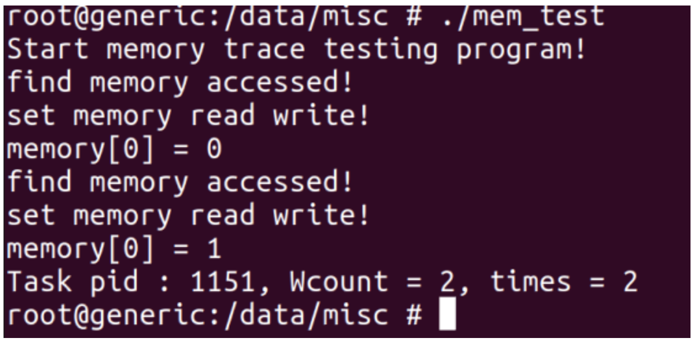
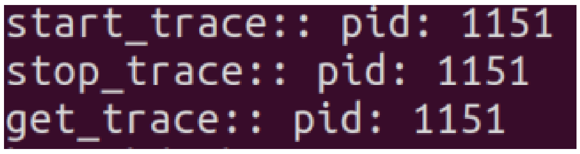
We use the process `mem_test` to test the page access tracing mechanism. The process starts page sccess tracing and allocates a set of memory, which can only be read. After that it try to write data to this memory space, and will receive a SIGSEGV. Then the function `__do_page_fault` will increase `wcounts`. At the same time, `segv_handler` will handle SIGSEGV by giving the memory write permission. After writing to this memory successfully, the process use `mprotect` to protect the memory again. If it try to write data to this memory space again, the same thing mentioned above will happen again. Finally, the process output the wcounts to show the result. Figure 1 and Figure 2 show the test result.

Here, we test page access tracing for only one process. We will test multiprocesses in the test `multiprocess`.



```
root@generic:/data/misc # ./mem_test
Start memory trace testing program!
find memory accessed!
set memory read write!
memory[0] = 0
find memory accessed!
set memory read write!
memory[0] = 1
Task pid : 1151, Wcount = 2, times = 2
root@generic:/data/misc #
```

Figure 1: `mem_test`: adb shell result



```
start_trace:: pid: 1151
stop_trace:: pid: 1151
get_trace:: pid: 1151
```

Figure 2: `mem_test`: kernel result