçlerdeki güvenlik açıklarının neden olabileceği hasarı sınırlar.

AppArmor'un sloganı, her şeye izin vermek ve ardından yavaş yavaş sıkılaştırılmaktır.

—> hostnamectl komutu nedir? (Hostname-control)

"hostnamectl komutu, Linux sistemi ana bilgisayar adını kontrol etmek ve ilgili ayarlarını değiştirmek için kullanılan uygun bir API sağlar. Komut, belirli bir sistemde /etc/hostname dosyasını gerçekten bulup düzenlemeden ana bilgisayar adını değiştirmeye de yardımcı olur.

değiştirmek için; "hostnamectl set-hostname <new_hostname>" komutu kullanılır.

```
akaraca@akaraca42:/home$ hostnamectl
  Static hostname: akaraca42
    Icon name: computer-vm
      Chassis: vm
    Machine ID: 7c47b500d01a4fccb86cac12b9ef96f7
        Boot ID: 3a0be86fde8e4a5a8c5b468a76557986
  Virtualization: oracle
Operating System: Debian GNU/Linux 11 (bullseye)
      Kernel: Linux 5.10.0-12-amd64
  Architecture: x86-64
```

—> cron nedir?

Cron, adı Yunanca zaman kelimesi olan Chronos'tan gelen bir saat arka plan programıdır(deamon). Kullanıcıların belirli zaman aralıklarında komutların, komut dosyalarının (bir grup komut) veya programların yürütülmesini otomatikleştirmesini sağlar.

Cron, Linux kullanıcılarının herhangi bir görevi zamanlamasına yardımcı olan bir sistemdir. Ancak, bir cron işi, belirli bir zaman diliminde çalıştırılacak tanımlanmış herhangi bir görevdir. Bir kabuk betiği veya basit bir bash komutu olabilir. Cron işi, rutin görevlerimizi otomatikleştirmemize yardımcı olur, saatlik, günlük, aylık vb.

crontab için komutlar (script çalıştırıyoruz bundan dolayı makro olarak adlandırılıyor);

> Hazırladığımız scripti root ile çalıştırılmalıdır(Subject'te var);

Aşağıda kodun beklenen çıktısı gösterilmiştir:

```
Broadcast message from root@wil (tty1) (Sun Apr 25 15:45:00 2021):  
#Architecture: Linux wil 4.19.0-16-amd64 #1 SMP Debian 4.19.181-1 (2021-03-19) x86_64 GNU/Linux  
#CPU physical : 1  
#vCPU : 1  
#Memory Usage: 74/987MB (7.50%)  
#Disk Usage: 1009/2Gb (39%)  
#CPU load: 6.7%  
#Last boot: 2021-04-25 14:45  
#LVM use: yes  
#Connexions TCP : 1 ESTABLISHED  
#User log: 1  
#Network: IP 10.0.2.15 (08:00:27:51:9b:a5)  
#Sudo : 42 cmd
```

> Sudo crontab -u root -e ile makro ayarı yapılıyor. -u user, -e ise edit anlamına geliyor.

```
akaraca@akaraca42:~$ sudo crontab -u  
crontab: option requires an argument -- 'u'  
crontab: usage error: unrecognized option  
usage: crontab [-u user] file  
    crontab [ -u user ] [ -i ] { -e | -l | -r }  
                  (default operation is replace, per 1003.2)  
    -e      (edit user's crontab)  
    -l      (list user's crontab)  
    -r      (delete user's crontab)  
    -i      (prompt before deleting user's crontab)
```

> Komutu yarıda kesmek için; "sudo /etc/init.d/cron stop"

```
akaraca@akaraca42:~$ sudo /etc/init.d/cron stop  
Stopping cron (via systemctl): cron.service.
```

> Başlatmak için; "sudo /etc/init.d/cron start"

```
akaraca@akaraca42:~$ sudo /etc/init.d/cron start  
Starting cron (via systemctl): cron.service.
```

> Durumu kontrol etmek için; "sudo /etc/init.d/cron states"

```

akaraca@akaraca42:~$ sudo /etc/init.d/cron status
● cron.service - Regular background program processing daemon
  Loaded: loaded (/lib/systemd/system/cron.service; enabled; vendor preset: enabled)
  Active: active (running) since Tue 2022-03-22 12:17:54 EDT; 45s ago
    Docs: man:crond(8)
   Main PID: 979 (cron)
     Tasks: 1 (limit: 2336)
    Memory: 352.0K
      CPU: 187ms
     CGroup: /system.slice/cron.service
             └─979 /usr/sbin/cron -f

Mar 22 12:17:54 akaraca42 systemd[1]: Started Regular background program processing daemon.
Mar 22 12:17:54 akaraca42 cron[979]: (CRON) INFO (pidfile fd = 3)
Mar 22 12:17:54 akaraca42 cron[979]: (CRON) INFO (Skipping @reboot jobs -- not system startup)
Mar 22 12:18:01 akaraca42 CRON[985]: pam_unix(cron:session): session opened for user akaraca(uid=1000) by (uid=0)
Mar 22 12:18:01 akaraca42 CRON[986]: (akaraca) CMD (bash /usr/local/bin/monitoring.sh)
Mar 22 12:18:01 akaraca42 CRON[985]: (CRON) info (No MTA installed, discarding output)
Mar 22 12:18:01 akaraca42 CRON[985]: pam_unix(cron:session): session closed for user akaraca
Hint: Some lines were ellipsized, use -l to show in full.

```

> Hangi komutlar kullanılabilir görmek için;

```

akaraca@akaraca42:~$ sudo /etc/init.d/cron .
Usage: /etc/init.d/cron {start|stop|status|restart|reload|force-reload} .

```

—> Evo için kullanılan bir kaç komut.

```

akaraca@akaraca42:~$ head -n 2 /etc/os-release
PRETTY_NAME="Debian GNU/Linux 11 (bullseye)"
NAME="Debian GNU/Linux"
akaraca@akaraca42:~$ /usr/sbin/aa-status
apparmor module is loaded.
You do not have enough privilege to read the profile set.
akaraca@akaraca42:~$ ss -tunlp
Netid      State      Recv-Q      Send-Q          Local Address:Port          Peer Address:Port      Process
udp        UNCONN      0            0              0.0.0.0:68          0.0.0.0:* 
tcp        LISTEN      0            128             0.0.0.0:4242          0.0.0.0:* 
tcp        LISTEN      0            128             [::]:4242           [::]:* 

```

—> Subject'te portların görüntülenmesi için "ss" komutu kullanılmış, bu komut değerleri içerisinde udp türü 68 portu bulunuyor bu portu kapatmak için "kill 540" komutunu kullanıyorum. 540 olarak belirtilen process id'sidir. Lakin sisteme reboot attığımızda bu port tekrardan açılacaktır.

dhclient, DHCP istemcisidir. UDP paketlerini 68 numaralı bağlantı noktasından alması gereklidir. Bilgisayarınız ağ bağlantısını yapılandırmak için DHCP kullanıyorsa, bunu açık tutmanız gereklidir. Bu bilgisayarda DHCP kullanmıyorsanız, dhclient'i tamamen kaldırabilirsiniz.

DHCP, 67 (sunucu) ve 68 (istemci) bağlantı noktalarını kullanır. Tasarım gereği UDP yayınlarıdır. Bunu durdurmanın tek yolu, DHCP sunucusunu ağınzınızdan kaldırarak ve IP adreslerini statik olarak atamak olacaktır.

DHCP Sunucusu Nedir? **DHCP** (Dinamik Ana Bilgisayar Yapılandırma Protokolü), IP ağları üzerindeki aygıtların yapılandırma sürecini otomatikleştirmek için kullanılan bir ağ yönetim protokolüdür, böylece DNS, NTP gibi ağ hizmetlerinin ve

UDP veya TCP'ye dayalı herhangi bir iletişim protokolünün kullanımına izin verir.
DHCP kullanmanız gerekiği sürece, DHCP istemcisinin ağdan DHCP mesajları alabilmesi gereklidir (aynı makinede bir DHCP sunucusu çalıştırılmıyorsanız).
kaldırma için; <https://www.snel.com/support/how-to-configure-static-ip-on-debian-10/>

```
root@akaraca42:/home/akaraca# ss -tunpa
Netid      State    Recv-Q    Send-Q      Local Address:Port          Peer Address:Port      Process
udp        UNCONN   0          0           0.0.0.0:68              0.0.0.0:*                users:(:"dhclient",pid=540,fd=9)
tcp        LISTEN   0          128         0.0.0.0:4242            0.0.0.0:*                users:(:"sshd",pid=569,fd=3)
tcp        ESTAB    0          0           10.0.2.15:4242          10.0.2.2:51818           users:(:"sshd",pid=601,fd=4),("sshd",pid=588,fd=4))
tcp        LISTEN   0          128         [::]:4242              [::]:*                  users:(:"sshd",pid=569,fd=4))
root@akaraca42:/home/akaraca# kill 540
root@akaraca42:/home/akaraca# ss -tunpa
Netid      State    Recv-Q    Send-Q      Local Address:Port          Peer Address:Port      Process
tcp        LISTEN   0          128         0.0.0.0:4242            0.0.0.0:*                users:(:"sshd",pid=569,fd=3)
tcp        ESTAB    0          0           10.0.2.15:4242          10.0.2.2:51818           users:(:"sshd",pid=601,fd=4),("sshd",pid=588,fd=4))
tcp        LISTEN   0          128         [::]:4242              [::]:*                  users:(:"sshd",pid=569,fd=4))
```

udp'yi sayısal olarak görüntüleyince bootpc portu 68 olarak adlandırıyor. bootpc için bağlantı durumunu dhclient olarak belirtmiş.

dhclient şu şekilde tanımlanmış; "Dinamik Ana Bilgisayar Yapılandırma Protokolü(DHCP), bir istemci-sunucu mimarisi kullanarak ağa bağlı cihazlara IP adreslerini ve diğer iletişim parametrelerini otomatik olarak atamak için İnternet Protokolü ağlarında kullanılan bir ağ yönetim protokolüdür."

Sadece tcp üzerinden değil udp üzerinden de bağlantı sağlanabildiği üzerine bilgi var. Bu yüzden güvenilir olmadığı hakkında yorumlarda bulunulmuş. Ne kadar doğru bilmiyorum.

>>>>><<https://explainshell.com/>><<<<<<

—> monitoring.sh olarak oluşturduğumuz scriptimiz.

```

#!/bin/bash
arc=$(uname -a)
pcpu=$(grep "physical id" /proc/cpuinfo | sort | uniq | wc -l)
vcpu=$(grep "^processor" /proc/cpuinfo | wc -l)
fram=$(free -m | awk '$1 == "Mem:" {print $2}')
uram=$(free -m | awk '$1 == "Mem:" {print $3}')
pram=$(free | awk '$1 == "Mem:" {printf("%.2F"), $3/$2*100}')
fdisk=$(df -Bm | grep '^/dev/' | grep -v '/boot$' | awk '{ft += $2} END {print ft}')
udisk=$(df -Bm | grep '^/dev/' | grep -v '/boot$' | awk '{ut += $3} END {print ut}')
pdisk=$(df -Bm | grep '^/dev/' | grep -v '/boot$' | awk '{ft+= $2} END {printf("%d"), ut/ft*100}')
cpul=$(top -bn1 | grep '^%Cpu' | cut -c 9- | xargs | awk '{printf("%.1f%%"), $1 + $3}')
lb=$(who -b | awk '$1 == "system" {print $3 " " $4}')
lvmt=$(lsblk | grep 'lvm' | wc -l)
lvmu=$(if [ $lvmt -eq 0 ]; then echo no; else echo yes; fi)
#You need to install net tools for the next step [ $ sudo apt install net-tools]
ctcp=$(cat /proc/net/sockstat[6] | awk '$1 == "TCP:" {print $3}')
ulog=$(users | wc -w)
ip=$(hostname -I)
mac=$(ip link show | awk '$1 == "link/ether" {print $2}')
cmds=$(journalctl _COMM=sudo | grep COMMAND | wc -l) # journalctl should be running as sudo but our script is running as root so we don't need in sudo here
wall "#Architecture: $arc
#CPU physical: $pcpu
#vCPU: $vcpu
#Memory Usage: $uram/${fram}MB ($pram%)
#Disk Usage: $udisk/${fdisk}Gb ($pdisk%)
#CPU load: $cpul
#Last boot: $lb
#LVM use: $lvmu
#Connexions TCP: $ctcp ESTABLISHED
#User log: $ulog
#Network: IP $ip ($mac)
#Sudo: $cmds cmd" # broadcast our system information on all terminals

```

Broadcast message from root@akaraca42 (somewhere) (Wed Mar 23 06:30:01 2022):

```

#Architecture: Linux akaraca42 5.10.0-12-amd64 #1 SMP Debian 5.10.103-1 (2022-03-07) x86_64 GNU/Linux
#CPU physical: 1
#vCPU: 2
#Memory Usage: 73/1982MB (3.72%)
#Disk Usage: 1520/28Gb (5%)
#CPU load: 6.2%
#Last boot: 2022-03-23 06:04
#LVM use: yes
#Connexions TCP: 2 ESTABLISHED
#User log: 1
#Network: IP 10.0.2.15 (08:00:27:ee:c6:71)
#Sudo: 180 cmd

```

Aşağıda kodun beklenen çıktısı gösterilmiştir:

Broadcast message from root@wil (tty1) (Sun Apr 25 15:45:00 2021):

```

#Architecture: Linux wil 4.19.0-16-amd64 #1 SMP Debian 4.19.181-1 (2021-03-19) x86_64 GNU/Linux
#CPU physical : 1
#vCPU : 1
#Memory Usage: 74/987MB (7.50%)
#Disk Usage: 1009/2Gb (39%)
#CPU load: 6.7%
#Last boot: 2021-04-25 14:45
#LVM use: yes
#Connexions TCP : 1 ESTABLISHED
#User log: 1
#Network: IP 10.0.2.15 (08:00:27:51:9b:a5)
#Sudo : 42 cmd

```

#####

uname ile sistem ile ilgili belirli sistem bilgilerinin çıktısı alınabilir. Çıktı sonucu bilinmeyen sonuçlar "all" durumunda sergilenebilir.

- İşletim sisteminizin mimarisi ve kernel versiyonu.

```
akaraca@akaraca42:~$ uname -a
Linux akaraca42 5.10.0-12-amd64 #1 SMP Debian 5.10.103-1 (2022-03-07) x86_64 GNU/Linux
```

Linux : İşletim sisteminin adı (İşletim sisteminin bir parçası olan sistem yazılımı)
(uname -s)

akaraca42 : hostname (uname -n)

5.10.0-12-amd64 : Çekirdek sürümü (uname -r)

#1 SMP Debian 5.10.103-1 (2022-03-07) : Kernel versiyonu (uname -v)

x86_64 : Makine donanım adı (uname -m)

GNU/Linux : İşletim sisteminin yazılım adını verir (uname -o)

GNU; çekirdeği, sistem araçlarını, açıcılarını, kütüphanelerini ve son kullanıcı yazılımlarını içeren, GNU Tasarısı kapsamında geliştirilen bir işletim sistemidir. İsminin açılımı "GNU's Not Unix" dir.

```
akaraca@akaraca42:~$ uname --help
Usage: uname [OPTION]...
Print certain system information. With no OPTION, same as -s.

-a, --all           print all information, in the following order,
                   except omit -p and -i if unknown:
-s, --kernel-name  print the kernel name
-n, --nodename     print the network node hostname
-r, --kernel-release  print the kernel release
-v, --kernel-version  print the kernel version
-m, --machine      print the machine hardware name
-p, --processor    print the processor type (non-portable)
-i, --hardware-platform  print the hardware platform (non-portable)
-o, --operating-system  print the operating system
--help            display this help and exit
--version         output version information and exit

GNU coreutils online help: <https://www.gnu.org/software/coreutils/>
Full documentation <https://www.gnu.org/software/coreutils/uname>
or available locally via: info '(coreutils) uname invocation'
```

#####

Fiziksel işlemci sayısı, "lscpu" komutundan da görebileceğimiz üzere socket ile belirtilmektedir. Bu durumu /proc/cpuinfo üzerinden de "physical id"'nin değeri ile anlaşılabilir.

- Fiziksel işlemci sayısı.

Soket, fiziksel CPU kapsüllerinin yerleştirildiği fiziksel yuvalıdır. Normal bir PC'de yalnızca bir soket bulunur. Çekirdekler, CPU kapsülü başına CPU çekirdeği sayısıdır. Standart bir PC için modern bir standart CPU genellikle iki veya dört

çekirdeğe sahiptir.

```
akaraca@akaraca42:~$ grep "physical id" /proc/cpuinfo | sort | uniq | wc -l  
1
```

Grep filtresi, belirli bir karakter kalıbı için bir dosya arar ve bu kalıbı içeren tüm satırları görüntüler.

uniq komutu, belirli bir girdideki benzersiz satırları bulur ve çoğaltılan satırları bildirir veya kaldırır. Bu komut yalnızca sıralanmış verilerle çalışır. Göründüğü gibi iki adet aynı satırdan bulunmaktadır bunu saydırırken çıktı 2 olacağinden uniq komutu kullanılır.

```
akaraca@akaraca42:~$ grep "physical id" /proc/cpuinfo | sort  
physical id      : 0  
physical id      : 0
```

sort komutu dosyaların içeriğini sıralamak için kullanılır.

wc bir dosyadaki karakter, kelime sayısını saymak için kullanılır.

| , komutları ayırmak için kullanılır.

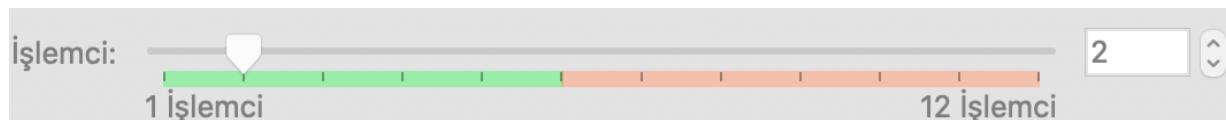
#####

Sanal işlemci sayısı. Bu işlemi yaparken farkı daha kolay anlamak için VM'de işlemci sayısını arttırdım.

Ortalama olarak, fiziksel çekirdek başına dört ila altı vCPU görmelisiniz. Her VM'nin ihtiyaç duyduğundan bir fazla vCPU'su varsa, çekirdek başına yalnızca iki ila üç vCPU alırsınız.

● Sanal işlemci sayısı.

Sanal işlemci sayısı olarak belirlediği şey vCPU'dur.



```
akaraca@akaraca42:~$ cat /proc/cpuinfo
```

VM, 1 işlemci olduğu zamanki çıktısı.

```
processor : 0
vendor_id : GenuineIntel
cpu family : 6
model : 165
model name : Intel(R) Core(TM) i5-10500 CPU @ 3.10GHz
stepping : 3
cpu MHz : 3095.998
cache size : 12288 KB
physical id : 0
siblings : 1
core id : 0
cpu cores : 1
apicid : 0
initial apicid : 0
fpu : yes
fpu_exception : yes
cpuid level : 22
wp : yes
flags : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 ht syscall nx rdtscp lm constant_tsc rep_good nopl xtopology nonstop_tsc cpuid tsc_known_freq pni pclmulqdq monitor ssse3 cx16 pcid sse4_1 sse4_2 x2apic movbe popcnt aes xsave avx rdrand hypervisorlahf_lm dbm 3dnowprefetch
bugs : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds swapgs itlb_multihit
bogomips : 6191.99
clflush size : 64
cache_alignment : 64
address sizes : 39 bits physical, 48 bits virtual
power management:
```

```
akaraca@akaraca42:~$ grep "processor" /proc/cpuinfo | sort | wc -l
1
```

VM, 2 işlemci olduğu zamanki çıktısı.

```

akaraca@akaraca42:~$ cat /proc/cpuinfo
processor      : 0
vendor_id     : GenuineIntel
cpu family    : 6
model         : 165
model name    : Intel(R) Core(TM) i5-10500 CPU @ 3.10GHz
stepping       : 3
cpu MHz       : 3095.998
cache size    : 12288 KB
physical id   : 0
siblings       : 2
core id        : 0
cpu cores     : 2
apicid         : 0
initial apicid: 0
fpu            : yes
fpu_exception  : yes
cpuid level   : 22
wp             : yes
flags          : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 ht syscall nx rdtscp lm constan
t_tsc rep_good nopl xtopology nonstop_tsc cpuid tsc_known_freq pni pclmulqdq ssse3 cx16 pcid sse4_1 sse4_2 x2apic movbe popcnt aes xsave avx rdrand
hypervisor lahf_lm abm 3dnowprefetch invpcid_single pti fsgsbase avx2 invpcid rdseed clflushopt md_clear flush_l1d arch_capabilities
bugs           : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds swapgs itlb_multihit
bogomips      : 6191.99
clflush size   : 64
cache_alignment: 64
address sizes  : 39 bits physical, 48 bits virtual
power management:

processor      : 1
vendor_id     : GenuineIntel
cpu family    : 6
model         : 165
model name    : Intel(R) Core(TM) i5-10500 CPU @ 3.10GHz
stepping       : 3
cpu MHz       : 3095.998
cache size    : 12288 KB
physical id   : 0
siblings       : 2
core id        : 1
cpu cores     : 2
apicid         : 1
initial apicid: 1
fpu            : yes
fpu_exception  : yes
cpuid level   : 22
wp             : yes
flags          : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 ht syscall nx rdtscp lm constan
t_tsc rep_good nopl xtopology nonstop_tsc cpuid tsc_known_freq pni pclmulqdq ssse3 cx16 pcid sse4_1 sse4_2 x2apic movbe popcnt aes xsave avx rdrand
hypervisor lahf_lm abm 3dnowprefetch invpcid_single pti fsgsbase avx2 invpcid rdseed clflushopt md_clear flush_l1d arch_capabilities
bugs           : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds swapgs itlb_multihit
bogomips      : 6191.99
clflush size   : 64
cache_alignment: 64
address sizes  : 39 bits physical, 48 bits virtual
power management:

```

```

akaraca@akaraca42:~$ grep "processor" /proc/cpuinfo | sort | wc -l
2

```

^ caret(şapka) işaretini bir karakter dizisininde başta eşleşen karakterleri bulmak için kullanılır.

```

akaraca@akaraca42:~$ lscpu | grep "CPU(s)"
CPU(s):                      2
On-line CPU(s) list:          0,1
NUMA node0 CPU(s):            0,1
akaraca@akaraca42:~$ lscpu | grep "^CPU(s)"
CPU(s):                      2

```

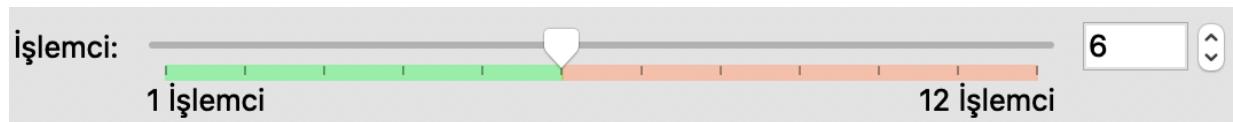
???Ana makinemizdeki İşlemci sayımız 1, Toplam çekirdek sayımızı 6'dır. Bunun nedeni sistemin şu anda 1 çekirdek ile işlem yaptığı anlamına gelir. Bu durumu

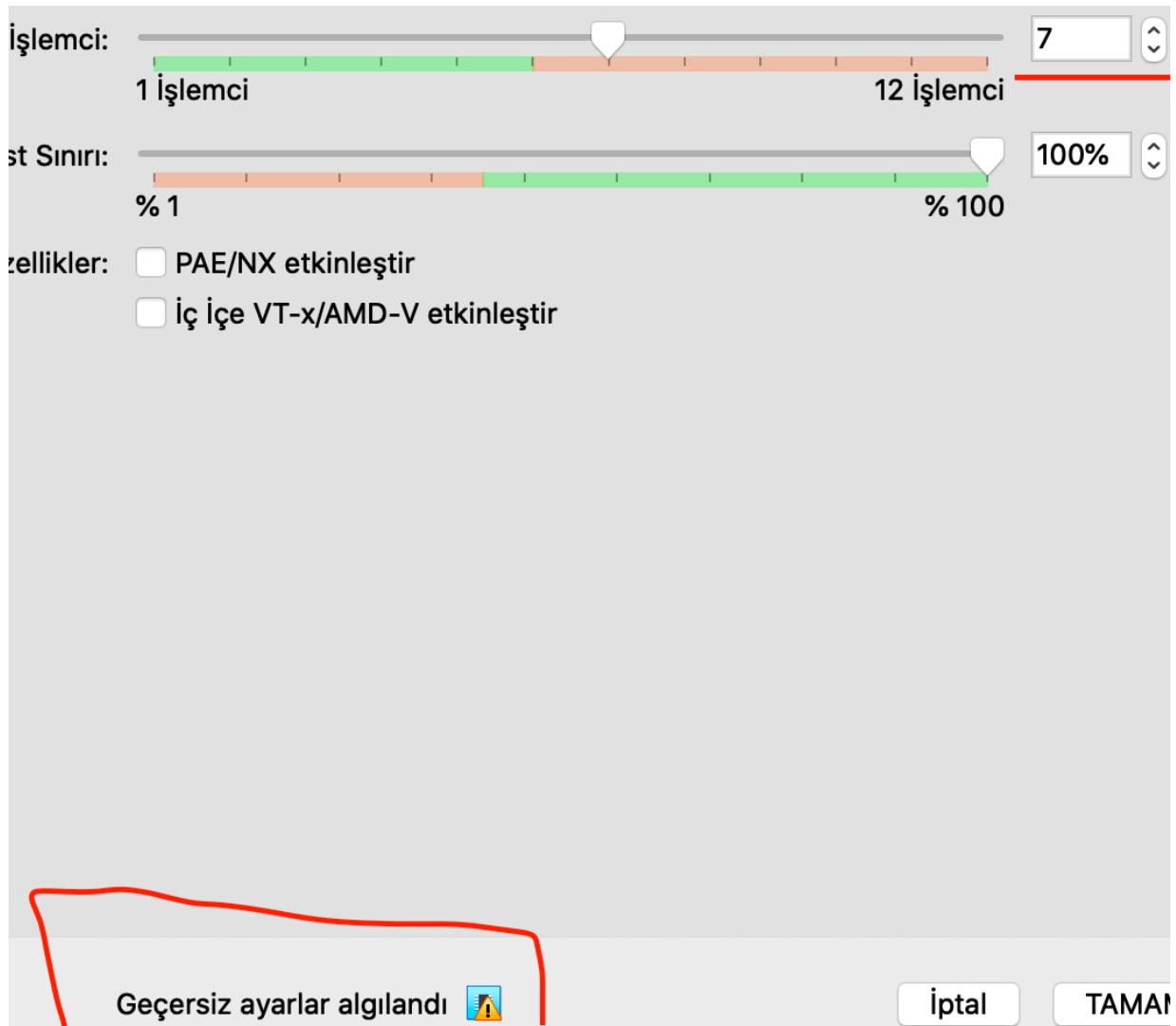
VM'de çekirdek sayını arttırken görebiliriz. ???

iMac

Genel Donanım Bilgileri:

Model Adı:	iMac
Model Tanıtıçısı:	iMac20,1
İşlemci Adı:	6-Core Intel Core i5
İşlemci Hızı:	3,1 GHz
İşlemci Sayısı:	1
Toplam Çekirdek Sayısı:	6
L2 Önbellek (her bir çekirdek için):	256 KB
L3 Önbellek:	12 MB
Hyper-Threading Teknolojisi:	Etkin
Bellek:	8 GB





#####

Ram değerleri:

- Sunucunun erişilebilir RAM'i ve yüzde olarak RAM'in kullanım oranı.

free komutu ile sistemde bulunan ram + swap değerleri gösterilir. Swap bölümü ram ile etkileşimli olduğundan dolayı çıktıda ikisi birlikte verilmektedir.

Free komutu, bellek hakkında birkaç satır gösteren çok basit bir komuttur. Free komutu, toplam hafızayı, kullanılan hafızayı, boş hafızayı, paylaşılan hafızayı ve RAM ve takas alanı ile ilgili kullanılabilir hafızayı göstermek için kullanılabilir.

```
akaraca@akaraca42:~$ free
              total        used        free      shared  buff/cache   available
Mem:       2030308       75112     1810876          572      144320     1815488
Swap:      2244604           0     2244604
```

Gösterim türü bayt türündedir. Anlaşılabilirliği arrtırmak için "free -m" yani

megabayt türünde görüntüleme yapılır.

```
akaraca@akaraca42:~$ free -m
              total        used        free      shared  buff/cache   available
Mem:       1982           73       1768           0         140       1772
Swap:      2191           0       2191
```

Ram için istenilenleri çekerememizden dolayı awk komutu ile 1.sütundaki Mem ile eşleşen dizeyi bul ve 2. sütun değerini yazdır komutunu uyguluyoruz. Erişilebilir ram oranını istediginden dolayı "total" kısmı yazdırırmak yeterlidir.

```
akaraca@akaraca42:~$ free -m | awk '$1 == "Mem:" {print $2}'
1982
```

Print{.....} şeklinde süslü parantez içerisindeki değerleri yazdırabiliyoruz.
awk, verileri işlemek ve raporlar oluşturmak için kullanılan bir betik dilidir. awk komut programlama dili derleme gerektirmez ve kullanıcının değişkenleri, sayısal işlevleri, dize işlevlerini ve mantıksal operatörleri kullanmasına izin verir.
Kullanım oranını istediginden dolayı anlık olarak kullanılan ram değerininde bulunması gerekmektedir. Bu durumda "used" kısmı yazdırılacaktır.

```
akaraca@akaraca42:~$ free -m | awk '$1 == "Mem:" {print $3}'
73
```

Oran olarak çıktı istediginden dolayı bu iki çıktıyı ekrana yazdırmadan bir değişken olarak tutup, çıktıyı verecek değişken içerisinde kullanmamız gerekmektedir. Yani "used/total MB" şeklinde yazdırılır.

```
fram=$(free -m | awk '$1 == "Mem:" {print $2}')
uram=$(free -m | awk '$1 == "Mem:" {print $3}')
```

```
#Memory Usage: ${uram}/${fram}MB
```

Yüzde olarak kullanım oranını istediginden dolayı, float tarzında noktadan sonra 2 basamaklı çıktı verecek şekilde oranı alınır ve değer çağrılır.

```
pram=$(free | awk '$1 == "Mem:" {printf("%.2f"), $3/$2*100}')
```

```
($pram%)
```

print yerine printf kullanılmasının nedeni istenilen çıktıya sınırlama getirmek için yönlendirici durumunu kullandık. Lakin bu komut değerin çağrılması durumunda düzgünce çalışacaktır.

```
3.67806.2fakaraca@akaraca42:~$ free | awk '$1 == "Mem:" {print("%s.2f"), $3/$2*100}'  
%s.2f 3.67727  
akaraca@akaraca42:~$ free | awk '$1 == "Mem:" {printf("%s.2f"), $3/$2*100}'  
3.67727.2fakaraca@akaraca42:~$
```

yada alttaki kod ile direkt çıktı alınabilir. "%.*2f*%" ne anlama geliyor: %s yerine % şeklinde belirttiğimiz durumda printf içeresine virgülden sonraki değeri koyarak yazdırma yerine, virgülden sonraki değeri nasıl yazdıracağını belirtir.

```
3.69%akaraca@akaraca42:~$ free | grep "Mem:" | awk '{printf("%.2f%%", $3/$2*100)}'
```

```
#####
# Disk değerlerini(used/total) ve oranını yazdırma
```

- Sunucunun erişilebilir depolama alanı ve yüzde olarak depolama alanı kullanım oranı.

```
fdisk=$(df -Bg | grep '^/dev/' | grep -v '/boot$' | awk '{ft += $2} END {print ft}')  
udisk=$(df -Bm | grep '^/dev/' | grep -v '/boot$' | awk '{ut += $3} END {print ut}')  
pdisk=$(df -Bm | grep '^/dev/' | grep -v '/boot$' | awk '{ut += $3} {ft+= $2} END {printf("%d"), ut/ft*100}')
```

```
#Disk Usage: ${udisk}/${fdisk}Gb (${pdisk}%)
```

'df' komutu "disk filesystem(disk dosya sistemi)" anlamına gelir, Linux sistemindeki dosya sisteminin kullanılabilir ve kullanılan disk alanı kullanımının tam bir özeti almak için kullanılır.

df -B ile dizinlerin boyutu görüntülenebilir lakin -B kendi başına yeterli değildir bu yüzden gösterim biçimini yani Kb, Mb, Gb... hangi türde göstermek istiyorsak buna göre karakterleri kullanmamız gerekiyor.

df -Bg veya df -Bm ile gigabayt/megabayt şeklinde görüntüyü alabiliriz.

```
-B, --block-size=SIZE scale sizes by SIZE before printing them; e.g.,  
'-BM' prints sizes in units of 1,048,576 bytes;  
see SIZE format below
```

Filesystem	1G-blocks	Used	Available	Use%	Mounted on
udev	1G	0G	1G	0%	/dev
tmpfs	1G	1G	1G	1%	/run
/dev/mapper/LVMGroup-root	10G	2G	8G	13%	/
tmpfs	1G	0G	1G	0%	/dev/shm
tmpfs	1G	0G	1G	0%	/run/lock
/dev/mapper/LVMGroup-var	3G	1G	3G	10%	/var
/dev/mapper/LVMGroup-srv	3G	1G	3G	1%	/srv
/dev/sda1	1G	1G	1G	21%	/boot
/dev/mapper/LVMGroup-tmp	3G	1G	3G	1%	/tmp
/dev/mapper/LVMGroup-var--log	4G	1G	4G	7%	/var/log
/dev/mapper/LVMGroup-home	5G	1G	5G	1%	/home
tmpfs	1G	0G	1G	0%	/run/user/1000

^ , caret(şapka) işaretini ile dizinin başında aranan kelimeyi bulur. Bu komut sayesinde geçici dosya olarak yer kaplayan dosyalar ayırtılırlar.

akaraca@akaraca42:~\$ df -Bg grep '/dev/'					
/dev/mapper/LVMGroup-root	10G	2G	8G	13%	/
tmpfs	1G	0G	1G	0%	/dev/shm
/dev/mapper/LVMGroup-var	3G	1G	3G	10%	/var
/dev/mapper/LVMGroup-srv	3G	1G	3G	1%	/srv
/dev/sda1	1G	1G	1G	21%	/boot
/dev/mapper/LVMGroup-tmp	3G	1G	3G	1%	/tmp
/dev/mapper/LVMGroup-var--log	4G	1G	4G	7%	/var/log
/dev/mapper/LVMGroup-home	5G	1G	5G	1%	/home
akaraca@akaraca42:~\$ df -Bg grep '^/dev/'					
/dev/mapper/LVMGroup-root	10G	2G	8G	13%	/
/dev/mapper/LVMGroup-var	3G	1G	3G	10%	/var
/dev/mapper/LVMGroup-srv	3G	1G	3G	1%	/srv
/dev/sda1	1G	1G	1G	21%	/boot
/dev/mapper/LVMGroup-tmp	3G	1G	3G	1%	/tmp
/dev/mapper/LVMGroup-var--log	4G	1G	4G	7%	/var/log
/dev/mapper/LVMGroup-home	5G	1G	5G	1%	/home

/dev/sda1 olarak ele alınan disk bölümü boot bölümü olduğundan dolayı bizim oturumumuz esnasında o bölümden kullanımı olmayacağından dolayı ayrı tutulması gerekiyor. Bunun için "grep -v" ile boot dizini ayırtılırlar.

"grep -v" , Eşleşmeyen çizgileri seçmek için eşleşme durumunu ters çevirir.

```

akaraca@akaraca42:~$ df -Bg | grep '^/dev/'
/dev/mapper/LVMGroup-root          10G   2G     8G  13% /
/dev/mapper/LVMGroup-var           3G   1G     3G  10% /var
/dev/mapper/LVMGroup-srv           3G   1G     3G  1% /srv
/dev/sda1                          1G   1G     1G  21% /boot
/dev/mapper/LVMGroup-tmp           3G   1G     3G  1% /tmp
/dev/mapper/LVMGroup-var--log      4G   1G     4G  7% /var/log
/dev/mapper/LVMGroup-home          5G   1G     5G  1% /home
akaraca@akaraca42:~$ df -Bg | grep '^/dev/' | grep -v '/boot$'
/dev/mapper/LVMGroup-root          10G   2G     8G  13% /
/dev/mapper/LVMGroup-var           3G   1G     3G  10% /var
/dev/mapper/LVMGroup-srv           3G   1G     3G  1% /srv
/dev/mapper/LVMGroup-tmp           3G   1G     3G  1% /tmp
/dev/mapper/LVMGroup-var--log      4G   1G     4G  7% /var/log
/dev/mapper/LVMGroup-home          5G   1G     5G  1% /home

```

Sadece oturum içerisindeki disk kullanımından sorumlu olan dizinleri elde etmiş olduk. Artık bu dizinlerdeki sütun değerlerini toplayıp çıktı almamız gerekmektedir.

script içinde tanımı kolay olması adına ft(format total)(toplam biçim, toplam boyut) olarak adlandırmış lakin bu isimlendirme istege görede yapılabilir.

awk ile çalıştırmayı istediğimiz komutta 2.sütundaki tüm değerleri sırasıyla toplayıp "ft = ft + \$2" şeklinde bulunduğu listenin sonuna kadar toplayıp ft'yi yazdırmayı istemektedir.

'/boot\$' şeklinde belirtmemizin nedeni boot'un bir bölüm olarak dahil edilmemesi içindir ve /dev/ içerisinde kullanıcı tarafından oluşturulan bir boot adlı dizin varsa bunu dahil etmek istemekteyiz.

\$ komut eklentisi aranan dizinin sonuna baktmaktadır.

```

akaraca@akaraca42:~$ df -Bg | grep '^/dev/' | grep -v '/boot$' | awk '{ft += $2} END {print ft}'
28
akaraca@akaraca42:~$ df -Bg | grep '^/dev/' | grep -v '/boot$' | awk '{aaa += $2} END {print aaa}'
28

```

aynı şekilde bunu ut(used total)(kullanılan toplam) içinde yapıyoruz lakin bu gigabayt şeklinde olması Subjectte istenilmemiş, megabayt türünde istenmiş.

```

akaraca@akaraca42:~$ df -Bg | grep '^/dev/' | grep -v '/boot$' | awk '{ut += $3} END {print ut}'
7

```

#Disk Usage: 1009/2Gb (39%)

Bundan dolayı "df -Bg" değil "df -Bm" kullanılır.

```

akaraca@akaraca42:~$ df -Bm | grep '^/dev/' | grep -v '/boot$' | awk '{ut += $3} END {print ut}'
1520

```

Gigabayt türünde oran alırsak sağlıklı bir oran çıkmayacaktır bunun nedeni yuvarlama işlemi yapıldığından dolayıdır. Hatasız bir oran alabilmek için kilobayt

veya megabayt türünde oran alınır.

```
akaraca@akaraca42:~$ df -Bg | grep '^/dev/' | grep -v '/boot$' | awk '{ut += $3} {ft+= $2} END {printf("%d"), ut/ft*100}'  
25  
akaraca@akaraca42:~$ df -Bm | grep '^/dev/' | grep -v '/boot$' | awk '{ut += $3} {ft+= $2} END {printf("%d"), ut/ft*100}'  
5  
akaraca@akaraca42:~$ df -Bk | grep '^/dev/' | grep -v '/boot$' | awk '{ut += $3} {ft+= $2} END {printf("%d"), ut/ft*100}'  
5  
akaraca@akaraca42:~$
```

Ya da bu komutta kullanılabilir.

```
akaraca@akaraca42:~$ df -Bm | grep '^/dev/' | grep -v '/boot$' | awk '{ut += $3} {ft+= $2} END {printf("%.2f%%"), ut/ft*100}'  
5.89%  
akaraca@akaraca42:~$
```

```
#####
```

İşlemcimizin anlık olarak kullanım oranının çıktısı istenmektedir.

● Yüzde olarak işlemcinin kullanım oranı.

```
cput=$(top -bn1 | grep '^%Cpu' | cut -c 9- | xargs | awk '{printf("%.1f%%"), $1 + $3}')
```

```
#CPU load: $cput
```

top (Total of processes)(işlem tablosu) komutu, Linux'ta çalışan işlemlerin gerçek zamanlı bir görünümünü gösterir ve çekirdek tarafından yönetilen görevleri görüntüler. Komut ayrıca, CPU ve bellek kullanımı dahil olmak üzere kaynak kullanımını gösteren bir sistem bilgisi özeti sağlar.

```
top - 09:06:45 up 3:04, 1 user, load average: 0.00, 0.00, 0.00  
Tasks: 107 total, 1 running, 105 sleeping, 1 stopped, 0 zombie  
%Cpu(s): 0.0 us, 0.2 sy, 0.0 ni, 99.8 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st  
MiB Mem : 1982.7 total, 1767.6 free, 73.4 used, 141.7 buff/cache  
MiB Swap: 2192.0 total, 2192.0 free, 0.0 used. 1772.9 avail Mem  


| PID | USER | PR | NI  | VIRT  | RES   | SHR  | S | %CPU | %MEM | TIME+   | COMMAND                     |
|-----|------|----|-----|-------|-------|------|---|------|------|---------|-----------------------------|
| 1   | root | 20 | 0   | 98404 | 10220 | 7748 | S | 0.0  | 0.5  | 0:00.92 | systemd                     |
| 2   | root | 20 | 0   | 0     | 0     | 0    | S | 0.0  | 0.0  | 0:00.00 | kthreadd                    |
| 3   | root | 0  | -20 | 0     | 0     | 0    | I | 0.0  | 0.0  | 0:00.00 | rcu_gp                      |
| 4   | root | 0  | -20 | 0     | 0     | 0    | I | 0.0  | 0.0  | 0:00.00 | rcu_par_gp                  |
| 6   | root | 0  | -20 | 0     | 0     | 0    | I | 0.0  | 0.0  | 0:00.00 | kworker/0:0H-events_highpri |
| 9   | root | 0  | -20 | 0     | 0     | 0    | I | 0.0  | 0.0  | 0:00.00 | mm_percpu_wq                |
| 10  | root | 20 | 0   | 0     | 0     | 0    | S | 0.0  | 0.0  | 0:00.00 | rcu_tasks_rude_             |
| 11  | root | 20 | 0   | 0     | 0     | 0    | S | 0.0  | 0.0  | 0:00.00 | rcu_tasks_trace             |
| 12  | root | 20 | 0   | 0     | 0     | 0    | S | 0.0  | 0.0  | 0:00.02 | ksoftirqd/0                 |


```

b (branch mode)(Toplu modu) "top -b" ile Komut çıktısının bir dosyaya veya diğer programlara gönderilmesine izin vererek toplu modda top başlatır. top toplu modda girişi kabul etmez ve öldürülene veya belirtilen yineleme sınırına ulaşana kadar çalışır.

"top -bn1" ile yineleme sınırını belirlemiş oluyoruz yani tek bir çıktı almasını istiyoruz.

n (Limit iteration number)(Yineleme sayısını sınırla), Bitirmeden önce topun üretmesini istediğiniz maksimum yineleme sayısını belirtmeye yarar.

altta bulunan görsel "top -bn2" komutunu kullandım ve 2 kerelik yineleme gerçekleştirdi yani 2 adet çıktı verdi.



%CPU -- CPU Kullanımı : İşlem tarafından kullanılan CPU'nuzun yüzdesi. Varsayılan olarak, top bunu tek bir CPU'nun yüzdesi olarak görüntüler. Çok çekirdekli sistemlerde %100'den büyük yüzdelere sahip olabilirsiniz. Örneğin, 3 çekirdek %60 kullanımdaysa, top %180 CPU kullanımını gösterir.
Bundan dolayı işlem tablosundan %Cpu değerini çekiyoruz.

```
akaraca@akaraca42:~$ top -bn1
top - 11:29:25 up 15 min,  1 user,  load average: 0.00, 0.00, 0.00
Tasks: 106 total,   1 running, 104 sleeping,   1 stopped,   0 zombie
%Cpu(s):  0.0 us,  3.1 sy,  0.0 ni, 96.9 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem : 1982.7 total, 1823.7 free,    70.6 used,   88.5 buff/cache
MiB Swap: 2192.0 total, 2192.0 free,      0.0 used. 1794.1 avail Mem

          PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND

```

```
akaraca@akaraca42:~$ top -bn1 | grep 'Cpu'
%Cpu(s):  0.0 us,  3.1 sy,  0.0 ni, 96.9 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
akaraca@akaraca42:~$ top -bn1 | grep '^Cpu'
akaraca@akaraca42:~$ top -bn1 | grep '^%Cpu'
%Cpu(s):  3.1 us,  0.0 sy,  0.0 ni, 96.9 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
```

Top komutunu kullandığımızda cpu kullanımının "sistem" ve "kullanıcı" gibi kategorileri ayrılır. Bu iki ana kategorinin yanı sıra, birkaç tane daha ve bazı alt kategoriler var. CPU darboğazlarını bulmak için bildirilen bilgileri kullanabilmeniz için bunların her birinin ne anlama geldiğine bakacağız.

<https://blog.appsignal.com/2018/03/06/understanding-cpu-statistics.html>

Bir Linux makinesinde, top çalıştırıldığında şuna benzeyen bir CPU satırı yazdırılır:

```
%Cpu(s): 13.2 us,  1.3 sy,  0.0 ni, 85.3 id,  0.0 wa,  0.0 hi,  0.2 si,  0.0 st
```

"user(kullanıcı)" (**us**), "system(sistem)", (**sy**), "nice(güzel)" (**ni**), "idle(boşta)" (**id**), "iowait" (**wa**), "hardware interrupt(donanım kesintisi) (**hi**)", "software interrupt(yazılım kesintisi)" (**si**), and "steal(hırsızlık, çalıntı) (**st**).

Bu sekizden "system(sistem)", "user(kullanıcı)" ve "idle(boşta)", CPU'nun içinde olabileceği ana durumlardır.

system (**sy**);
 > "Sistem" CPU durumu, çekirdek tarafından kullanılan CPU süresini gösterir. Çekirdek, donanımla etkileşim, bellek ayırma, işletim sistemi süreçleri arasında iletişim kurma, aygit sürücülerini çalıştırma ve dosya sistemini yönetme gibi düşük seviyeli görevlerden sorumludur. Hangi işlemin CPU'ya erişeceğini belirleyen CPU zamanlayıcısı bile çekirdek tarafından çalıştırılır.

> Genellikle düşük olmakla birlikte, örneğin diskten çok fazla veri okunurken veya diske yazılırken "sistem" kategorisi ani bir artış gösterebilir. Daha uzun süre yüksek kalırsa, aygıt sürücüsüyle ilgili bir sorununuz olabilir.

user (**us**);

> Bir seviye yukarı, "kullanıcı" CPU durumu, kullanıcı alanı süreçleri tarafından kullanılan CPU zamanını gösterir. Bunlar, uygulamanız veya makinenizde çalışan veritabanı sunucusu gibi daha üst düzey işlemlerdir. Kisacası, herhangi bir kullanıcı hesabından başlatılmamış olsa bile, çekirdek dışında herhangi bir şey tarafından kullanılan her CPU zamanı "kullanıcı" olarak işaretlenir. Bir kullanıcı alanı işleminin donanıma erişmesi gerekiyorsa, çekirdeğe sorması gereklidir, bu da "sistem" durumuna sayılacağı anlamına gelir.

> Genellikle "kullanıcı" kategorisi CPU zamanınızın çoğunu kullanır. Çok uzun süre boşta kalma süresi bırakmadan maksimum değere yakın kalırsa, uygulamanızda veya makinede çalışan başka bir yardımcı programda bir sorun olabilir.

nice (**ni**);

> "nice(Güzel)" CPU durumu, "kullanıcı" durumunun bir alt kümesidir ve diğer görevlerden daha düşük bir öncelik anlamına gelen pozitif bir güzelliğe sahip işlemler tarafından kullanılan CPU zamanını gösterir. Nice yardımcı programı, belirli bir önceliğe sahip bir programı başlatmak için kullanılır. Varsayılan incelik 0'dır, ancak en yüksek öncelik için -20 ile en düşük öncelik için 19 arasında herhangi bir yere ayarlanabilir. "Güzel" kategorisindeki CPU zamanı, CPU'nun ekstra görevleri yapmak için biraz zamanı kaldığında çalıştırılan daha düşük öncelikli görevleri işaretler.

idle (**id**);

> "idle(Boşta)" CPU durumu, aktif olarak kullanılmayan CPU zamanını gösterir. Dahili olarak, boşta kalma süresi genellikle mümkün olan en düşük önceliğe sahip bir görev tarafından hesaplanır (pozitif bir Nice değeri kullanılarak).

iowait (**wa**);

> "iowait", "idle(boşta)" durumunun bir alt kategorisidir. Diske okuma veya yazma gibi giriş veya çıkış işlemlerini beklemek için harcanan zamanı işaretler. Örneğin işlemci bir dosyanın açılmasını beklediğinde, harcanan zaman "iowait" olarak işaretlenir.

> "iowait" kategorisindeki yüksek CPU süresi, işlemci dışındaki sorunları ortaya çıkarabilir. Örneğin, bir bellek içi veritabanının çok fazla veriyi diske boşaltması gerektiğinde veya bellek diske değiştirildiğinde.

Diğerleri;

> "user(Kullanıcı)"da "nice(güzel)" ve "idle(boşta)" "iowait"ın yanı sıra, ana

CPU durumlarının bölünebileceği daha fazla alt kategori vardır. "Donanım kesintisi" (hi veya irq) ve "yazılım kesintisi" (si veya softirq) kategorileri kesintilere hizmet etmek için harcanan zamandır ve "steal(çalmak)" (st) alt kategorisi, sanal bir makinede sanal bir CPU'yu beklemek için harcanan zamanı gösterir.

İşlemci kullanım oranı olarak kullanıcıya ait olan durumları çekmemiz gerekmektedir. Bu durumda us ve ni sütun değerleri toplanır ve çıktı olarak alınır.

```
akaraca@akaraca42:~$ top -bn1 | grep '^%Cpu' | cut -c 9- | xargs | awk '{printf("%.1f%%"), $1 + $3}'  
3.2%akaraca@akaraca42:~$
```

"cut -c 9-" karakter listesinden baştan 9 karakter haricini göstermesi anlamına gelmektedir. Bunun nedeni ilk sütunun adlandırma kısmı olmasında dolayı.

```
akaraca@akaraca42:~$ top -bn1 | grep '^%Cpu' | awk '{print($1)}'  
%Cpu(s):
```

"xargs" komutu ise ("genişletilmiş ARGumentler"in kısaltması), standart girdiden komutlar oluşturmak ve yürütmek için kullanılan Unix ve çoğu Unix benzeri işletim sistemlerinde bir komuttur. Girdiyi standart girdiden bağımsız değişkenlere bir komuta dönüştürür.

```
akaraca@akaraca42:~$ top -bn1 | grep '^%Cpu' | cut -c 9-  
0.0 us, 0.0 sy, 0.0 ni, 100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st  
akaraca@akaraca42:~$ top -bn1 | grep '^%Cpu' | cut -c 9- | xargs  
0.0 us, 3.1 sy, 0.0 ni, 96.9 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
```

awk komutu ile tek tırnak içerisinde belirlediğimiz fonksiyonun çalışmasını sağlıyoruz.

```
#####  
# Son yeniden başlatma tarihi ve saatini yani son reboot zamanını istemektedir.  
(makineyi kapatmak bunu etkiler)
```

- Son yeniden başlatmanın tarihi ve saati.

```
lb=$(who -b | awk '$1 == "system" {print $3 " " $4}')
```

```
#Last boot: $lb
```

"who", Linux "who(kim)" komutu, UNIX veya Linux işletim sisteminizde oturum açmış olan kullanıcıları görüntülemenizi sağlar. Bir kullanıcının belirli bir Linux tabanlı işletim sistemini kaç kullanıcının kullandığını veya oturum açtığını bilmesi gerektiğinde, bu bilgiyi almak için "who(kim)" komutunu kullanabilir.

```
akaraca@akaraca42:~$ who  
akaraca pts/0 Mar 23 11:41 (10.0.2.2)
```

"who -b" ile sistemin son yeniden başlatma zamanı elde edilir.

```
akaraca@akaraca42:~$ who -b  
system boot Mar 23 11:14
```

```
akaraca@akaraca42:~$ who --help  
Usage: who [OPTION]... [ FILE | ARG1 ARG2 ]  
Print information about users who are currently logged in.  
  
-a, --all same as -b -d --login -p -r -t -T -u  
-b, --boot time of last system boot
```

Elde ettiğimiz bu zaman aralığını yazdırınmak "awk" komutu kullanılır ve system sütunu aranarak zaman yazdırılır.

```
akaraca@akaraca42:~$ who -b | awk '$1 == "system" {print $3 " " $4}'  
Mar 23
```

```
#####  
# lvm (logical volume manager)(mantıksal birim yöneticisi) ile birim ayırma işlemi  
yapıldımı kontrol etmek istenmektedir.
```

- LVM'nin aktif olup olmadığı bilgisi.

```
lvmt=$(lsblk | grep "lvm" | wc -l)  
lvmu=$(if [ $lvmt -eq 0 ]; then echo no; else echo yes; fi)
```

```
#LVM use: $lvmu
```

lsblk(list block) komutu ile tüm veya belirtilen blok aygıtlarıyla ilgili bilgileri listeler.

```
akaraca@akaraca42:~$ lsblk
NAME          MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda            8:0    0 30.8G  0 disk 
| -sda1        8:1    0  476M  0 part /boot
| -sda2        8:2    0   1K  0 part 
` -sda5        8:5    0 30.3G  0 part 
  ` -sda5_crypt 254:0  0 30.3G  0 crypt
    | -LVMGroup-root 254:1  0  9.3G  0 lvm   /
    | -LVMGroup-swap 254:2  0  2.1G  0 lvm   [SWAP]
    | -LVMGroup-home 254:3  0  4.7G  0 lvm   /home
    | -LVMGroup-var  254:4  0  2.8G  0 lvm   /var
    | -LVMGroup-srv  254:5  0  2.8G  0 lvm   /srv
    | -LVMGroup-tmp  254:6  0  2.8G  0 lvm   /tmp
    ` -LVMGroup-var--log 254:7  0  3.7G  0 lvm   /var/log
sr0           11:0   1 1024M  0 rom
```

lvm yapısına sahip kısımların varlığı kontrol edileceğinden dolayı lvm tipi aranır.

```
akaraca@akaraca42:~$ lsblk | grep 'lvm'
| -LVMGroup-root  254:1  0  9.3G  0 lvm   /
| -LVMGroup-swap 254:2  0  2.1G  0 lvm   [SWAP]
| -LVMGroup-home 254:3  0  4.7G  0 lvm   /home
| -LVMGroup-var  254:4  0  2.8G  0 lvm   /var
| -LVMGroup-srv  254:5  0  2.8G  0 lvm   /srv
| -LVMGroup-tmp  254:6  0  2.8G  0 lvm   /tmp
` -LVMGroup-var--log 254:7  0  3.7G  0 lvm   /var/log
```

"wc(wordcount) -l" ile satır sayısını yazdırıyoruz.

```
akaraca@akaraca42:~$ lsblk | grep "lvm" | wc -l
7
```

Eğerki satır sayısı 0'dan farklı ise yes, değilse no çıktısını vermesini istiyoruz.
Tümünü tek satır kodda belirttiğimizden dolayı fazla argumen hatası vermekte.
Bundan dolayı 2 komuta bölmek daha uygundur.

```
akaraca@akaraca42:~$ if [ $(lsblk | grep "lvm" | wc -l) -eq 0 ]; then echo no; else echo yes; fi
-bash: [: too many arguments
yes
```

```
if [ <some test> ]
then
    <commands>
else
    <other commands>
fi
```

Temel bir if deyimi, belirli bir test doğruysa, belirli bir dizi eylemi gerçekleştirin, etkili bir şekilde söyler. Eğer doğru değilse bu işlemleri yapmayın. Bir sonraki işlemi takip eder.

Then ile fi (geriye doğruysa) arasındaki herhangi bir şey, yalnızca test (köşeli parantezler arasında) doğruysa yürütülür.

Bazen bir ifade doğruysa belirli bir dizi eylemi, yanlışsa başka bir dizi eylemi gerçekleştirmek isteriz. Bunu else mekanizması ile karşılayabiliriz.

-eq sayısal bir karşılaştırma yapar [001 -eq 1] true döndürür.

echo karşısında bulunan değerin çıktısını almamıza yarar.

komutlar ; (noktalı virgül) ile ayrılır.

```
#####
```

