

19335015 陈恩婷

编程题：

利用 LLVM（C、C++）或者 Soot（Java）等工具检测多线程程序中潜在的数据竞争，并对比加上锁以后检测结果的变化，分析及给出案例程序并提交分析报告。

实验过程：

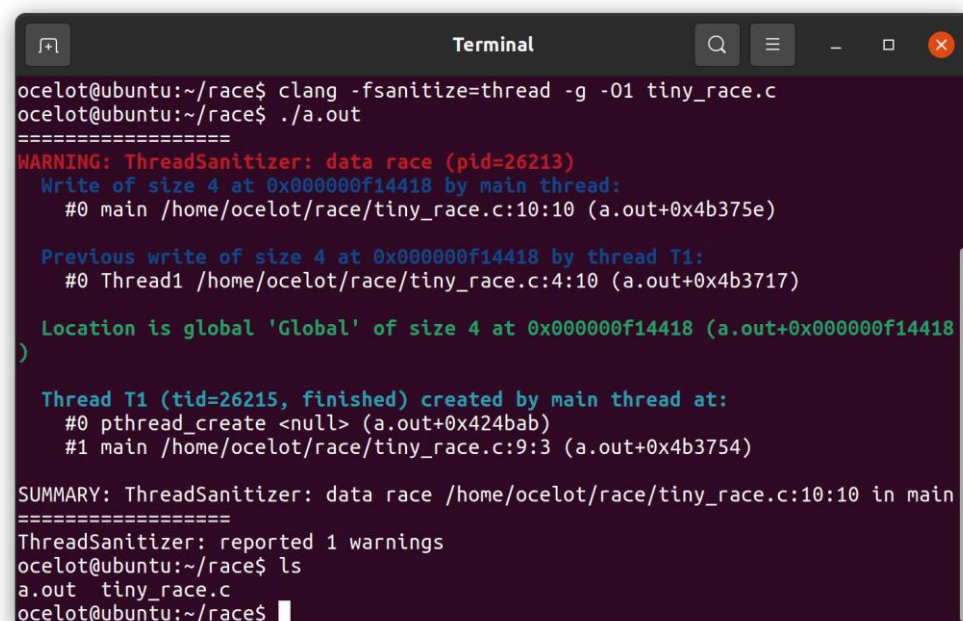
1. 编写具有数据竞争的多线程程序（C 或者 Java）；



```
1#include <pthread.h>
2int Global;
3void *Thread1(void *x) {
4    Global = 42;
5    return x;
6}
7int main() {
8    pthread_t t;
9    pthread_create(&t, NULL, Thread1, NULL);
10   Global = 43;
11   pthread_join(t, NULL);
12   return Global;
13}
```

如图所示，参考链接中的文档编写了以上程序。

2. 利用 LLVM 或者 Soot 将 C 或者 Java 程序编译成字节码；
3. 利用 LLVM 或者 soot 提供的数据竞争检测工具检测；



```
ocelot@ubuntu:~/race$ clang -fsanitize=thread -g -O1 tiny_race.c
ocelot@ubuntu:~/race$ ./a.out
=====
WARNING: ThreadSanitizer: data race (pid=26213)
  Write of size 4 at 0x000000f14418 by main thread:
    #0 main /home/ocelot/race/tiny_race.c:10:10 (a.out+0x4b375e)

  Previous write of size 4 at 0x000000f14418 by thread T1:
    #0 Thread1 /home/ocelot/race/tiny_race.c:4:10 (a.out+0x4b3717)

  Location is global 'Global' of size 4 at 0x000000f14418 (a.out+0x000000f14418)
)

Thread T1 (tid=26215, finished) created by main thread at:
  #0 pthread_create <null> (a.out+0x424bab)
  #1 main /home/ocelot/race/tiny_race.c:9:3 (a.out+0x4b3754)

SUMMARY: ThreadSanitizer: data race /home/ocelot/race/tiny_race.c:10:10 in main
=====
ThreadSanitizer: reported 1 warnings
ocelot@ubuntu:~/race$ ls
a.out tiny_race.c
ocelot@ubuntu:~/race$
```

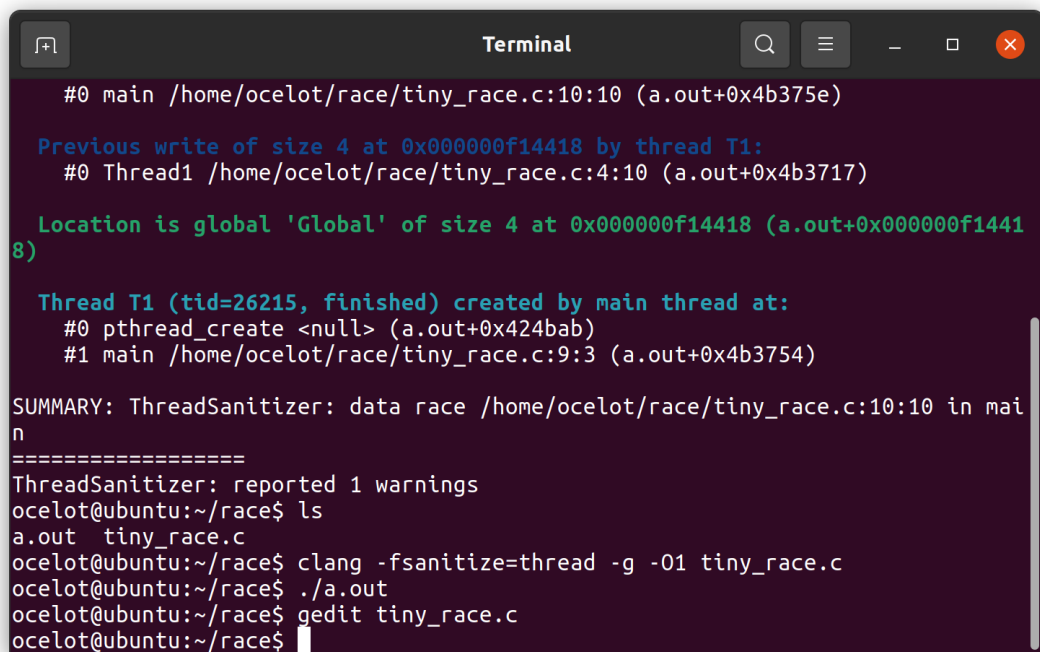
如图所示，在命令行中完成以上两步，可以看到在 Global 变量上出现了数据竞争，这是因为主线程和 pthread_create 创建的线程都在同一段时间尝试修改 Global 的值，导致 Global 的值受两次赋值操作的先后顺序的影响。

4. 对有数据竞争的地方加锁，观察检测结果；



```
1 #include <pthread.h>
2
3 pthread_rwlock_t rwlock;           //声明读写锁
4 int Global;
5 void *Thread1(void *x) {
6     pthread_rwlock_wrlock(&rwlock);
7     Global = 42;
8     pthread_rwlock_unlock(&rwlock);
9     return x;
10 }
11 int main() {
12     pthread_t t;
13     pthread_create(&t, NULL, Thread1, NULL);
14     pthread_rwlock_wrlock(&rwlock);
15     Global = 43;
16     pthread_rwlock_unlock(&rwlock);
17     pthread_join(t, NULL);
18     cout
19     return Global;
20 }
```

如图所示，利用 pthread 提供的读写锁，为 Global 变量加锁，修改变量的值前后分别对其上锁和解锁，再使用之前的命令进行编译和执行，就没有关于数据竞争的数据了：



```
#0 main /home/ocelot/race/tiny_race.c:10:10 (a.out+0x4b375e)
Previous write of size 4 at 0x000000f14418 by thread T1:
#0 Thread1 /home/ocelot/race/tiny_race.c:4:10 (a.out+0x4b3717)

Location is global 'Global' of size 4 at 0x000000f14418 (a.out+0x000000f14418)

Thread T1 (tid=26215, finished) created by main thread at:
#0 pthread_create <null> (a.out+0x424bab)
#1 main /home/ocelot/race/tiny_race.c:9:3 (a.out+0x4b3754)

SUMMARY: ThreadSanitizer: data race /home/ocelot/race/tiny_race.c:10:10 in main
=====
ThreadSanitizer: reported 1 warnings
ocelot@ubuntu:~/race$ ls
a.out  tiny_race.c
ocelot@ubuntu:~/race$ clang -fsanitize=thread -g -O1 tiny_race.c
ocelot@ubuntu:~/race$ ./a.out
ocelot@ubuntu:~/race$ gedit tiny_race.c
ocelot@ubuntu:~/race$
```

参考：<http://clang.llvm.org/docs/ThreadSanitizer.html>。