Assignment 5: Event Handle and Signaling – CSE 438/598, Fall 2014

Brahmesh Jain S D

December 5, 2014

1 Report on Task 2

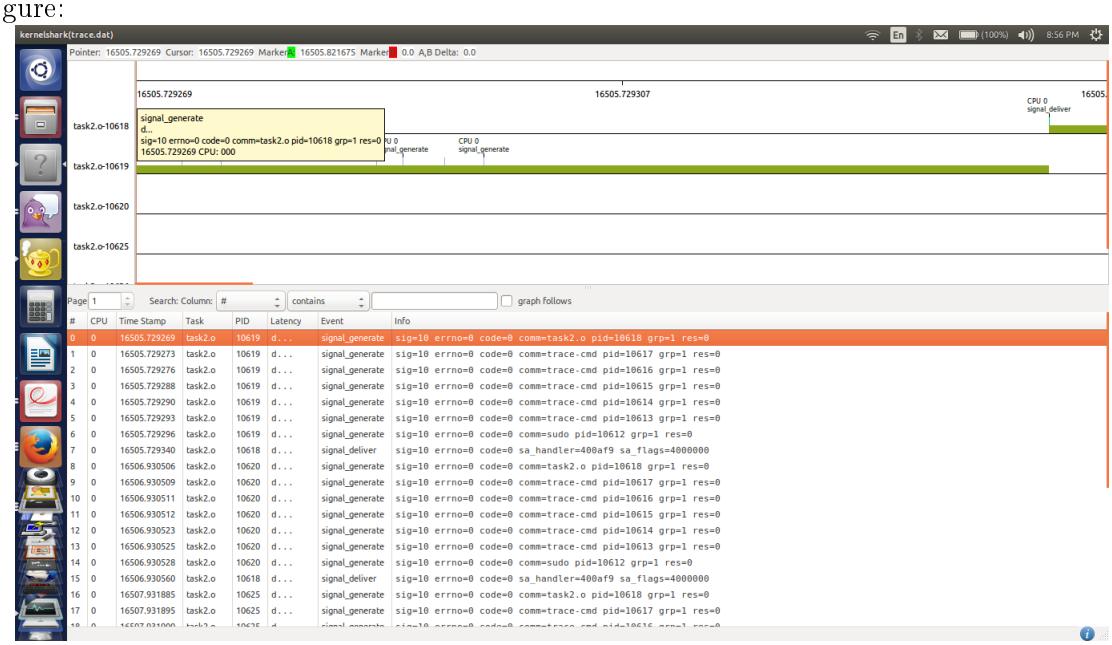
As a part of task 2 , we were asked to submit a report on the profiling of the program, illuminating the execution of signal handlers. Following sections discusses on how each part of the task was carried out.

Trace was carriend out by running the command:

sudo trace-cmd record -e signal taskset -c 0 ./task2.o

1.1 The thread is running

Main task runs continuosly, and anothter thread is created that sends the SIGUSR1 after a delay of 1 seconds. This signal will inturn call signal handler. This is as shown in the figure:

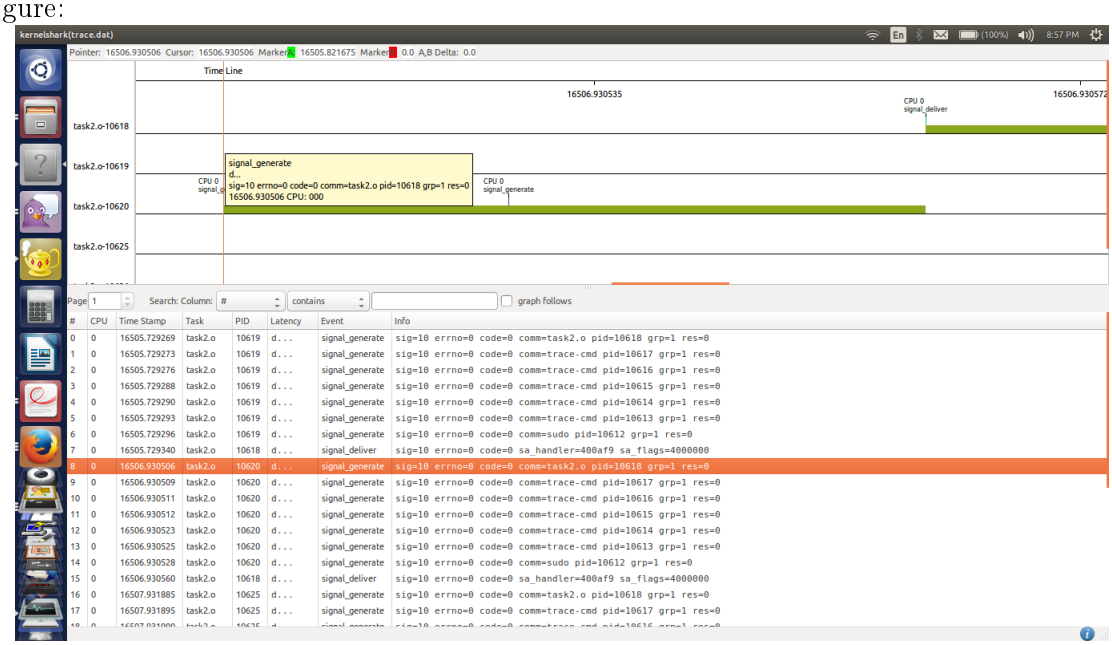


As we can see, signal was generated at 16505.729269 using kill function. This will be sent to kernel. Kernel has delivered to signal handler at 16505.729340. ie after 71us.

After the signal is received, signal handler will set a flag to main task to end the while loop.

1.2 The thread is runnable (Realtime Priority)

Main task runs continuosly, and anothter thread is created that sends the SIGUSR1 after a delay of 1 seconds. This signal will inturn call signal handler. This is as shown in the figure:

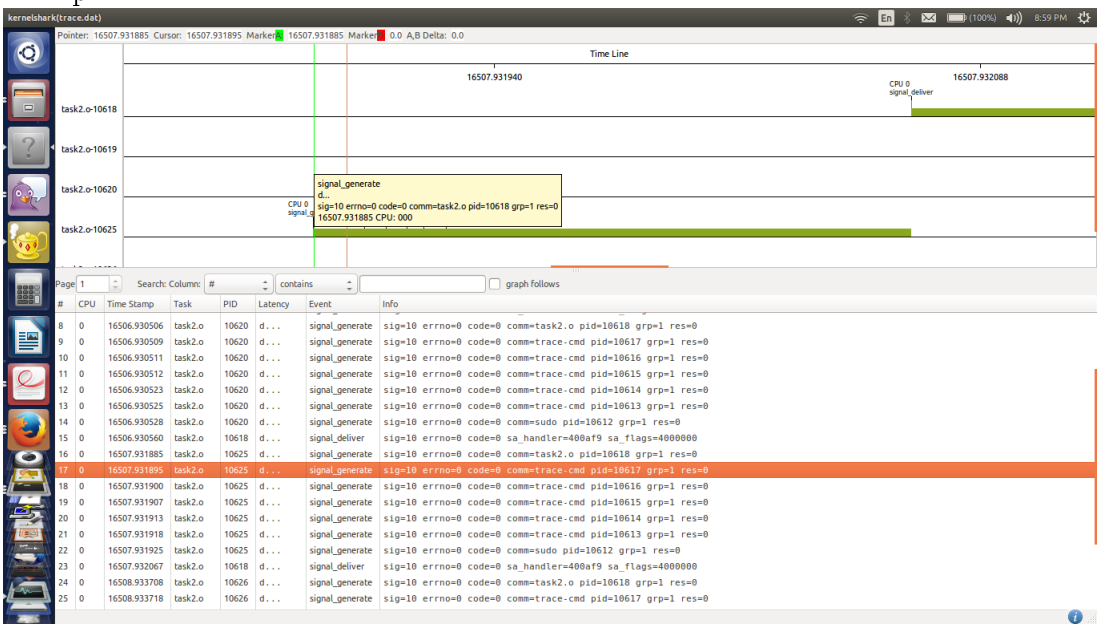


As we can see, signal was generated at 16506.930506 using kill function. This will be sent to kernel. Kernel has delivered to signal handler at 16506.930560. ie after 54us.

After the signal is received, signal handler will set a flag to main task to end the while loop.

1.3 The thread is blocked by a semaphore

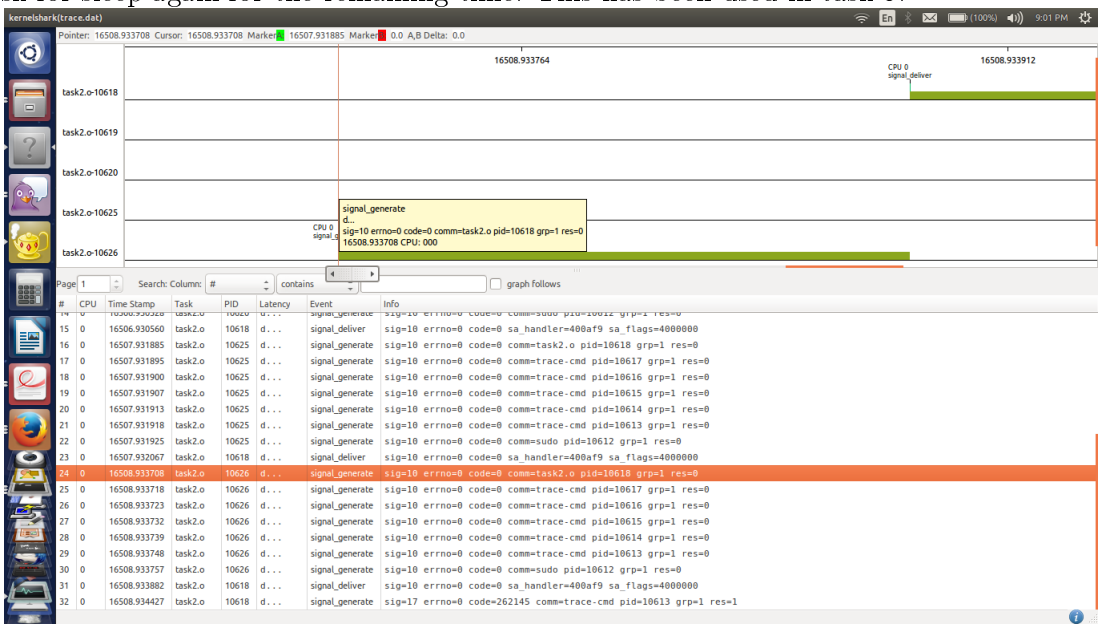
Main task creates a thread to send a SIGUSR1 after 1sec. In the mean time it blocks itself on waiting for a semaphore. Signal generated after 1second, which will send a kill function call to kernel. This signal is delivered to the main task. After it is delivered, the main task which was blocked by semaphore will become running even before acquiring the semphore.



As we can see, signal was generated at 16507.931895 using kill function. This will be sent to kernel. Kernel has delivered to signal handler at 16507.932067. ie after 172us.

1.4 The thread is delayed

Main task creates a thread to send a SIGUSR1 after 1sec. In the mean time it delyas itself by calling nanosleep(10sec). Signal generated after 1second, which will send a kill function call to kernel. This signal is delivered to the main task. After it is delivered, the main task which was in sleep for 10secs will wakeup and becomes running. Nanosleep also send the remaining time which was left for sleep. This time can be used to make task for sleep again for the remaining time. This has been used in task 3.



As we can see, signal was generated at 16508.933708 using kill function. This will be sent to kernel. Kernel has delivered to signal handler at 16508.933882. ie after 174us.