

Linux 2.6 ekirdeęi

Stoned Beaver



Murat Ko

Manager & IT Consultant

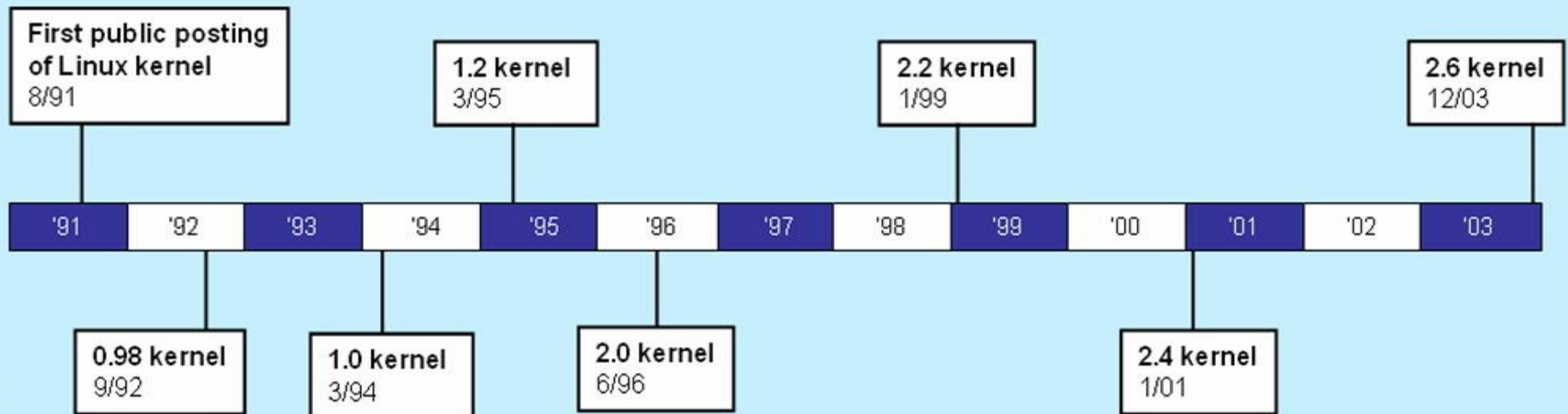
murat.koc@frontsite.com.tr

Başlıklar

- **Tarihçe**
- **Bir Bakışta 2.4 ve 2.6**
- **Geliştirme Süreci Akışı**
- **Çekirdek Bileşenleri**
 - **Yapılandırma Araçları**
 - **Memory Management**
 - **Process Scheduling**
 - **IO Scheduling**
 - **File Systems**
 - **Device Drivers**
 - **Security**
 - **Networking**
 - **Linux Internals**

Tarihçe

Linux Kernel Development Timeline



© OSDL, 2003

Tarihçe

1 Ağustos 1991

v0.01 yayınlandı. Linus Torwald USENET' e duyurdu.

16 Eylül 1992

v0.12 yayınlandı. Linux Çekirdeği GNU GPL lisansı altında lisanslandı.

8 Mart 1992

v0.95 yayınlandı. Harddisk' ten boot etme ve login desteği verildi.

14 Mart 1994

v1.0 yayınlandı. x86 mimaride kararlılık sağlandı.

Mart 1995

v1.2 yayınlandı. Alpha, Sparc ve MIPS mimarileri için destek eklendi. Birçok sürücü ve ağ protokolu için destek eklendi.

6 Haziran 1996

v2.0 yayınladı. 16 CPU' a kadar SMP desteği duyuruldu.

Tarihçe

25 Ocak 1999

v2.2 yayınlandı. PPC ve M68K gibi mimarilerin de olduğu 6 mimariye destek. PCMCIA, USB ve IrDA desteği. NTFS (yalnızca okunabilir), FAT32 ve SMB network paylaşımları için destek. IPv6 ve Software RAID destekleri.

4 Ocak 2001

v2.4 yayınlandı. 2GB dan büyük dosya, 4GB dan fazla RAM desteği. IEEE 1394 desteği. IA-64 ve IBM S/390 mimarileri desteği. LVM, ext3, reiserfs desteği. Networking system nereden ise baştan yazıldı ve ATM, PPPoE destekleri ile daha daha gelişmiş firewall desteği.

17 Aralık 2003

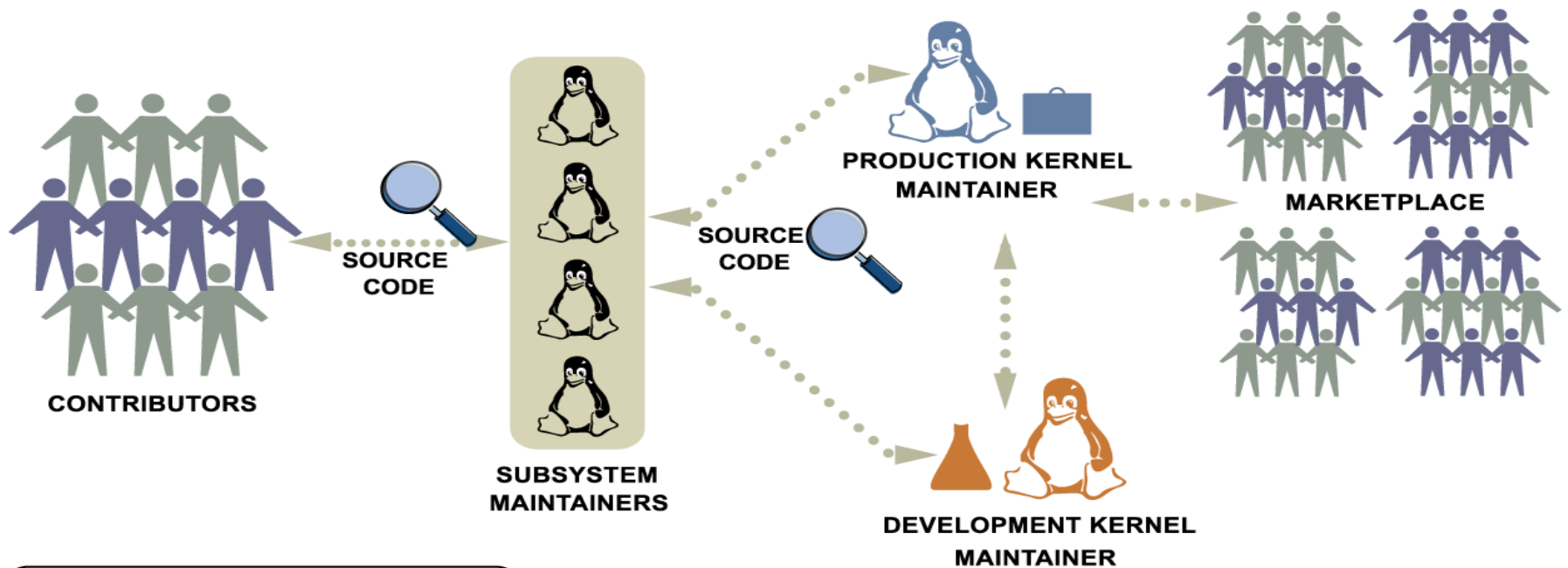
v2.6 yayınlandı.

Bir Bakışta v2.4 ve v2.6

	v2.4	v2.6
Maksimum CPU sayısı	16	64
Maksimum RAM	16	64
Maksimum major aygıt sayısı	255	4095
Maksimum dosya sistemi boyutu	2TB	16TB
Desteklenen dosya sistemleri	Ext2, Ext3, Reiserfs, FFS, HFS, HFS+, FAT, MSDOS, VFAT, ISO9660, JFS, HPFS	Ext2, Ext3, Reiserfs, FFS, HFS, HFS+, FAT, MSDOS, VFAT, ISO9660, JFS, HPFS, NTFS, XFS, CIFS, windows Dynamic Disks
Threading Library	Linux Threads	NPTL
Schedulers	Default Scheduler	O(1) , Anticipatory, Deadline
IPSEC	yok	var
swap dosya sayısı	64	32

Geliştirme Süreci Akışı

LINUX KERNEL DEVELOPMENT PROCESS



Ongoing peer review of code
Continuously available online
for public review

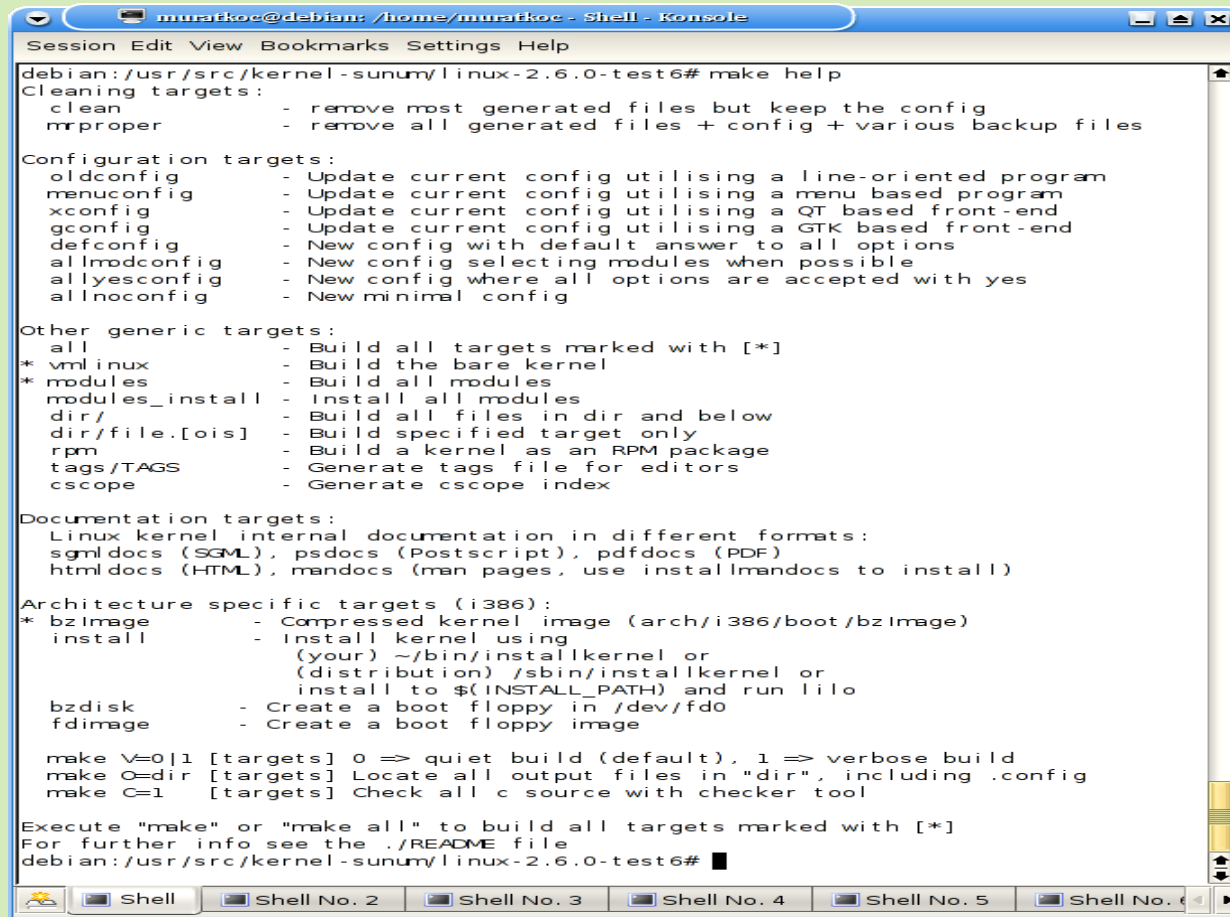
© 2003 Open Source Development Labs

Çekirdek Bileşenleri Yapılandırma Araçları

- ***Kbuild yapılandırma sistemi***
 - ***Gelişmiş yardım menüsü***
 - ***Yeni seçim opsiyonları***
 - ***Daha sade derleme ekranı***
 - ***make xconfig - qconfig***
 - ***make gconfig***

Çekirdek Bileşenleri Yapılandırma Araçları

■ Gelişmiş yardım menüsü



```
muratkoc@debian: /home/muratkoc - Shell - Konsole
Session Edit View Bookmarks Settings Help
debian:/usr/src/kernel-sunum/linux-2.6.0-test6# make help
Cleaning targets:
  clean          - remove most generated files but keep the config
  mrproper       - remove all generated files + config + various backup files

Configuration targets:
  oldconfig      - Update current config utilising a line-oriented program
  menuconfig     - Update current config utilising a menu based program
  xconfig        - Update current config utilising a QT based front-end
  gconfig        - Update current config utilising a GTK based front-end
  defconfig      - New config with default answer to all options
  allmodconfig   - New config selecting modules when possible
  allyesconfig   - New config where all options are accepted with yes
  allnoconfig    - New minimal config

Other generic targets:
  all            - Build all targets marked with [*]
  * vmlinux      - Build the bare kernel
  * modules      - Build all modules
  modules_install - Install all modules
  dir/           - Build all files in dir and below
  dir/file.[ois] - Build specified target only
  rpm            - Build a kernel as an RPM package
  tags/TAGS      - Generate tags file for editors
  cscope         - Generate cscope index

Documentation targets:
Linux kernel internal documentation in different formats:
  sgmldocs (SGML), psdocs (Postscript), pdfdocs (PDF)
  htmldocs (HTML), mandocs (man pages, use installmandocs to install)

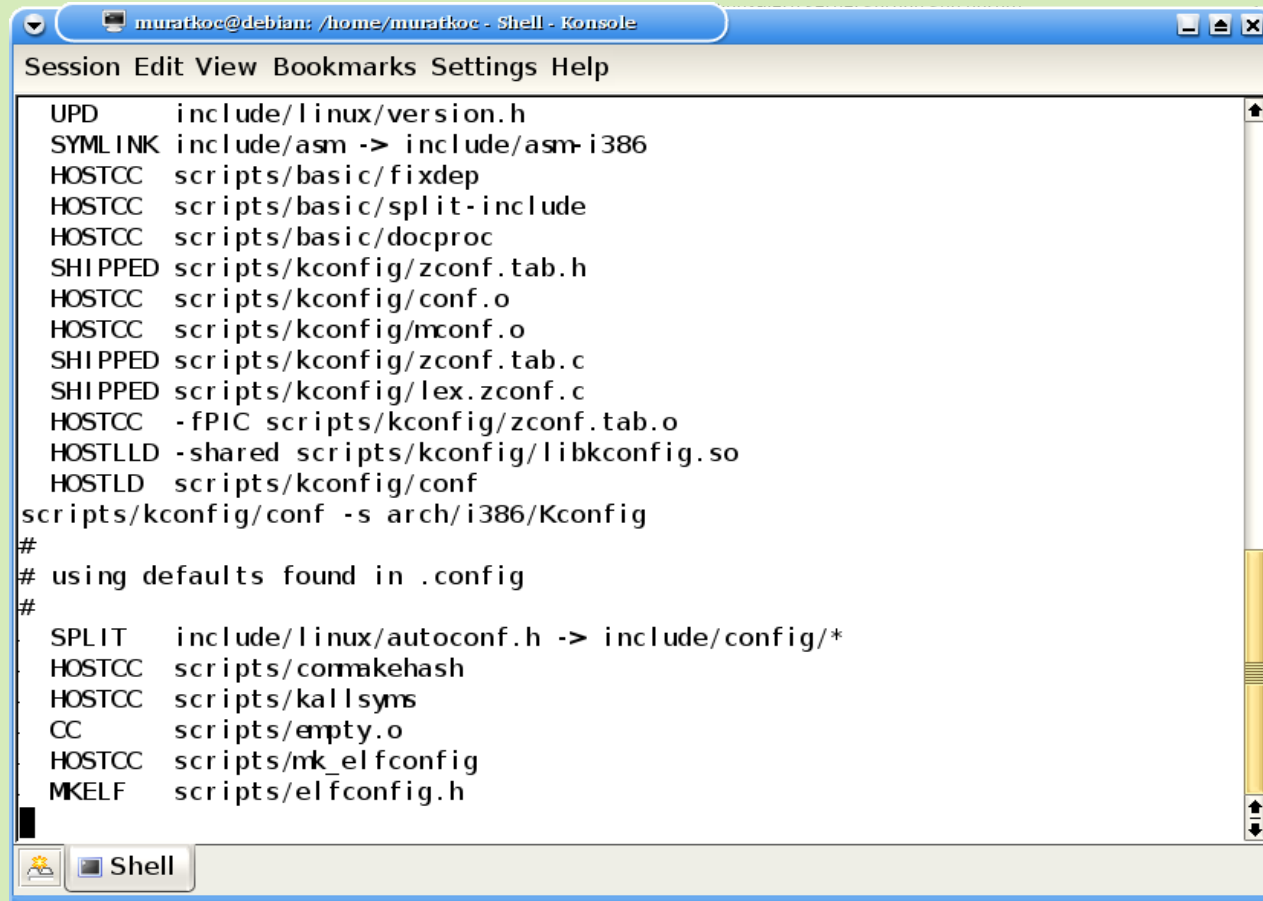
Architecture specific targets (i386):
  * bzimage      - Compressed kernel image (arch/i386/boot/bzImage)
  install        - Install kernel using
                  (your) ~/bin/installkernel or
                  (distribution) /sbin/installkernel or
                  install to $(INSTALL_PATH) and run lilo
  bzdisk         - Create a boot floppy in /dev/fd0
  fdimage        - Create a boot floppy image

make V=0|1 [targets] 0 => quiet build (default), 1 => verbose build
make O=dir [targets] Locate all output files in "dir", including .config
make C=1 [targets] Check all c source with checker tool

Execute "make" or "make all" to build all targets marked with [*]
For further info see the ./README file
debian:/usr/src/kernel-sunum/linux-2.6.0-test6#
```

Çekirdek Bileşenleri Yapılandırma Araçları

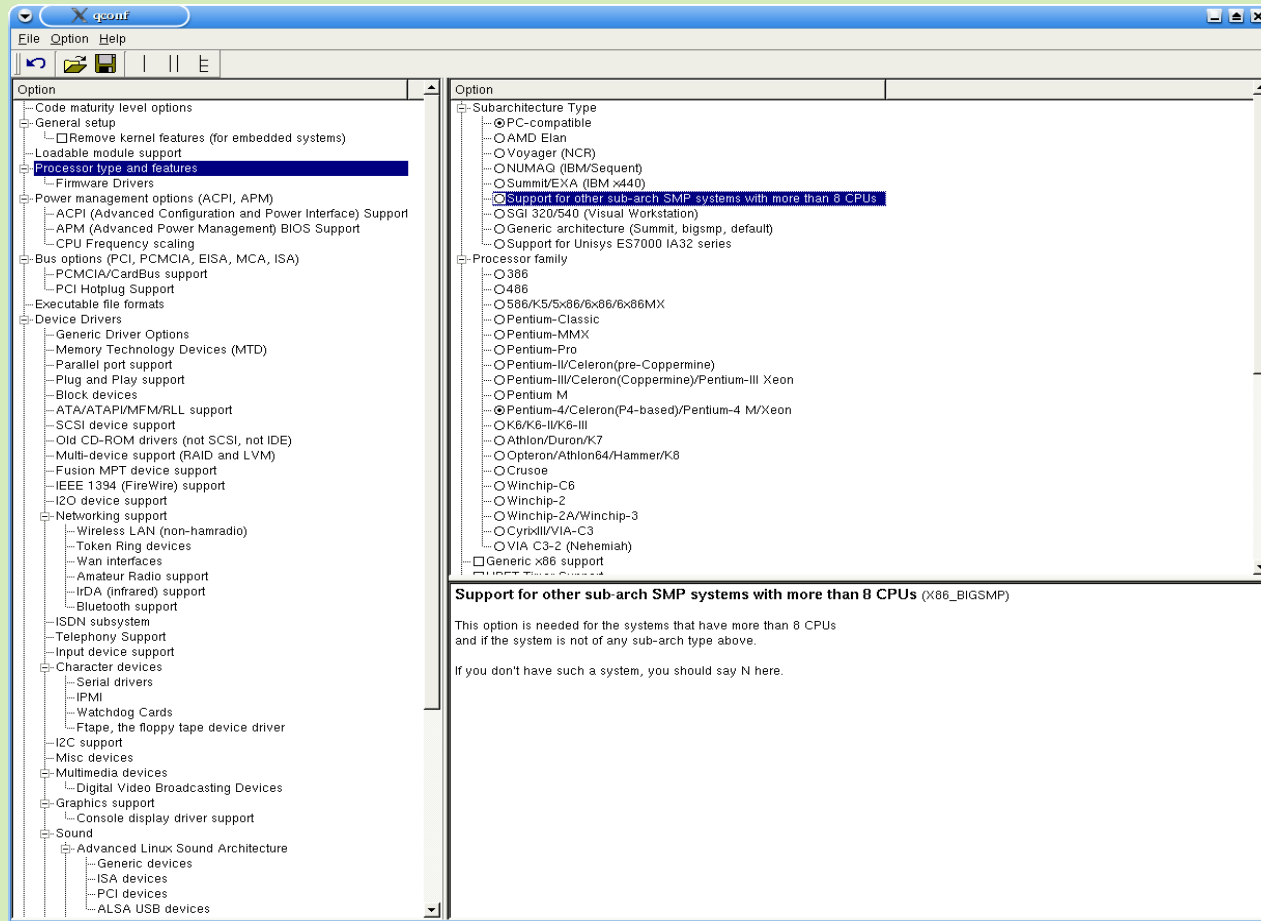
- **Daha sade derleme ekranı**



The screenshot shows a terminal window titled "muratkoc@debian: /home/muratkoc - Shell - Konsole". The window has a menu bar with "Session", "Edit", "View", "Bookmarks", "Settings", and "Help". The terminal output displays the results of a kernel configuration process, listing various components and their corresponding source files and build rules. The output is as follows:

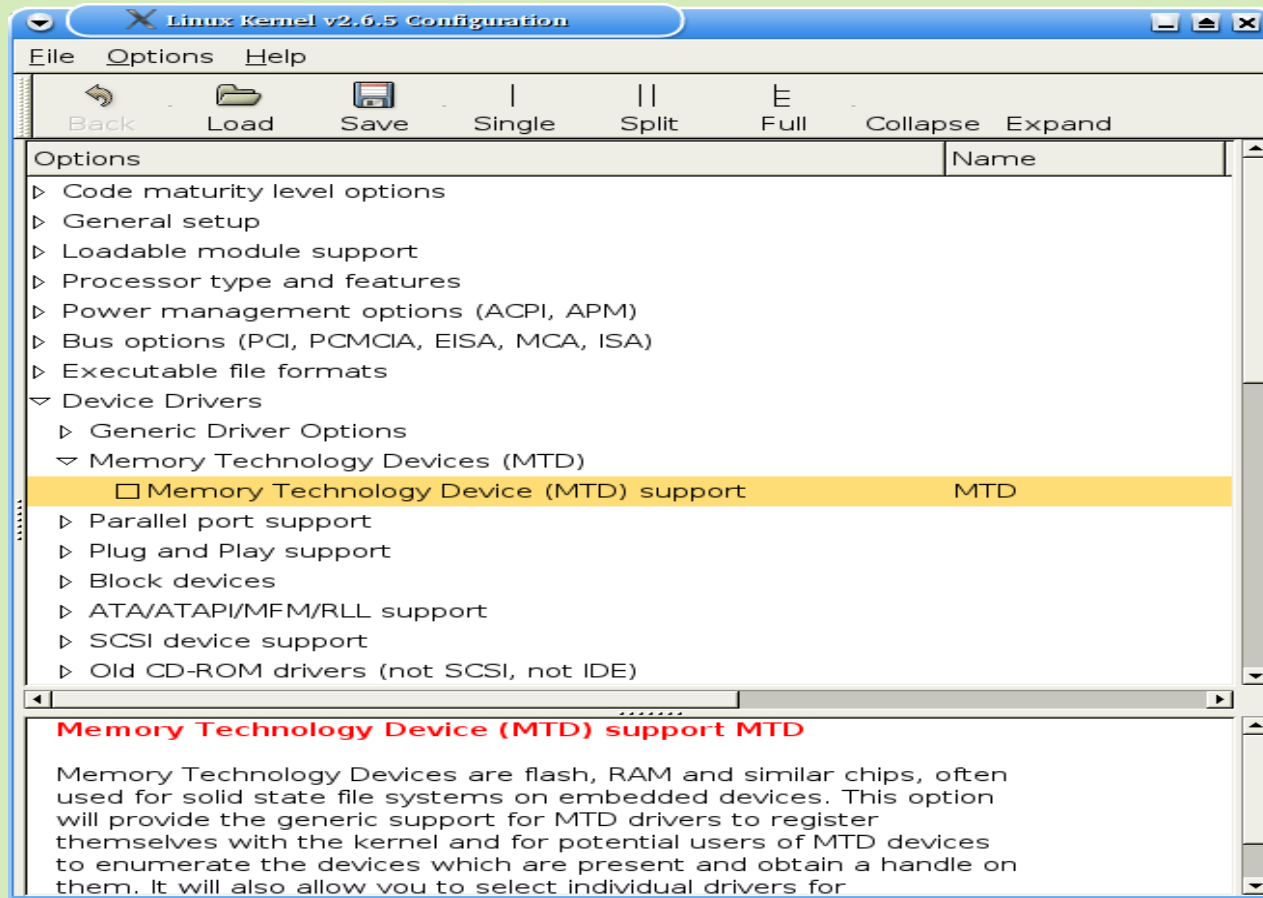
```
UPD      include/linux/version.h
SYMLINK  include/asm -> include/asm-i386
HOSTCC   scripts/basic/fixdep
HOSTCC   scripts/basic/split-include
HOSTCC   scripts/basic/docproc
SHIPPED  scripts/kconfig/zconf.tab.h
HOSTCC   scripts/kconfig/conf.o
HOSTCC   scripts/kconfig/mconf.o
SHIPPED  scripts/kconfig/zconf.tab.c
SHIPPED  scripts/kconfig/lex.zconf.c
HOSTCC   -fPIC scripts/kconfig/zconf.tab.o
HOSTLLD  -shared scripts/kconfig/libkconfig.so
HOSTLD   scripts/kconfig/conf
scripts/kconfig/conf -s arch/i386/Kconfig
#
# using defaults found in .config
#
SPLIT    include/linux/autoconf.h -> include/config/*
HOSTCC   scripts/comakehash
HOSTCC   scripts/kallsyms
CC       scripts/empty.o
HOSTCC   scripts/nk_elfconfig
MKELF    scripts/elfconfig.h
```

 make xconfig - qconfig



Çekirdek Bileşenleri Yapılandırma Araçları

■ **make gconfig**



Çekirdek Bileşenleri

Memory Management

- *Reverse Mapping (rmap)*
- *Daha büyük memory page kullanımı*
- *PTE'lerin High Memory de tutulması*
- *Tekrar tasarlanmış, daha basit algoritmalar*
- *VFS ile daha sıkı entegrasyon*
- *pdflush daemon desteği*
- *swap dosya formatı değişikliği ve hızı*

Çekirdek Bileşenleri

Memory Management

■ **Reverse Mapping (rmap)**

Reverse mapping'ler process page table'lerinin tam tersi olarak düşünülebilir. Bunlar hangi virtual adreste hangi fiziksel adresin hangi process tarafından kullanıldığı bilgisini tutarlar. Bu sayede pageout kodu aşağıdakileri yapabilir;

- **Bütün process'lerin virtual memory alanlarını taramaya gerek kalmaksızın bir page'i bütün processlerden unmap edebilir.**
- **Bütün process'lerin virtual memory alanlarını tarayarak gerekenden daha fazla page'in tahliye edilmesi yerine gerçekten gereken page'leri tahliye ederek page fault sayısının azalmasını sağlar.**
- **Belirli fiziksel adres aralığındaki page'leri tahliye edebilir.**
- **Bilinen inactive page'ler içinde tarama yaparak pageout kodun daha küçük bir alanda çalışmasını sağlar.**

Çekirdek Bileşenleri

Memory Management

- ***Daha büyük memory page kullanımı özellikle büyük veritabanlarının istemiş olacağı çok miktarda memory'nin map edilmesi sırasında oluşacak ekstra yükü azaltmak için kullanılır. Bu sayede örnek olarak 1GB memory map etmek için;***
 - ***262,144 pte yerine 256 pte kullanılır.***
 - ***Sisteme 2MB yerine 2KB yük biner.***
 - ***TLB sayısında azalma sağlayarak performans arttırır.***
- ***PTE'lerin High Memory'de tutulması ile low memory'de tutulması gereken kernel verileri için daha fazla yer açılması sağlanıyor.***

Çekirdek Bileşenleri

Process Scheduling

- ***O(1) Scheduler***
- ***Hyperthreading Scheduler***
- ***Preemption***

Çekirdek Bileşenleri

Process Scheduling – O(1) Scheduler

- **Scheduler dizaynında etkili olan belli başlı faktörler şunlardır;**
 - **Adaletli olma**
Scheduler her process için CPU zamanını dengeli bir şekilde dağıtmalıdır.
 - **Minimum Yük**
Schedulerın kendisi mümkün olan en kısa sürede çalışmalıdır.
 - **Öncelik tabanlı scheduling kullanılması**
Bazı processlerin diğerlerinden daha fazla önceliği olması gerekmektedir.
 - **Turnaround ve bekleme zamanının azaltılması**
queue'da bekleme ve servis süresinin düşürülmesi
 - **Cevap süresi ve dağılımı**
Bir programın cevap süresi mümkün olduğu kadar kısa olmalı ve dağılımı düzgün olmalı

Çekirdek Bileşenleri

Process Scheduling – O(1) Scheduler

- **O(1) Scheduler'in amaçları;**
 - **Yüksek yük altında iyi bir interaktif performans sağlanması**
 - **Adaletli olma**
 - **Öncelikler**
 - **SMP sistemler için etkinlik**
 - **RT Scheduling**
 - **Tamamen O(1) scheduling**
 - **Mükemmel SMP ölçeklenebilirlik**
 - **Her CPU için ayrı locklar ve runqueue lar**
 - **Global locklar ve runqueue'ların kaldırılması**
 - **Bütün operasyonların (wakeup, schedule, context-switching vs) paralel yapılması**
 - **Batch Scheduling**
 - **O(1) RT Scheduling**

Çekirdek Bileşenleri

Process Scheduling – O(1) Scheduler

- **2.6 ve 2.4 scheduler**
 - **2.4 kernel global bir runqueue'ya sahip. Bu nedenle bütün CPU'lar diğer CPU'ların işlerini bitirmesini bekliyor.**
 - **O(n) scheduler**
 - **2.4 kernel'da scheduler global runqueue içinde sonraki task'ı belirlemek için tarama yapıyor. Bu durumda process sayısına bağlı olarak süre uzuyor.**
- **Bunların sonucu olarak yüksek yük altında yeteri kadar verimli çalışamıyor.**

Çekirdek Bileşenleri

Process Scheduling – O(1) Scheduler

- **2.6'daki scheduling politikaları**
 - **İki tane öncelik sıralı öncelik dizisi**
 - **Active dizi: timeslice'ı bulunan tasklar**
 - **Expired dizi: tamamlanmış tasklar**
 - **140 tane öncelik seviyesi bulunuyor.**
 - **1 - 100: RT prio**
 - **101 - 140: User task prio**
 - **Üç farklı scheduling politikası bulunuyor**
 - **Bir tane normal task'lar için**
 - **İki tane RT task'lar için**

Çekirdek Bileşenleri

Process Scheduling – O(1) Scheduler

- **Normal task'lar için;**
 - *Her task'a bir Nice değeri atanıyor*
 - *Her task'a bir timeslice atanıyor*
 - *Aynı öncelikteki task'lar RR*
- **RT task'lar için**
 - **FIFO RT task'lar**
 - *Kendileri CPU'yu serbest bırakana kadar çalışırlar*
 - *Pre-empted değiller*
 - **RR RT Tasklar**
 - *Timeslice atanıyor ve dolana kadar çalışıyorlar.*
 - *Belli seviyedeki öncelik sırasına sahip task'lar timeslice'larını doldurdukları zaman tekrar timeslice veriliyor ve çalışmaya devam ediyorlar.*

Çekirdek Bileşenleri

Process Scheduling – O(1) Scheduler

- **İnteraktiflik tahmini**
 - **Dinamik olarak task'ların öncelikleri interaktifliğine bağlı olarak ölçekleniyor**
 - **İnteraktif task'lar daha fazla öncelik alıyorlar (-5)**
 - **CPU bağlı task'lar daha az öncelik alıyorlar (+5)**
 - **İnteraktiflik tahmini task'ın ortalama uyuma süresinin sabit maksimum uyuma süresine oranı ile karar veriliyor.**
 - **Bunlar RT task'lara uygulanmıyor**
- **Önceliklerin tekrar hesaplanması**
 - **Bir task timeslice'ı bitirdiği zaman**
 - **İnteraktifliği tahmin edilir**
 - **İnteraktif task'lar tekrar Active diziye konulabilir**
 - **Eğer değilse önceliği tekrar hesaplanır**
 - **Yeni öncelik seviyesi ile Expired dizisine konulur**

Çekirdek Bileşenleri

Process Scheduling – O(1) Scheduler

- **Timeslice dağılımı**
 - **Öncelik sadece timeslice tükendikten sonra tekrar hesaplanır**
 - **İnteraktif task'lar geniş timeslice süreleri içinde non-interaktif olabilirler. Bunu önlemek için timeslice'lar 20ms lik parçalara bölünür.**
 - **Aynı seviyede önceliği olan bir task çalışan taskı 20ms de bir preempt edebilir.**
 - **Preempted task tekrar queue'a atılır ve aynı öncelik seviyesindeki tasklar arasından RR olarak çağrılır.**
 - **Öncelik hesaplaması 20ms de bir yapılır.**

Çekirdek Bileşenleri

Process Scheduling – O(1) Scheduler

■ SMP Sistemler

- *Her CPU için ayrı runqueue var. Bundan dolayı her CPU kendi process'lerini işliyor ve diğer CPU'ların işlerini bitirmesini beklemiyor.*
- *Her CPU için bir migration thread çalışıyor*
- *load_balance() fonksiyonu ile runqueue'lar dengesiz olduğu zamanlarda ve belirli zaman aralıklarında çağrılarak bir CPU'dan diğerine task'ların atanması ve CPU kullanımlarının dengelenmesi sağlanır.*
- *Process'ler belirli bir CPU'a daha yakın yapılabilirler.*

Çekirdek Bileşenleri

IO Scheduling

- ***Anticipatory ve Deadline scheduler***

- ***Anticipatory Scheduler***

- ***Temel olarak her process bazlı istatistikler tutarak yapılabilecek olan yeni birbirine bağlı okumaları önceden tahmin etmeye çalışıyor. Bu sayede disk kafasının hareketini minimuma indirerek performans artışı sağlıyor.***
- ***Bir okuma isteği bittiği zaman sonraki isteği hemen işleme almıyor. 6ms bekleyerek uygulamanın başka bir okuma isteğinde bulunup bulunmayacağına bakar. Eğer varsa bu isteği işleme alır.***

Çekirdek Bileşenleri

IO Scheduling

- **Anticipatory ve Deadline scheduler**

- **Deadline Scheduler**

- İşlemlere bir sonlanma zamanı atıyor
- İki yeni queue eklendi. Bunlar;
 - 500ms deadline a sahip olan FIFO read queue
 - 5s deadline a sahip olan FIFO write queue
- Sıralı queue'a gelen istek uygun olan queue'nun sonuna atılıyor
- Eğer FIFO queue'daki istek zaman aşımına uğramışsa scheduler FIFO queue'dan çıkarıyor
- Aynı zamanda isteklerin cevapsız kalmamasına çalışıyor

Çekirdek Bileşenleri

Dosya Sistemleri

- **Genel VFS değişiklikleri**
 - *Atomic olarak bir subtree başka bir yere taşınabilir hale getirildi.*
 - *Dizinler artık senkron olarak tanımlanabiliyor. Bu sayede yapılan değişiklikler hemen diske yazılmış oluyor.*
- **Yeni eklenen destekler ile aşağıdaki dosya sistemleri kullanılabilir;**
 - *ext2, ext3, reiserfs, xfs, minix, romfs, iso9660, udf, msdos, vfat, ntfs, adfs, amiga ffs, apple macintosh hfs, BeOS befs, bfs, efs, cramfs, free vxfs, os/2 hpfs, qnx4fs, sysvfs, ufs*
- **Önemli dosya sistemlerinde yapılan değişiklikler;**
 - **EXT3**
 - *Indexlenmiş dizin yapısı geliştirildi. Bu sayede performans artışı sağlandı.*
 - *Orlov allocator geliştirildi. Bu sayede altdizinler disk üstünde yakın yerleştirilerek performans artışı sağlandı.*
 - **Reiserfs**
 - *Artık inode attribute'ları destekliyor.*
 - *4KB dan büyük boyutlarda yazma imkanı.*
 - *Değişken block boyutları*

Çekirdek Bileşenleri

Dosya Sistemleri

- **NFS**

- ***NFSv4 desteği eklendi***
- ***TCP transport desteği eklendi***

- **NTFS**

- ***SMP desteği***
- ***4KB tan büyük cluster boyutu desteği***
- ***dosya boyutunda değişiklik yapılmaksızın yazma desteği***

- **POSIX ACL desteği**

Çekirdek Bileşenleri

Device Drivers

- **PCI**
 - *PCI Domain desteği eklendi. Bu sayede büyük sistemlerdeki kullanıcılar bütün PCI aygıtlarına ulaşabilir oldular.*
 - *Sahte bir PCI hotplug aygıt desteği verildi*
- **Input Layer**
 - *En fazla değişikliğin görüldüğü alanlardan bir tanesi. Bütün klavye, mouse veya diğer input aygıtları bir sistem altında toplandı ve kullanıcıya bunlarla ilgili daha fazla esneklik sağlandı.*
- **ALSA**
 - *Advanced Linux Sound Architecture eklendi. İlerde tamamen OSS nin yerini alacak.*
- **AGP**
 - *AGP 3.0 desteği eklendi*
- **Framebuffer**
 - *Framebuffer ve console layer yaklaşık olarak baştan yazıldı. Birçok sürücüde iyileştirmeler sağlandı.*

Çekirdek Bileşenleri

Device Drivers

■ IDE

- *IDE TCQ desteği eklendi.*
- *ATAPI aygıtlar için cd yazarken DMA kullanma desteği eklendi. Bu sayede ide-scsi kullanmaksızın IDE CD yazıcılar kullanılabilir oldu.*

■ USB

- *USB Host Controller'ların isimleri değişti.*
- *USB storage'da performans artışları sağlandı*
- *USB Gadget API desteği eklendi. Bu sayede sürücü eklenmesi daha kolaylaştırıldı.*

■ SCSI

- *2.4 serisinde 128 veya 256 tane olan disk desteği binlerce disk desteğine çıkartıldı.*
- *Block Layer (bio) için destek sağlandı*
- *Her HBA için lock yapısı geliştirildi.*
- *Hotplug desteği arttırıldı.*

Çekirdek Bileşenleri

Device Drivers

- **LVM2 - Devicemapper**

- **LVM1 kaldırıldı. Fakat LVM1 disk formatına uygunluk devam ettirildi.**
- **Daha kararlı ve güvenli olan dm-crypt sistemi geliştirildi**

- **SBF**

- **Simple Boot Flag desteği verilerek daha hızlı boot işlemi yapılması sağlandı**

Çekirdek Bileşenleri

Güvenlik

- ***NSA SELinux eklendi***
- ***Root Plug desteği verildi***
- ***Genel bir Crypto API geliştirildi***

Çekirdek Bileşenleri

Networking

- ***Ebtables***
- ***IPSec***
- ***IP Payload Compression***
- ***IPv6 Privacy Extensions***
- ***SCTP***

Çekirdek Bileşenleri

Linux Internals

- ***NPTL***
- ***Futex***
- ***epoll***
- ***Daha hızlı system çağrıları***
- ***AIO***
- ***Güç yönetimi***
- ***Profile desteği***

Sorular ?

Teşekkürler....