



**LABORATORIUM JARINGAN KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA**

PRAKTIKUM SISTEM OPERASI

SEMESTER : GENAP

TAHUN : 2015/2016

BAB I

JUDUL BAB : STRUKTUR SISTEM OPERASI
DISUSUN OLEH : MOH ARIF ANDRIAN
NIM : 156150600111002
ASISTEN : SISKI PERMATASARI
ZAENAL KURNIAWAN
KOORDINATOR ASISTEN : DANY RAHMANA



LAPORAN PRAKTIKUM SISTEM OPERASI FAKULTAS ILMU KOMPUTER UNIVERSITAS BRAWIJAYA

Nama : Moh. Arif Andrian
NIM : 156150600111002
Laporan : BAB I
Asisten : Siska Permatasari
Zaenal Kurniawan

BAB I STRUKTUR SISTEM OPERASI

1. Dasar Teori

Sebuah sistem yang besar dan kompleks seperti sistem operasi modern harus diatur dengan cara membagi tugas ke dalam beberapa komponen-komponen kecil agar dapat berfungsi dengan baik dan mudah dimodifikasi. Ada beberapa jenis struktur pembagian komponen sistem operasi antara lain :

- a. Struktur Sederhana
- b. Struktur Lapisan
- c. Struktur Monolitik
- d. Struktur Microkernel
- e. Struktur Modular

Struktur Sederhana

Ada sejumlah sistem komersial yang tidak memiliki struktur yang cukup baik. Sistem operasi tersebut sangat kecil, sederhana dan memiliki banyak keterbatasan. Salah satu contoh sistem tersebut adalah MS-DOS. MS-DOS dirancang oleh orang-orang yang tidak memikirkan akan kepopuleran software tersebut. Sistem operasi tersebut terbatas pada perangkat keras sehingga tidak terbagi menjadi modul-modul. Meskipun MS-DOS mempunyai beberapa struktur, antar muka dan tingkatan fungsionalitas tidak terpisah secara baik.

Struktur Monolitik

Pada perkembangannya, mulai terjadi pemisahan struktur sistem operasi yaitu antara bagian kernel dan program sistem. Kernel berada di bawah tingkat antarmuka system call dan diatas perangkat lunak secara fisik. Kernel ini berisi sistem file, penjadwalan CPU, manajemen memori, dan fungsi sistem operasi lainnya yang ada pada sistem call berupa sejumlah fungsi yang besar pada satu level. Program sistem meminta bantuan kernel untuk memanggil fungsi-fungsi dalam kompilasi dan manipulasi file. Salah satu contoh sistem operasi dengan struktur monolitik adalah sistem operasi UNIX generasi awal.

Struktur Lapisan

Sistem operasi dibagi menjadi sejumlah lapisan yang masing-masing dibangun di atas lapisan yang lebih rendah. Lapisan yang lebih rendah menyediakan layanan

untuk lapisan yang lebih tinggi. Lapisan yang paling bawah adalah perangkat keras, dan yang paling tinggi adalah user-interface.

Struktur Microkernel

Metode ini menyusun sistem operasi dengan mengeluarkan semua komponen yang tidak esensial dari kernel, dan mengimplementasikannya sebagai program sistem dan level pengguna. Hasilnya kernel yang lebih kecil. Pada umumnya mikrokernel mendukung proses dan manajemen memori yang minimal, sebagai tambahan untuk fasilitas komunikasi.

Struktur Modular

Pada struktur modular, kernel hanya menyediakan layanan inti dari sistem operasi sedangkan layanan lain diimplementasikan secara dinamis ketika kernel sedang berjalan. Sebagai contoh, layanan inti seperti penjadwalan CPU dan manajemen memori ditangani langsung oleh kernel, sedangkan layanan-layanan yang lain seperti dukungan untuk beragam jenis file system dilakukan pada modul-modul yang dapat dipasang dan dilepas saat kernel utama berjalan. Modul-modul tersebut dikenal dengan istilah *loadable kernel module*.

Salah satu contoh sistem operasi yang memakai pendekatan struktur modular adalah Linux. Salah satu contoh kernel module di Linux adalah modul `ath3k`, `ath9k` yang berfungsi sebagai device driver untuk perangkat wifi dengan chipset Atheros. Pada bagian selanjutnya, akan disajikan contoh kode kernel module sederhana dalam bahasa C beserta cara kompilasi dan pemasangannya.

2. Materi Praktikum

Pada bagian ini akan dijelaskan mengenai tahapan kompilasi dan pemasangan sebuah kernel module sederhana.

1. Sebelum melakukan kompilasi kernel module, terlebih dahulu kita harus memasang paket yang dibutuhkan antara lain: library header sesuai versi kernel Linux, gcc (GNU C Compiler) dan make dengan perintah:

```
sudo apt-get install linux-headers-generic make gcc
```

```
andrian@156150600111002:~  
andrian@156150600111002:~$ sudo apt-get install linux-headers-generic make gcc  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
gcc is already the newest version (4:5.3.1-1ubuntu1).  
make is already the newest version (4.1-6).  
make set to manually installed.  
The following additional packages will be installed:  
  linux-generic linux-headers-4.4.0-22 linux-headers-4.4.0-22-generic  
  linux-image-4.4.0-22-generic linux-image-extra-4.4.0-22-generic linux-image-generic  
Suggested packages:  
  fdutils linux-doc-4.4.0 | linux-source-4.4.0 linux-tools  
The following NEW packages will be installed:  
  linux-headers-4.4.0-22 linux-headers-4.4.0-22-generic linux-image-4.4.0-22-generic  
  linux-image-extra-4.4.0-22-generic  
The following packages will be upgraded:  
  linux-generic linux-headers-generic linux-image-generic  
3 upgraded, 4 newly installed, 0 to remove and 15 not upgraded.  
Need to get 68,3 MB of archives.  
After this operation, 295 MB of additional disk space will be used.  
Do you want to continue? [Y/n] y  
Get:1 http://us.archive.ubuntu.com/ubuntu xenial-proposed/main amd64 linux-image-4.4.0-  
22-generic amd64 4.4.0-22.38 [18,7 MB]  
4% [1 linux-image-4.4.0-22-generic 3.642 kB/18,7 MB 19%] 68,8 kB/s 15min 39s  
  
andrian@156150600111002:~  
andrian@156150600111002:~$ sudo apt-get install linux-headers-generic make gcc  
[sudo] password for andrian:  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
gcc is already the newest version (4:5.3.1-1ubuntu1).  
make is already the newest version (4.1-6).  
linux-headers-generic is already the newest version (4.4.0.22.23).  
0 upgraded, 0 newly installed, 0 to remove and 15 not upgraded.  
andrian@156150600111002:~$
```

2. Buat file **hello.c** dengan editor favorit anda dengan isi sebagai berikut

```
hello.c  
#include <linux/module.h>      /* Needed by all modules */  
#include <linux/kernel.h>     /* Needed for KERN_INFO */  
#include <linux/init.h>       /* Needed for the macros */  
  
static int __init hello_start(void)  
{  
    printk(KERN_INFO "Loading hello module...\n");  
    printk(KERN_INFO "Hello world Mr. Praktikan\n");  
    return 0;  
}  
  
static void __exit hello_end(void)  
{  
    printk(KERN_INFO "Goodbye Mr. Praktikan\n");  
}  
  
module_init(hello_start);  
module_exit(hello_end);
```

```
andrian@156150600111002: ~
GNU nano 2.5.3 File: hello.c Modified
#include <linux/module.h> /* Needed by all modules */
#include <linux/kernel.h> /* Needed for KERN_INFO */
#include <linux/init.h> /* Needed for the macros */
static int __init hello_start(void)
{
    printk(KERN_INFO "Loading hello module...\n");
    printk(KERN_INFO "Hello world Mr. Praktikan\n");
    return 0;
}
static void __exit hello_end(void)
{
    printk(KERN_INFO "Goodbye Mr. Praktikan\n");
}
module_init(hello_start);
module_exit(hello_end);

```

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line

Penjelasan source code:

Pada kode hello.c tersebut terdapat dua fungsi callback yang dipanggil yaitu **module_init()** dan **module_exit()**. Fungsi callback **module_init()** akan dipanggil ketika kernel module dipasang. Selanjutnya, fungsi **hello_start()** sebagai parameter dari **module_init()** akan dieksekusi. Pada **hello_start()** dilakukan pencetakan string dengan fungsi **printk()**. Prosedur yang sama berlaku untuk fungsi **module_exit()** yang dipanggil ketika sebuah kernel module dilepas.

3. Buat sebuah file bernama **Makefile** di direktori yang sama dengan **hello.c**. **Makefile** berisi informasi source code mana yang akan dikompilasi, lokasi library yang dibutuhkan dan juga output dari proses kompilasi. Berikut isi Makefile

```
Makefile
obj-m = hello.o
KVERSION = $(shell uname -r)
all:
    make -C /lib/modules/$(KVERSION)/build M=$(PWD) modules
clean:
    make -C /lib/modules/$(KVERSION)/build M=$(PWD) clean
```

```
andrian@156150600111002: ~/Desktop
GNU nano 2.5.3 File: Makefile

obj-m = hello.o
KVERSION = $(shell uname -r)
all:
    make -C /lib/modules/$$(KVERSION)/build M=$(PWD) modules
clean:
    make -C /lib/modules/$$(KVERSION)/build M=$(PWD) clean

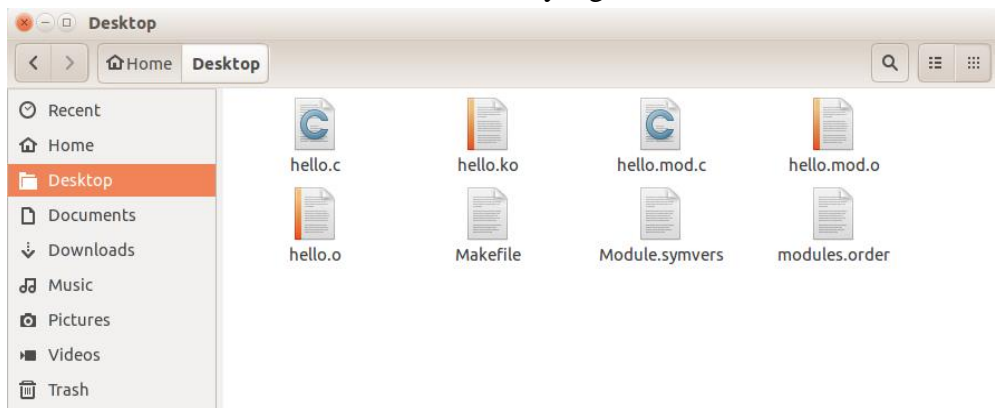
^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text ^T To Spell   ^_ Go To Line
```

4. Compile source code dengan perintah

```
make
```

```
andrian@156150600111002: ~/Desktop
andrian@156150600111002:~/Desktop$ make
make -C /lib/modules/4.4.0-21-generic/build M=/home/andrian/Desktop modules
make[1]: Entering directory '/usr/src/linux-headers-4.4.0-21-generic'
CC [M] /home/andrian/Desktop/hello.o
Building modules, stage 2.
MODPOST 1 modules
CC /home/andrian/Desktop/hello.mod.o
LD [M] /home/andrian/Desktop/hello.ko
make[1]: Leaving directory '/usr/src/linux-headers-4.4.0-21-generic'
```

5. Pada direktori tersebut akan muncul file baru yaitu **hello.ko**. File tersebut adalah file executable dari kernel module yang kita buat.



```
andrian@156150600111002: ~/Desktop
andrian@156150600111002:~/Desktop$ ls
hello.c  hello.mod.c  hello.o  modules.order
hello.ko  hello.mod.o  Makefile  Module.symvers
andrian@156150600111002:~/Desktop$
```

6. Pasang kernel module tadi dengan perintah

```
sudo insmod hello.ko
```

```
andrian@156150600111002: ~/Desktop
andrian@156150600111002:~/Desktop$ sudo insmod hello.ko
[sudo] password for andrian:
andrian@156150600111002:~/Desktop$ ls
```

7. Cek apakah kernel module berhasil dipasang dengan perintah lsmod. Jika sudah terpasang dengan benar, maka nama kernel module hello akan ada pada daftar modul yang terpasang.

```
andrian@156150600111002: ~/Desktop
andrian@156150600111002:~/Desktop$ lsmod
Module                Size  Used by
hello                 16384  0
btrfs                 987136 0
```

8. Cek output dari kernel module setelah dipasang dengan perintah dmesg.

```
andrian@156150600111002: ~/Desktop
[ 4669.927247] hello: module license 'unspecified' taints kernel.
[ 4669.927257] Disabling lock debugging due to kernel taint
[ 4669.927344] hello: module verification failed: signature and/or required key missing
- tainting kernel
[ 4669.927901] Loading hello module...
[ 4669.927904] Hello world Mr. Praktikan
andrian@156150600111002:~/Desktop$
```

9. Untuk melepas kernel module hello yang sudah terpasang tadi, kita dapat memakai perintah

```
sudo rmmod hello
```

```
andrian@156150600111002: ~/Desktop
andrian@156150600111002:~/Desktop$ sudo rmmod hello
andrian@156150600111002:~/Desktop$
```

10. Cek output dari kernel module setelah dilepas dengan perintah dmesg.

```
andrian@156150600111002: ~/Desktop
[ 4669.927247] hello: module license 'unspecified' taints kernel.
[ 4669.927257] Disabling lock debugging due to kernel taint
[ 4669.927344] hello: module verification failed: signature and/or required key missing
- tainting kernel
[ 4669.927901] Loading hello module...
[ 4669.927904] Hello world Mr. Praktikan
[ 5594.849124] Goodbye Mr. Praktikan
andrian@156150600111002:~/Desktop$
```



LAPORAN PRAKTIKUM SISTEM OPERASI

FAKULTAS ILMU KOMPUTER

UNIVERSITAS BRAWIJAYA

Nama : Moh. Arif Andrian
NIM : 156150600111002
Tugas : BAB I
Asisten : Siska Permatasari
Zaenal Kurniawan

TUGAS PRAKTIKUM

1. Download the Latest Stable Kernel

```
andrian@156150600111002: ~/Downloads
andrian@156150600111002:~$ cd Downloads
andrian@156150600111002:~/Downloads$ ls
BAB VI.docx          linux-4.5.2.tar.xz
Firefox_wallpaper.png opera-11.00-1156.x86_64.linux
andrian@156150600111002:~/Downloads$
```

2. extract kernel source

```
andrian@156150600111002: ~/Downloads
andrian@156150600111002:~/Downloads$ tar -xf linux-4.5.2.tar.xz
andrian@156150600111002:~/Downloads$ ls
BAB VI.docx          linux-4.5.2.tar.xz
Firefox_wallpaper.png opera-11.00-1156.x86_64.linux
linux-4.5.2
andrian@156150600111002:~/Downloads$
```

3. configure kernel

- sebelum melakukan konfigurasi pindahkan hasil extract ke direktory /usr/src/

```
root@156150600111002: /usr/src
andrian@156150600111002:~/Downloads$ sudo su
[sudo] password for andrian:
root@156150600111002:/home/andrian/Downloads# ls
BAB VI.docx          linux-4.5.2.tar.xz
Firefox_wallpaper.png opera-11.00-1156.x86_64.linux
root@156150600111002:/home/andrian/Downloads# mv linux-4.5.2 /usr/src/
root@156150600111002:/home/andrian/Downloads# cd /usr/src/
root@156150600111002:/usr/src# ls
linux-4.5.2          linux-headers-4.2.0-35-generic
linux-headers-4.2.0-30      linux-headers-4.4.0-21
linux-headers-4.2.0-30-generic  linux-headers-4.4.0-21-generic
linux-headers-4.2.0-34      linux-headers-4.4.0-22
linux-headers-4.2.0-34-generic  linux-headers-4.4.0-22-generic
linux-headers-4.2.0-35
root@156150600111002:/usr/src#
```

- Install paket dependensi untuk dukungan kompilasi nantinya.

```
Apt-get install aptitude
```



```
root@156150600111002: /usr/src
root@156150600111002:/usr/src# apt-get install aptitude
Reading package lists... Done
Building dependency tree
Reading state information... Done
aptitude is already the newest version (0.7.4-2ubuntu2).
0 upgraded, 0 newly installed, 0 to remove and 15 not upgraded.
root@156150600111002:/usr/src#
```

aptitude install build-essential qt4-dev-tools ncurses-dev

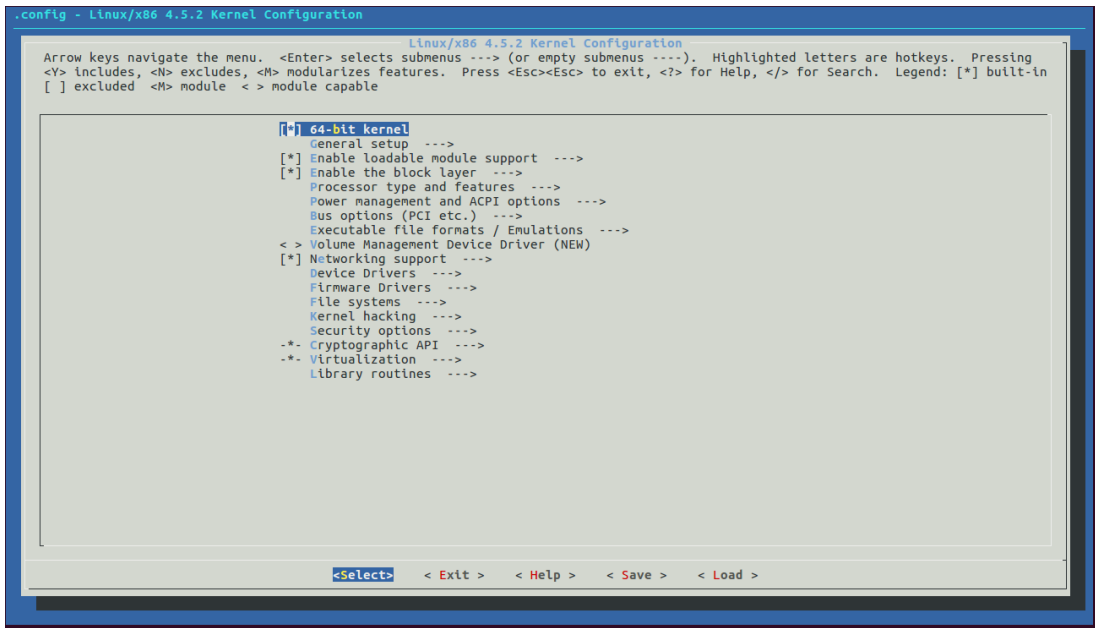
```
root@156150600111002: /usr/src
root@156150600111002:/usr/src# aptitude install build-essential qt4-dev-t
ools ncurses-dev
build-essential is already installed at the requested version (12.1ubuntu
2)
qt4-dev-tools is already installed at the requested version (4:4.8.7+dfsg
-5ubuntu2)
build-essential is already installed at the requested version (12.1ubuntu
2)
qt4-dev-tools is already installed at the requested version (4:4.8.7+dfsg
-5ubuntu2)
No packages will be installed, upgraded, or removed.
0 packages upgraded, 0 newly installed, 0 to remove and 15 not upgraded.
Need to get 0 B of archives. After unpacking 0 B will be used.
root@156150600111002:/usr/src#
```

- masuk pada direktori kernel yang akan di konfigurasi.

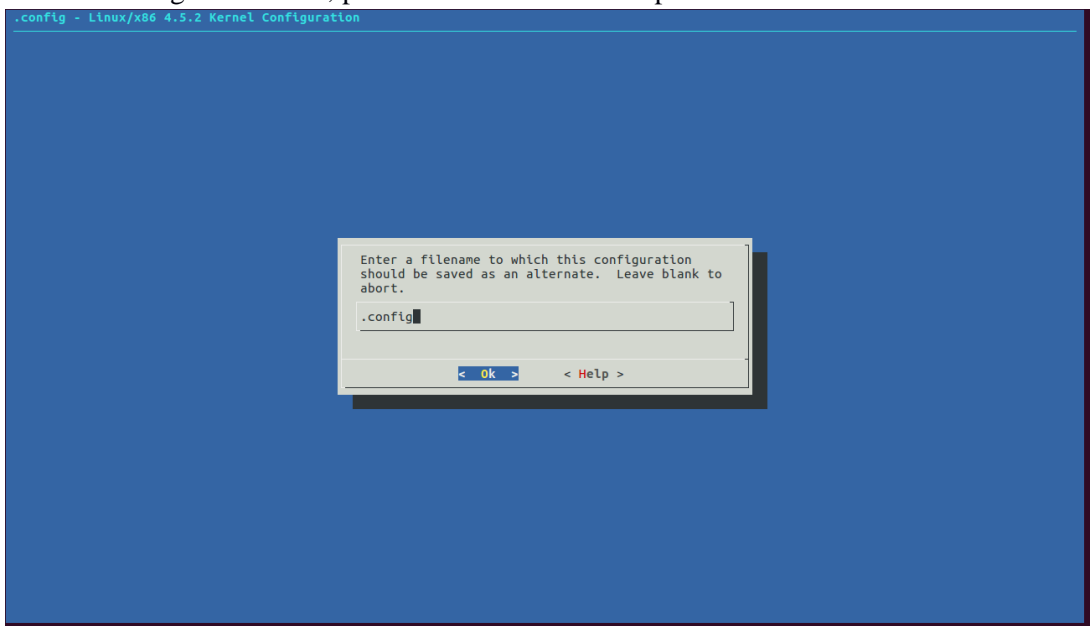
```
root@156150600111002: /usr/src/linux-4.5.2
root@156150600111002:/usr/src# ls
linux-4.5.2          linux-headers-4.2.0-35-generic
linux-headers-4.2.0-30      linux-headers-4.4.0-21
linux-headers-4.2.0-30-generic  linux-headers-4.4.0-21-generic
linux-headers-4.2.0-34      linux-headers-4.4.0-22
linux-headers-4.2.0-34-generic  linux-headers-4.4.0-22-generic
linux-headers-4.2.0-35
root@156150600111002:/usr/src# cd linux-4.5.2
root@156150600111002:/usr/src/linux-4.5.2#
```

- konfigurasi kernel.

make menuconfig



setelah konfigurasi selesai, pilih save -> beri nama -> pilih OK.



+ Tip untuk konfigurasi kernel

- Selalu aktifkan "Prompt for development... drivers".
- Sejak kernel 2.6.8, kita dapat menambahkan string sendiri (seperti inisial anda) di "Local version - append to kernel release" untuk personalisasi string versi kernel (untuk kernel yang lama, kita harus mengedit kalimat EXTRAVERSION di Makefile).
- Selalu matikan module versioning, tapi selalu nyalakan kernel module loader, kernel modules dan module unloading.
- Di bagian kernel hacking, nyalakan semua opsi kecuali "Use 4Kb for kernel stacks instead of 8Kb". Jika kita menggunakan mesin yang pelan, jangan nyalakan "Debug memory allocations".

- Matikan fitur yang tidak kita butuhkan.
- Hanya gunakan module, jika kita mempunyai alasan yang kuat untuk menggunakan module. Contoh, kita sedang mengerjakan sebuah driver, kita ingin me-load versi baru tanpa rebooting.
- Pastikan secara teliti kita memilih tipe processor yang benar. Cari tahu jenis processor yang kita pakai menggunakan perintah
cat /proc/cpuinfo.
- Cari tahu PCI device apa yang kita install, gunakan perintah
lspci -v.
- Cari tahu kernel yang kita buat apakah sudah di compile dengan baik menggunakan perintah

dmesg | less.

4. Compile the Linux Kernel

Untuk melakukan proses compile, gunakan perintah :

make

```

root@156150600111002: /usr/src/linux-4.5.2
IHEX    firmware/tigon/tg3.bin
IHEX    firmware/tigon/tg3_tso.bin
IHEX    firmware/tigon/tg3_tso5.bin
IHEX    firmware/3com/typhoon.bin
IHEX2FW firmware/emi26/loader.fw
IHEX2FW firmware/emi26/firmware.fw
IHEX2FW firmware/emi26/bitstream.fw
IHEX2FW firmware/emi62/loader.fw
IHEX2FW firmware/emi62/bitstream.fw
IHEX2FW firmware/emi62/spdif.fw
IHEX2FW firmware/emi62/midi.fw
IHEX    firmware/kaweth/new_code.bin
IHEX    firmware/kaweth/trigger_code.bin
IHEX    firmware/kaweth/new_code_fix.bin
IHEX    firmware/kaweth/trigger_code_fix.bin
IHEX    firmware/ti_3410.fw
IHEX    firmware/ti_5052.fw
IHEX    firmware/mts_cdma.fw
IHEX    firmware/mts_gsm.fw
IHEX    firmware/mts_edge.fw
H16T0FW firmware/edgeport/boot.fw
H16T0FW firmware/edgeport/boot2.fw
H16T0FW firmware/edgeport/down.fw
H16T0FW firmware/edgeport/down2.fw
IHEX    firmware/edgeport/down3.bin
IHEX2FW firmware/whiteheat_loader.fw
IHEX2FW firmware/whiteheat.fw
IHEX2FW firmware/keyspan_pda/keyspan_pda.fw
IHEX2FW firmware/keyspan_pda/xircom_pgs.fw
IHEX    firmware/cpia2/stv0672_vp4.bin
IHEX    firmware/yam/1200.bin
IHEX    firmware/yam/9600.bin
root@156150600111002: /usr/src/linux-4.5.2#

```

Catatan : Proses ini akan memakan waktu yang lumayan lama.

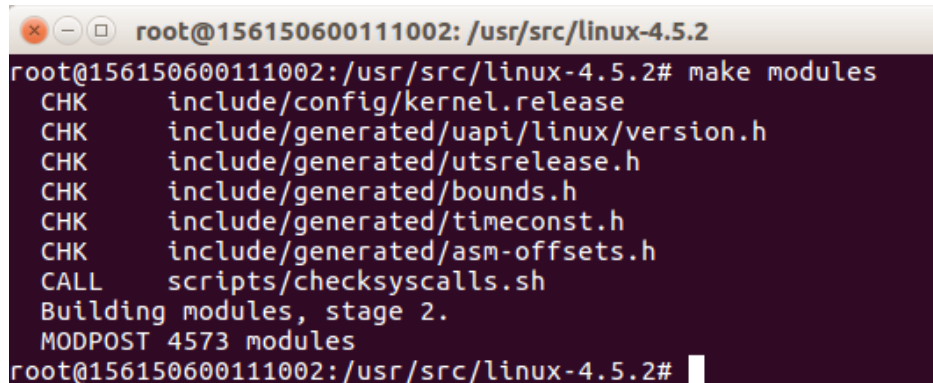
Apabila terjadi kesalahan pada saat proses compile, gunakan syntax berikut untuk memperbaikinya.

```
# apt-get install libssl-dev
```

5. Install the New Kernel

- Membuat paket modul

```
# make modules
```



```
root@156150600111002: /usr/src/linux-4.5.2
root@156150600111002: /usr/src/linux-4.5.2# make modules
CHK      include/config/kernel.release
CHK      include/generated/uapi/linux/version.h
CHK      include/generated/utsrelease.h
CHK      include/generated/bounds.h
CHK      include/generated/timeconst.h
CHK      include/generated/asm-offsets.h
CALL     scripts/checksyscalls.sh
Building modules, stage 2.
MODPOST 4573 modules
root@156150600111002: /usr/src/linux-4.5.2#
```

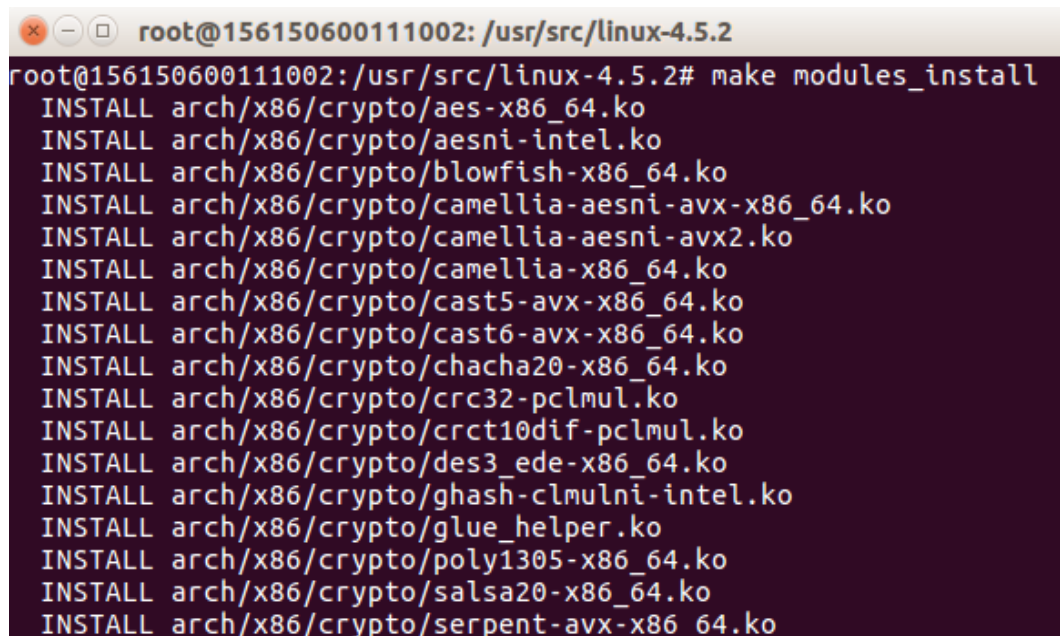
Jika terjadi error, gunakan perintah berikut :

```
# make CONFIG_DEBUG_SECTION_MISMATCH=y
```

Perintah ini akan melakukan re-compile terhadap paket-paket error yang sudah dikompilasi sebelumnya, ini akan memakan waktu yang lumayan lama juga.

- Menginstall modul-modul kernel

```
# make modules_install
```



```
root@156150600111002: /usr/src/linux-4.5.2
root@156150600111002: /usr/src/linux-4.5.2# make modules_install
INSTALL arch/x86/crypto/aes-x86_64.ko
INSTALL arch/x86/crypto/aesni-intel.ko
INSTALL arch/x86/crypto/blowfish-x86_64.ko
INSTALL arch/x86/crypto/camellia-aesni-avx-x86_64.ko
INSTALL arch/x86/crypto/camellia-aesni-avx2.ko
INSTALL arch/x86/crypto/camellia-x86_64.ko
INSTALL arch/x86/crypto/cast5-avx-x86_64.ko
INSTALL arch/x86/crypto/cast6-avx-x86_64.ko
INSTALL arch/x86/crypto/chacha20-x86_64.ko
INSTALL arch/x86/crypto/crc32-pclmul.ko
INSTALL arch/x86/crypto/crct10dif-pclmul.ko
INSTALL arch/x86/crypto/des3-ede-x86_64.ko
INSTALL arch/x86/crypto/ghash-clmulni-intel.ko
INSTALL arch/x86/crypto/glue_helper.ko
INSTALL arch/x86/crypto/poly1305-x86_64.ko
INSTALL arch/x86/crypto/salsa20-x86_64.ko
INSTALL arch/x86/crypto/serpent-avx-x86_64.ko
```

- Installasi kernel

```
# make install
```

```
root@156150600111002: /usr/src/linux-4.5.2
root@156150600111002: /usr/src/linux-4.5.2# make install
sh ./arch/x86/boot/install.sh 4.5.2 arch/x86/boot/bzImage \
    System.map "/boot"
run-parts: executing /etc/kernel/postinst.d/apt-auto-removal 4.5.2 /boot/vmlinuz-4.5.2
run-parts: executing /etc/kernel/postinst.d/initramfs-tools 4.5.2 /boot/vmlinuz-4.5.2
update-initramfs: Generating /boot/initrd.img-4.5.2
run-parts: executing /etc/kernel/postinst.d/pm-utils 4.5.2 /boot/vmlinuz-4.5.2
run-parts: executing /etc/kernel/postinst.d/unattended-upgrades 4.5.2 /boot/vmlinuz-4.5.2
run-parts: executing /etc/kernel/postinst.d/update-notifier 4.5.2 /boot/vmlinuz-4.5.2
run-parts: executing /etc/kernel/postinst.d/zz-update-grub 4.5.2 /boot/vmlinuz-4.5.2
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-4.5.2
Found initrd image: /boot/initrd.img-4.5.2
Found linux image: /boot/vmlinuz-4.4.0-22-generic
Found initrd image: /boot/initrd.img-4.4.0-22-generic
Found linux image: /boot/vmlinuz-4.4.0-21-generic
Found initrd image: /boot/initrd.img-4.4.0-21-generic
Found linux image: /boot/vmlinuz-4.2.0-35-generic
Found initrd image: /boot/initrd.img-4.2.0-35-generic
Found linux image: /boot/vmlinuz-4.2.0-34-generic
Found initrd image: /boot/initrd.img-4.2.0-34-generic
Found linux image: /boot/vmlinuz-4.2.0-30-generic
Found initrd image: /boot/initrd.img-4.2.0-30-generic
Found memtest86+ image: /boot/memtest86+.elf
Found memtest86+ image: /boot/memtest86+.bin
Found Windows 8 (loader) on /dev/sda1
done
root@156150600111002: /usr/src/linux-4.5.2#
```

- Membuat initial ramdisk image

```
# cd /boot
# mkinitramfs -o initrd.img-3.2.22
```

```
root@156150600111002: /boot
root@156150600111002: /usr/src/linux-4.5.2# cd /boot
root@156150600111002: /boot# mkinitramfs -o initrd.img-4.5.2
root@156150600111002: /boot#
```

- Perbaharui grub

```
# update-grub
```



```
root@156150600111002: /boot
root@156150600111002: /boot# update-grub
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-4.5.2
Found initrd image: /boot/initrd.img-4.5.2
Found linux image: /boot/vmlinuz-4.4.0-22-generic
Found initrd image: /boot/initrd.img-4.4.0-22-generic
Found linux image: /boot/vmlinuz-4.4.0-21-generic
Found initrd image: /boot/initrd.img-4.4.0-21-generic
Found linux image: /boot/vmlinuz-4.2.0-35-generic
Found initrd image: /boot/initrd.img-4.2.0-35-generic
Found linux image: /boot/vmlinuz-4.2.0-34-generic
Found initrd image: /boot/initrd.img-4.2.0-34-generic
Found linux image: /boot/vmlinuz-4.2.0-30-generic
Found initrd image: /boot/initrd.img-4.2.0-30-generic
Found memtest86+ image: /boot/memtest86+.elf
Found memtest86+ image: /boot/memtest86+.bin
Found Windows 8 (loader) on /dev/sda1
done
root@156150600111002: /boot#
```

- Proses selesai dan restart untuk melihat perubahan
reboot

6. Boot Linux to the new Kernel

```
Ubuntu, with Linux 4.5.2
Ubuntu, with Linux 4.5.2 (upstart)
Ubuntu, with Linux 4.5.2 (recovery mode)
Ubuntu, with Linux 4.4.0-22-generic
Ubuntu, with Linux 4.4.0-22-generic (upstart)
Ubuntu, with Linux 4.4.0-22-generic (recovery mode)
Ubuntu, with Linux 4.4.0-21-generic
Ubuntu, with Linux 4.4.0-21-generic (upstart)
Ubuntu, with Linux 4.4.0-21-generic (recovery mode)
Ubuntu, with Linux 4.2.0-35-generic
Ubuntu, with Linux 4.2.0-35-generic (upstart)
Ubuntu, with Linux 4.2.0-35-generic (recovery mode)
Ubuntu, with Linux 4.2.0-34-generic
Ubuntu, with Linux 4.2.0-34-generic (upstart)
Ubuntu, with Linux 4.2.0-34-generic (recovery mode)
Ubuntu, with Linux 4.2.0-30-generic
Ubuntu, with Linux 4.2.0-30-generic (upstart)
Ubuntu, with Linux 4.2.0-30-generic (recovery mode)
```



LAPORAN PRAKTIKUM SISTEM OPERASI FAKULTAS ILMU KOMPUTER UNIVERSITAS BRAWIJAYA

Nama : Moh. Arif Andrian
NIM : 156150600111002
Kesimpulan : BAB I
Asisten : Siska Permatasari
Zaenal Kurniawan

KESIMPULAN

Secara umum, Sistem Operasi adalah software pada lapisan pertama yang ditempatkan pada memori komputer pada saat komputer dinyalakan. Sedangkan software lainnya dijalankan setelah Sistem Operasi berjalan, dan Sistem Operasi akan melakukan layanan inti umum untuk software-software itu. Layanan inti umum tersebut seperti akses ke disk, manajemen memori, skeduling task, dan antar-muka user. Sehingga masing-masing software tidak perlu lagi melakukan tugas-tugas inti umum tersebut, karena dapat dilayani dan dilakukan oleh Sistem Operasi. Bagian kode yang melakukan tugas-tugas inti dan umum tersebut dinamakan dengan “kernel” suatu Sistem Operasi.

Pendekatan yang umum suatu sistem yang besar dan kompleks adalah dengan memecah tugas-tugas (task) ke bentuk komponen-komponen kecil dibandingkan dalam bentuk sistem tunggal (monolithic). Ada beberapa jenis struktur pembagian komponen sistem operasi antara lain Struktur Sederhana, Struktur Lapisan, Struktur Monolitik, Struktur Microkernel, Struktur Modular.