Sistemas Operativos
Edición 2022

Práctico 2: Compilación, enlazado y build systems

Ejercicios:

Nota: Para quienes tengan sistemas MacOS, pueden usar el toolchain para riscv instaladas como se indica en https://pdos.csail.mit.edu/6.S081/2021/tools.html. Las herramientas (gcc, objdump, ld, …) se invocan como `riscv64-...`

1. Dados los siguientes programas fuente:

```c
/* file: main.c */
int main(void)
{
    extern void f(void);

    f();
    return 0;
}
```

```c
/* file: f.c */

static void g(void)
{
    return;
}

void f(void)
{
    g();
}
```

   a. Compilar a código assembly cada módulo (`gcc -S main.c`).
   b. Generar los correspondientes archivos objeto (`main.o` y `f.o`).
   c. Generar el ejecutable myprog a partir de los códigos objeto.

2. Hacer un Makefile que contenga los objetivos asm, obj, myprog que construyan los productos de los puntos 1.a, 1.b y 1.c, respectivamente.

3. Observar las secciones de `main.o` y `f.o` (con `objdump -x <object_file>`) y desensamblar el código de ambos archivos objeto (`objdump -d <object_file>`).
   a. Determinar el valor del operando de la instrucción *`call`* (a *`f`*) en `main.o`.
   b. Describir la información provista al linker como *referencias externas a resolver* en `main.o`.
   c. Determinar si la invocación a `g()` en `f.c` está resuelta o no. Explicar.

4. Modificar main.c para que invoque a la función `g()`, incluyendo la declaración externa correspondiente.
    a. Generar nuevamente `main.o`.
    b. Tratar de generar el ejecutable `myprog`. Describa porqué falla el linking. Qué diferencia en la tabla de símbolos de `f.o` hay entre `f` y `g`?

5. Nuevamente, con las versiones originales de `main.c` y `f.c`, analizar los contenidos del *ELF* `myprog` con `objdump -x`.
    a. Analizar las secciones y el punto de entrada (`entry`) del programa.
    b. Desensamblar `myprog` y analizar cómo el linker resolvió las referencias externas.
    c. Determinar la información que el linker estático incluyó para el enlazado dinámico de la biblioteca estándar.
    ¿Por qué este programa debe enlazarse con la biblioteca estándar?

6. Crear un archivo f2.c que contenga alguna función a invocar desde `main.c`.
    a. Crear una biblioteca estática (llamada `libmyf.a`) que contenga los archivos objeto `f.o` y `f2.o` con el comando *`ar rcs libmyf.a f.o f2.o`*.
    Es posible listar los archivos contenidos en el contenedor *`libmyf.a`* mediante *`ar t libmyf.a`*. Para más información sobre `ar`, hacer `man ar`.
    b. Generar myprog enlazando main.o con la biblioteca:
    `gcc -o myprog main.o -L. -lmyf`

7. Crear una biblioteca compartida (`libmyf.so`) que contenga los archivos objetos `f.o` y `f1.o` (usar el flag `-shared` del gcc).
    a. Crear el ejecutable myprog2 enlazando `main.o` con `libmyf.so`.
    b. Comparar los tamaños y contenidos de `myprog` y `myprog2`.