Ejercicio

Un obstinado desarollador de software está utilizando una compleja biblioteca de funciones desarrollada en un extraño y desconocido lenguaje de programación. Esta biblioteca tiene errores, por lo que el astuto desarrollador planteó un experimento para analizar todo el árbol de llamadas a funciones.

El experimento consta de reemplazar en el código compilado todas las instrucciones `call` x, por la instrucción `call log_call`. La función `log_call` tiene como objetivo loggear datos del llamado antes de hacerlo propiamente. Para ello se encargará de realizar los tres pasos siguientes:

1. Obtener la dirección de la función original (es decir la dirección de x)
2. Almacenar en el log de llamadas datos para el seguimiento
3. Llamar a la función x, como si nunca se hubiera llamado a `log_call`

Para los pasos 1 y 2 se proveen las siguiente funciones que deberán utilizar:

1. `void* get_func(void* addr_call)`
   Retorna el puntero a la función original dada la posición de memoria de la instrucción `call`.
2. `void store_data(void* rsp, void* rbp, void* func)`
   Almacena en el log de llamadas los valores que tenían `rsp` y `rbp` justo antes de la llamada a la función x, y el puntero `func` obtenido mediante la función anterior.

A. Programar en ASM la función `log_call`. Recordar que se debe respetar el estado del programa original en todo momento.

Nota: Considerar que la instrucción `call` ocupa exactamente 10 bytes. Además las funciónes dadas respetan convención C, con la salvedad que no afectan los registos `xmm1` a `xmm15`.