$ mkdir build

$ gunzip config.gz > build/.config # или что угодно, для того, чтобы приготовить .config

$ make silentoldconfig prepare headers\_install scripts ARCH=arm CROSS\_COMPILE=arm-eabi- O=build KERNELRELEASE=`adb shell uname -r`

#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 "Hello world\n");

return 0;

}

static void \_\_exit hello\_end(void)

{

printk(KERN\_INFO "Goodbye world\n");

}

module\_init(hello\_start);

module\_exit(hello\_end);

obj-m = hello.o

MODULE\_INFO(vermagic, VERMAGIC\_STRING);

hello: version magic '3.0.21-perf-ge728813-00399-gd5fa0c9' should be '3.0.8-perf'

struct module \_\_this\_module

\_\_attribute\_\_((section(".gnu.linkonce.this\_module"))) = {

.name = KBUILD\_MODNAME,

.init = init\_module,

#ifdef CONFIG\_MODULE\_UNLOAD

.exit = cleanup\_module,

#endif

.arch = MODULE\_ARCH\_INIT,

};

struct module

{

(...)

#ifdef CONFIG\_UNUSED\_SYMBOLS

(...)

#endif

(...)

/\* Startup function. \*/

int (\*init)(void);

(...)

#ifdef CONFIG\_GENERIC\_BUG

(...)

#endif

#ifdef CONFIG\_KALLSYMS

(...)

#endif

(...)

(... plenty more ifdefs ...)

#ifdef CONFIG\_MODULE\_UNLOAD

(...)

/\* Destruction function. \*/

void (\*exit)(void);

(...)

#endif

(...)

}

static const struct modversion\_info \_\_\_\_versions[]

\_\_used

\_\_attribute\_\_((section("\_\_versions"))) = {

{ 0xsomehex, "module\_layout" },

{ 0xsomehex, "\_\_aeabi\_unwind\_cpp\_pr0" },

{ 0xsomehex, "printk" },

};

bool each\_symbol\_section(bool (\*fn)(const struct symsearch \*arr,

struct module \*owner,

void \*data),

void \*data)

{

struct module \*mod;

static const struct symsearch arr[] = {

{ \_\_start\_\_\_ksymtab, \_\_stop\_\_\_ksymtab, \_\_start\_\_\_kcrctab,

NOT\_GPL\_ONLY, false },

{ \_\_start\_\_\_ksymtab\_gpl, \_\_stop\_\_\_ksymtab\_gpl,

\_\_start\_\_\_kcrctab\_gpl,

GPL\_ONLY, false },

{ \_\_start\_\_\_ksymtab\_gpl\_future, \_\_stop\_\_\_ksymtab\_gpl\_future,

\_\_start\_\_\_kcrctab\_gpl\_future,

WILL\_BE\_GPL\_ONLY, false },

#ifdef CONFIG\_UNUSED\_SYMBOLS

{ \_\_start\_\_\_ksymtab\_unused, \_\_stop\_\_\_ksymtab\_unused,

\_\_start\_\_\_kcrctab\_unused,

NOT\_GPL\_ONLY, true },

{ \_\_start\_\_\_ksymtab\_unused\_gpl, \_\_stop\_\_\_ksymtab\_unused\_gpl,

\_\_start\_\_\_kcrctab\_unused\_gpl,

GPL\_ONLY, true },

#endif

};

if (each\_symbol\_in\_section(arr, ARRAY\_SIZE(arr), NULL, fn, data))

return true;

(...)

struct symsearch {

const struct kernel\_symbol \*start, \*stop;

const unsigned long \*crcs;

enum {

NOT\_GPL\_ONLY,

GPL\_ONLY,

WILL\_BE\_GPL\_ONLY,

} licence;

bool unused;

};