code/link/interpose/int.c

```
#include <stdio.h>
#include <malloc.h>

int main()
{
    int *p = malloc(32);
    free(p);
    return(0);
}
```

code/link/interpose/int.c

a) 示例程序 int.c

code/link/interpose/malloc.h

```
#define malloc(size) mymalloc(size)
#define free(ptr) myfree(ptr)

void *mymalloc(size_t size);
void myfree(void *ptr);
```

code/link/interpose/malloc.h

b) 本地 malloc.h 文件

code/link/interpose/mymalloc.c

```
#ifdef COMPILETIME
#include <stdio.h>
#include <malloc.h>

/* malloc wrapper function */
void *mymalloc(size_t size)
{
    void *ptr = malloc(size);
    printf("malloc(%d)=%p\n",
           (int)size, ptr);
    return ptr;
}

/* free wrapper function */
void myfree(void *ptr)
{
    free(ptr);
    printf("free(%p)\n", ptr);
}
#endif
```

code/link/interpose/mymalloc.c

c) mymalloc.c 中的包装函数

图 7-20 用 C 预处理器进行编译时打桩

用下述方法把这些源文件编译成可重定位目标文件：

```
linux> gcc -DLINKTIME -c mymalloc.c
linux> gcc -c int.c
```

然后把目标文件链接成可执行文件：

```
linux> gcc -Wl,--wrap,malloc -Wl,--wrap,free -o intl int.o mymalloc.o
```

-Wl,option 标志把 option 传递给链接器。option 中的每个逗号都要替换为一个空