کامپایل سیستم عامل

یا چگونه لینوکس را سفارشی‌سازی کنیم.

آزمایشگاه سیستم عامل دانشگاه شهید چمران اهواز

پس از پایان این بخش:

- **شما توانسته‌اید تا لینوکس را سفارشی‌سازی کنید.**

پیش‌نیازها:

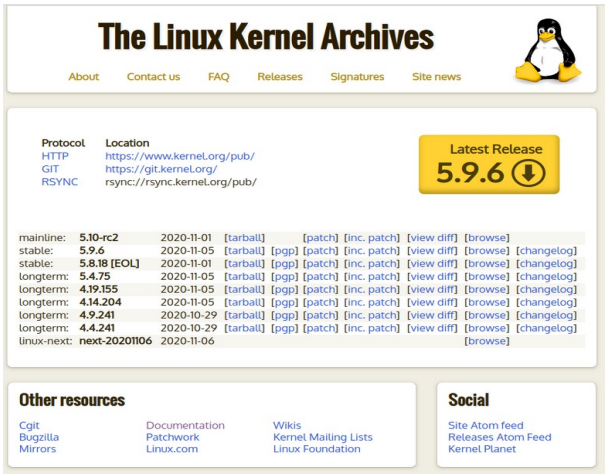
- **آشنایی با دستورها در لینوکس**
- **آزمایش شماره‌ی ۱**
- **شیفتگی به ژرفای هسته!**

وحید محمدی صفارزاده

برای آشنایی با کامپایل هسته‌ی سیستم‌عامل این آزمایش فراهم شده است. این ورزش به شما می‌آموزد که چگونه دگرگونی‌هایی که در برنامه‌ها و فایل‌های هسته پدید می‌آورید را می‌توانید به زبان پایین‌رده و دودویی برگردانید (کامپایل کنید) تا از آن‌ها در «ویرایش» تازه‌ی لینوکس خود بهره ببرید (این کار را در آزمایش فراخوانی سیستمی انجام خواهید داد).

۱.۱ دریافت برنامه‌ی هسته

برنامه‌ها و پرونده‌هایی که برای کامپایل لینوکس نیاز است، در یک پرونده‌ی فشرده tarball جای دارند (این پرونده با دستور tar ساخته شده است). این پرونده برای هر ویرایش هسته‌ی لینوکس در نشانی www.kernel.org در دسترس است.



نگاره ۱: تارنمایی که برای دریافت ویرایش‌های گوناگون هسته‌ی لینوکس در نشانی kernel.org در دسترس است. در زمان نوشتن این نوشته ویرایش شماره‌ی ۵.۹.۶ در دسترس بود.

هم‌اکنون فایلهای هسته با پسوند xz. هستند که برای نافشرده کردن آن از دستور زیر می‌توان بهره برد:

```
$ tar -xf  <نام پرونده‌ی فشرده>
```

پوشه‌ی نافشرده را در جایی بگذارید که در نام‌های نشانی آن همه‌ی واژه‌ها به‌هم‌پیوسته باشند. اگر چنین نباشد در آغاز کامپایل با پیام هشدار زیر روبرو خواهید شد:

```
Makefile:151: *** source directory cannot contain spaces or colons.  Stop.
```

بهترین جا برای کار روی هسته همان نشانی «خانه یا home» است. در اینجا با هسته‌ی ویرایش ۵,۹,۶ کار می‌کنیم.

۱.۲ درختواره‌ی هسته

ساختار پرونده‌های درون پوشه‌ی بارگیری‌شده در جدول زیر آمده است.

پوشه	کارکرد
arch	دربرگیرنده‌ی برنامه‌های ویژه‌ی هر معماری/سامانه (architecture) مانند x86 یا arm

block	دارای برنامه‌هایی که کار آن‌ها مدیریت دستگاه‌های ورودی_خروجی بسته‌ای (block devices)
crypto	دارای میآنای رمزنگاری[1]، (این میآنا بستری برای رمزنگاری در بخش‌های گوناگون است)
Documentation	دربرگیرنده‌ی توضیحات برنامه‌های هسته
drivers	دربرگیرنده‌ی برنامه‌های گرداننده‌ی سخت‌افزارها
fs	دربرگیرنده‌ی برنامه‌های پیاده‌سازی ساختمان‌های فایلی مجازی و دیگر ساختمان‌های فایلی
include	دارنده‌ی سربرگ‌های زبان سی که برای هسته ساخته شده‌اند
init	دربرگیرنده‌ی برنامه‌های راه‌اندازی (بوت) و آغازش (initialization) هسته
ipc	دارنده‌ی برنامه‌هایی که برای فراهم‌کردن ارتباط‌های میان‌فرایندی نیاز هستند.
kernel	دربرگیرنده‌ی برنامه‌های زیرسامانه‌های مرکزی هسته، مانند «زمان‌بندی»
lib	دربرگیرنده‌ی تکه‌برنامه‌های کمکی
LICENSES	
mm	دربرگیرنده‌ی برنامه‌های زیرسامانه‌های حافظه‌گردانی و حافظه‌ی مجازی
net	دربرگیرنده‌ی برنامه‌هایی برای زیرسامانه‌های شبکه
samples	دارنده‌ی برنامه‌های نمونه که برخی کارکردها را در هسته نشان می‌دهند.
scripts	دربرگیرنده‌ی برنامه‌هایی (اسکریپت‌هایی) که برای ساخت هسته هستند.
security	پودمان (ماژول) امنیت لینوکس
sound	زیرسامانه‌ی صدا
tools	ابزارهایی که برای گسترش لینوکس نیاز هستند.
usr	نخستین برنامه‌ها (کدهای) تراز کاربر (به نام initramfs شناخته می‌شوند.)
virt	سازوکارهای مجازی‌سازی
پرونده	**کارکرد**
COPYING	پروانه یا لیسانس هسته است (GNU GPL v2)
CREDITS	فهرست نام برنامه‌نویسان هسته که بخش بسزایی از برنامه را نوشته‌اند یا گسترش داده‌اند.
Kbuild	
Kconfig	
MAINTAINERS	فهرست نام کسانی که کارشان نگهداری زیرسامانه‌ها و گردانندگان سخت‌افزار است.
Makefiles	فایل سازنده‌ی اصلی هسته

1 Crypto API

۱.۳ روش ساخت (build) سیستم عامل لینوکس[۲]

با نگاهی از بالا، هسته‌ی لینوکس مانند هر برنامه‌ی دیگری که به زبان سی نوشته شده است به زبان پایین‌رده برده می‌شود (کامپایل می‌شود). ولی از آنجا که هسته برنامه‌ای پیچیده‌تر و دارای ساختاری ویژه است، در نگاه نخست، گام‌های کامپایل چهره‌ای ناهمسان با برنامه‌هایی که تاکنون نوشته‌اید خواهند داشت. به زبان دیگر، ما هسته را «می‌سازیم» و در زمان ساختن هسته برخی بخش‌های آن نیاز به «کامپایل‌شدن» دارند. از این‌رو، برای ساخت بهینه‌ی هسته‌ی لینوکس و دوری جستن از انجام کار بیهوده باید با ابزار تازه‌ای هم آشنا شوید.

۱.۳.۱ ابزار (برنامه‌ی) گرداورنده‌ی make

در ترمینال لینوکس خود دستور man make را انجام دهید. این ابزار برای گرداوری و کامپایل برنامه‌های بزرگ به کار می‌رود. چنین برنامه‌هایی دارای چندین بخش هستند و گاهی با چند زبان نوشته می‌شوند. دستور make به‌صورت پیش‌فرض از یک فایل به نام Makefile برای انجام گام‌به‌گام کامپایل و ساخت فایل اجرایی بهره می‌برد. ساخت این فایل کار زمان‌بر و گاهی پیچیده است ولی کار کامپایل را ساده می‌کند و سازندگان برنامه‌های بزرگ برای کامپایل برنامه‌های خود آن را برای گسترش‌دهندگان فراهم می‌سازند. اگر به پرونده‌های درون پوشه‌ی هسته‌ی بارگیری‌شده نگاه کنید، این فایل را خواهید یافت. هنگامی که در گام‌های آینده دستور make را اجرا می‌کنید، این دستور خودش به دنبال این فایل می‌گردد و برپایه‌ی آنچه در این فایل گنجانده شده هسته را می‌سازد. اگر make در سیستم شما نیست، با دستور زیر می‌توانید آن را بارگیری و نصب کنید:

```
$sudo apt install make
```

۱.۴ پیکربندی[۳] هسته

پیش از ساخت هسته باید برخی پیکربندی‌ها را در آن انجام دهیم. چرا که هسته امکانات فراوانی را فراهم می سازد، از این رو می‌توان آنها را درخواست کرد که در هسته‌ی ساخته‌شده در دسترس باشند یا نه. پیکربندی هسته با «گزینه»هایی انجام می‌شود که دارای پیشوند CONFIG هستند. برای نمونه «چندپردازشی آینه‌ای[۴]» با گزینه‌ی پیکربندی CONFIG_SMP در دسترس است. اگر این گزینه خواسته شود SMP به‌کار گرفته خواهد شد. همین پیکربندی‌ها می‌نماید که کدام بخش‌های هسته نیاز به ساخت و کامپایل دارند. بیشتر این گزینه‌ها دوگانه (دوتایی[۵] یا سه‌گانه (سه‌تایی) [۶] هستند.

به هر روی، ما برای سفارشی‌سازی هسته در زمینه‌ی نیاز کاری خود می‌بایست آن را پیکربندی کنیم و تنها بخش‌هایی را که نیاز داریم در آن نگه داریم.

---

۲ البته این تنها یک روش کامپایل است با یک نسخه‌ی خاص. ممکن است در زمان آینده کمی تغییر ایجاد شود. تلاش کنید خودتان با کمک این راهنما و اینترنت کار کامپایل را انجام دهید.

3 Confuguration
4 Symmetrical Multi-Processing (SMP)
5 boolean
6 tristate

۱.۴.۱روش‌های انجام پیکربندی هسته

با **یکی از روش‌های** زیر می‌توانید به گزینه‌های پیکربندی هسته‌ی تازه دسترسی پیدا کنید و آن‌ها را درخواست یا از آن‌ها چشم‌پوشی کنید. برای انجام این دستورها در اوبونتوی ۲۰,۰۴ سه ابزار (بسته) زیر را باید نصب کنید.

ابزار یا بسته	دستور جایگذاری
build-essential	sudo apt install build-essential
bison	sudo apt install bison
flex	sudo apt install flex

به‌طور خلاصه، شما نخست یک فایل پیکربندی (به نام config.) باید بسازید و سپس، باید هسته‌ی تازه را برپایه‌ی همین فایل بسازید. روش‌های شماره‌ی ۱ تا ۶ برای ساخت چنین فایلی است (پیشنهاد می‌شود برای نخستین بار روش شماره‌ی ۶ را به کار گیرید.).

۱. **در پوشه‌ای که از نافشرده‌سازی فایل بارگیری‌شده به دست آمده (نام این پوشه چیزی مانند linux-5.9.6 است)**، برای پیکربندی به‌سادگی می‌توان از دستور زیر بهره برد:

```
$ make config
```

با این دستور همه‌ی گزینه‌های پیکربندی یک‌به‌یک می‌آیند و کاربر می‌تواند با «yes یا y یا Y» آن‌ها را در هسته بگنجاند یا با « no یا n یا N» از گنجاندن آن‌ها چشم‌پوشی کند. نمونه‌ای از خروجی این دستور و نیز پرسش‌هایی که این دستور می‌پرسد و پاسخ‌هایی که کاربر داده است در زیر آمده است. می‌بینید که پرسش‌هایی در زمینه‌ی گنجاندن گزینه‌های حافظه‌ای، نام میزبان یا درباره‌ی شیوه‌ی فشرده‌سازی هسته و ... پرسیده شده است.

```
scripts/kconfig/conf  --oldaskconfig Kconfig
#
# using defaults found in /boot/config-5.4.0-52-generic
#
/boot/config-5.4.0-52-generic:3815:warning: symbol value 'm' invalid for ISDN_CAPI
/boot/config-5.4.0-52-generic:8246:warning: symbol value 'm' invalid for ASHMEM
/boot/config-5.4.0-52-generic:9206:warning: symbol value 'm' invalid for ANDROID_BINDER_IPC
/boot/config-5.4.0-52-generic:9207:warning: symbol value 'm' invalid for ANDROID_BINDERFS
/boot/config-5.4.0-52-generic:9275:warning: symbol value 'm' invalid for INTERCONNECT
*
* Linux/x86 5.9.0 Kernel Configuration
*
*
* General setup
*
Compile also drivers which will not load (COMPILE_TEST) [N/y/?] N
Local version - append to kernel release (LOCALVERSION) [] scuoslabversion
Automatically append version information to the version string (LOCALVERSION_AUTO) [N/y/?] y
Build ID Salt (BUILD_SALT) []
Kernel compression mode
  1. Gzip (KERNEL_GZIP)
  2. Bzip2 (KERNEL_BZIP2)
  3. LZMA (KERNEL_LZMA)
  4. XZ (KERNEL_XZ)
  5. LZO (KERNEL_LZO)
> 6. LZ4 (KERNEL_LZ4)
  7. ZSTD (KERNEL_ZSTD) (NEW)
choice[1-7?]: 1
Default init path (DEFAULT_INIT) [] (NEW)
Default hostname (DEFAULT_HOSTNAME) [(none)] OSLab
Support for paging of anonymous memory (swap) (SWAP) [Y/n/?] Y
System V IPC (SYSVIPC) [Y/n/?] Y
POSIX Message Queues (POSIX_MQUEUE) [Y/n/?] Y
General notification queue (WATCH_QUEUE) [N/y/?] (NEW) Y
Enable process_vm_readv/writev syscalls (CROSS_MEMORY_ATTACH) [Y/n/?] Y
uselib syscall (USELIB) [Y/n/?] Y
Auditing support (AUDIT) [Y/?] y
*
* IRQ subsystem
*
Expose irq internals in debugfs (GENERIC_IRQ_DEBUGFS) [N/y/?] Y
*
* Timers subsystem
*
Timer tick handling
  1. Periodic timer ticks (constant rate, no dynticks) (HZ_PERIODIC)
> 2. Idle dynticks system (tickless idle) (NO_HZ_IDLE)
  3. Full dynticks system (tickless) (NO_HZ_FULL)
choice[1-3?]: 3
```

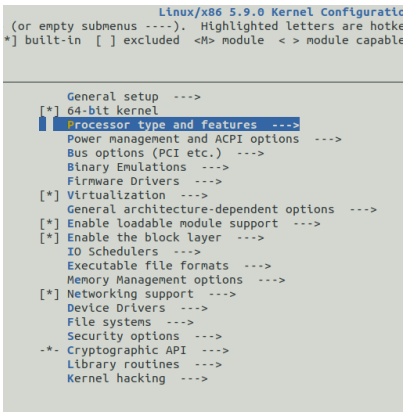
۲. به جای این دستور می‌توانید از دستور (ابزار) زیر برای کار در یک بستر نمایشی و ساده‌تر بهره ببرید:

```
$make menuconfig
```

این ابزار نیز خود از بسته‌ی گرافیکی ncurses بهره می‌برد. پس نخست باید ویرایش dev از این بسته را با دستور زیر نصب و جایگذاری کنید:

```
$ sudo apt install ncurses-dev
```

نمونه‌ای از انجام این دستور make در زیر آمده است که دسته‌بندی گزینه‌های پیکربندی در آن نمایان است.



۳. همچنین می‌توانید از ابزار زیر که با بسته‌ی نمایشی gtk+ است بهره ببرید. برای کار با این ابزار هم به بسته‌هایی نیاز دارید که باید جایگذاری کنید.

```
$ make gconfig
```

۴. اگر شناختی از گزینه‌های پیکربندی ندارید می‌توانید با اجرای دستور زیر یک پیکربندی «آماده[۷]» را برای ساخت هسته به کار بگیرید تا بتوانید کار را زودتر دنبال کنید.

```
$make defconfig
```

۵. کمترین پیکربندی هسته که دارای کمترین ویژگی‌هاست را می‌توانید با دستور زیر بسازید:

```
$make tinyconfig
```

تنها باید بدانید که این هسته هیچ امکاناتی (حتی پشتیبانی از سیستم ۶۴ بیت) ندارد. می‌توانید برای نمونه پشتیبانی از ۶۴ بیت را با make menuconfig به آن بیافزایید ولی باز هم محیط کار سیستم‌عامل به چنین هسته‌ای فراهم نخواهد بود. بهتر است تنها برای دیدن روند کار و فایل bzImage این دستور را به‌کار گیرید چرا که زودتر از دستورهای دیگر هسته را می‌سازد.

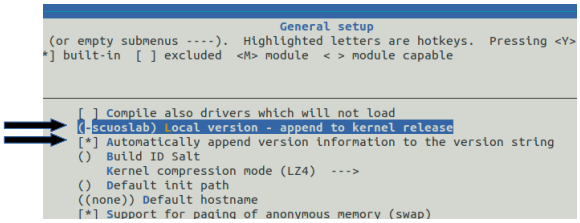
7 default

۶. **ساده‌ترین و کم‌خطرترین راه:** می‌توانید از فایل پیکربندی هسته‌ی کنونی سیستم خود بهره ببرید. برای این کار باید یک رونوشت از فایل پیکربندی هسته‌ی کنونی که در پوشه‌ی boot/ سیستم است به پوشه‌ی هسته‌ی تازه بیاورید. این کار را با اجرای دستور زیر در **نشانی پوشه‌ی هسته‌ی تازه** انجام دهید:

```
$cp /boot/config-$(uname -r) .config
```

دستور uname -r شماره‌ی ویرایش هسته‌ی کنونی سیستم را می‌دهد.

سپس، اگر تغییری را می‌خواهید، با دستور make menuconfig می‌توانید این کار را انجام دهید. برای نمونه، ما با تغییر دو گزینه‌ی زیر در دسته‌ی General Setup رشته‌ی scuoslab را به پایان شماره‌ی ویرایش (version) هسته افزوده‌ایم.



گزینه‌های پیکربندی در پوشه‌ی هسته‌ی بارگیری‌شده و در پرونده‌ی config. نگداری می‌شود. ویرایش این پرونده هم می‌تواند روشی ساده برای گزینه‌های پیکربندی شما باشد. (**یادآوری: در لینوکس فایل‌هایی که با «.» آغاز می‌شوند پنهان هستند و برای دیدن آن‌ها در ترمینال باید از دستور ls -a و در پنجره‌ها از کلید ترکیبی ctrl+h کمک بگیرید.**)

پس از ساخت (یا ویرایش) پرونده config. برای درستی‌سنجی و به‌روزرسانی پیکربندی دستور زیر را باید انجام دهید:

```
$make oldconfig
```

۱.۵ ساخت هسته

ساخت هسته کاری ساده است. پس از انجام پیکربندی هسته (به هر روش) با دستور زیر آن را بسازید (باید گنجایش حافظه‌ی بلندمدت سیستم یا ماشین‌مجازی شما کافی باشد):

```
$make
```

برای کاهش خروجی‌های دستور و نوشتن آنها در یک پرونده‌ی دلخواه می‌توانید این دستور را اینگونه به‌کار بگیرید:

```
$make > ../detritus
```

یا اگر خروجی را هم نمی‌خواهید، اینگونه:

```
$make > /dev/null
```

/dev/null/ یک فضای بی‌بازگشت (!) برای خروجی دستور است.

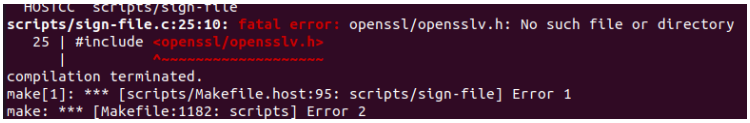
همچنین، برای پایان زودتر دستور می‌توانید از ویژگی چندکاری که پرونده‌های Makefile دارند بهره ببرید و آن را روی چند **پردازنده و همزمان با دستور** زیر اجرا کنید:

```
$make -jn
```

n شمار فرایندها یا پردازنده‌هایی است که برای انجام کار ساخت هسته به کار گرفته می‌شوند. برای نمونه، در یک رایانه‌ی ۸ هسته‌ای می‌توانیم به هر هسته دو کار بدهید که دستور اینگونه می‌شود:

```
$make -j16 > /dev/null
```

اگر در زمان ساخت با هشدار زیر روبرو شدید:



با انجام دستور زیر (نصب بسته‌ی openssl) آن را برطرف کنید.

```
$sudo apt install libssl-dev
```

هشدار: دو گام «پیکربندی» و «ساخت» هسته را می‌توانید در یک سیستم دیگر انجام دهید. ولی، از این پس، کارها را باید در همان سیستمی که می‌خواهید هسته روی آن کار کند انجام دهید. برای نمونه، اگر می‌خواهید هسته را در یک ماشین مجازی بگذارید، می‌توانید ساخت هسته را که زمان‌بر است و سخت‌افزار نیرومندتری می‌خواهد در سیستم میزبان انجام دهید و سپس پوشه‌ی هسته را به ماشین مجازی ببرید و دنباله‌ی کارها را در ماشین مجازی انجام دهید.

۱.۶ جای‌دهی هسته‌ی کامپایل‌شده در سیستم (به کارگیری آن)

پس از ساخت هسته‌ی تازه باید آن را نصب کنیم؛ پس باید پرونده‌های به‌دست‌آمده از دستور make را در جاهایی درست بگذارید تا در هنگام بالا آمدن (boot)، راه‌انداز (bootloader) هسته‌ی تازه را نیز ببیند. همواره یک هسته درست و آزمایش‌پس‌داده در دسترس داشته باشید چراکه شاید هسته‌ی تازه دچار ناهم‌خوانی‌هایی باشد؛ اگر هسته‌ی پیشین را دستکاری یا پاک نکنید، این هسته در دسترس خواهد ماند و در فهرست گراب (در هنگام بالا آمدن لینوکس) می‌توانید آن را برگزینید.

۱.۶.۱جایگذاری پودمان‌ها

دستور زیر (با دسترسی ریشه[8]، اجرا با sudo) برای نصب پودمان‌ها (ماژول‌ها) است:

```
$make modules_install
```

8 root, admin

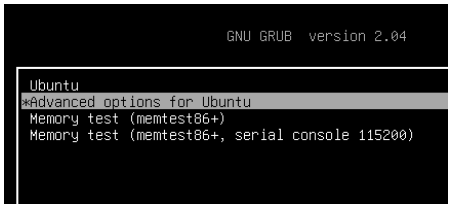
این دستور همه‌ی پودمان‌های کامپایل‌شده را در نشانی درست خودشان (/lib/modules) در سیستم می‌گذارد.

۱.۶.۲جایگذاری‌های پایانی و پیکربندی گراب

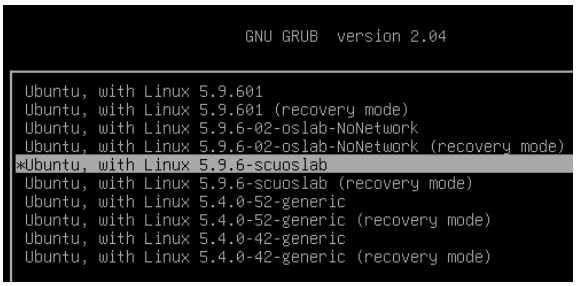
سپس، برای نمونه، در یک سیستم x86 که از گراب (grub) بهره می‌برد، باید یک رونوشت از فایلی که در فرایند ساخت هسته به دست آمد (همان arch/x86/boot/bzImage) را به boot/ ببریم و نام آن را چیزی مانند vmlimuz-> <version بگذاریم . سپس، باید فایل پیکربندی گراب را که در نشانی boot/grub/ است ویرایش کنیم و یک ورودی برای هسته تازه به آن بیافزاییم. همه‌ی این کارها را در ویرایش کنونی لینوکس می‌توانید با دستور زیر انجام دهید (گزینه‌ی j- همان کاربرد پیشین را دارد):

```
$make install -j 8
```

در اینجا کار ساخت و جایگذاری هسته پایان می‌یابد. اکنون باید سیستم را دوباره راه‌اندازی (reboot) کنید. و از فهرست زیر گزینه‌ی نشانده‌شده را برگزینید.



از فهرست زیر نیز گزینه‌ی هسته‌ای که ساخته‌اید (در اینجا رشته‌ی scuoslab- را در پایانش گذاشته بودیم) برگزینید.



همانگونه که می‌بینید در اینجا هسته‌های دیگر نیز در دسترس هستند. اگر سیستم با هسته‌ی شما بالا نیامد می‌توانید پس از راه‌اندازی دوباره، هسته‌های پیشین را برگزینید. برای نمونه، در اینجا هسته با ویرایش زیر را می‌توان انتخاب کرد:

5.4.0-52-generic

هشدار: اگر این فهرست گراب برای شما نمایش داده نشد، باید فایل گراب در نشانی زیر مانند نگاره ۲ بشود:

/etc/default/grub

گزینه‌ی GRUB_TIMEOUT_STYLE باید مقدارش menu باشد (به جای hidden) و گزینه‌ی GRUB_TIMEOUT باید مقداری بیش از ۵ داشته باشد.

نگاره ۲: برای نمایش فهرست گراب، فایل /etc/default/grub باید اینگونه باشد.

۱.۷ برخی از گزینه‌های پیکربندی

دستور شماره‌ی ۲ را بررسی کنید.

۱.۸ دستورکار

۱. پیکربندی هسته‌ی بارگیری‌شده به‌گونه‌ای بگذارید که «نام شما» را در پایان شماره‌ی ویرایش نمایش دهد. هسته را کامپایل کنید و اوبونتوی خود را با این هسته‌ی تازه راه‌اندازی کنید. برای این کار از فایل پیکربندی هسته‌ی کنونی اوبونتوی خود کمک بگیرید. پس از بالا آمدن سیستم با هسته‌ی ساخت شما، دستور زیر را اجرا کنید و نشان دهید که سیستم با هسته‌ی شما بالا آمده است:

$uname -r

برای نمونه اگر نام شما sohrab rostami است. این دستور باید چنین چیزی نمایش دهد:

5.9.6-sohrabrostami

۲. (دستور امتیازی) دو گزینه‌ی پیکربندی را در این آزمایش تغییر دادیم (گزینه‌های مربوط به نام ویرایش که آن را scuoslab گذاشتیم). همانگونه که در دستور make menuconfig هم دیدید این گزینه‌ها فراوانند. کارکرد چند نمونه از این گزینه‌ها را در گزارش کار خود بیاورید و بگویید که بود و نبود آن‌ها چه امکاناتی را به هسته می‌دهد یا از آنجا حذف می‌کند. همچنین آن‌ها را کم کنید یا اضافه کنید و هسته را با توجه به این تغییرات بسازید و نتیجه‌ی آن را گزارش کنید.

۱.۹ پرسش‌هایی برای درک ژرف‌تر

۱. چرا نیاز داریم که هسته‌ی سیستم عامل را کامپایل کنیم؟

۲. بسته‌ی نرم‌افزاری build-essential را چرا نصب می‌کنیم؟

۳. بسته‌ی libssl-dev به چه کاری می‌آید؟