



## Linux debugging, tracing, profiling & perf. analysis

Check our new training course with Creative Commons CC-BY-SA lecture and lab materials

## Elixir Cross Referencer

≡ / samples / kprobes / kretprobe\_example.c

All symbols Search Identifier



```

1  // SPDX-License-Identifier: GPL-2.0-only
2  /*
3   * kretprobe_example.c
4   *
5   * Here's a sample kernel module showing the use of return probes to
6   * report the return value and total time taken for probed function
7   * to run.
8   *
9   * usage: insmod kretprobe_example.ko func=<func_name>
10  *
11  * If no func_name is specified, kernel_clone is instrumented
12  *
13  * For more information on theory of operation of kretprobes, see
14  * Documentation/trace/kprobes.rst
15  *
16  * Build and insert the kernel module as done in the kprobe example.
17  * You will see the trace data in /var/log/messages and on the console
18  * whenever the probed function returns. (Some messages may be suppressed
19  * if syslogd is configured to eliminate duplicate messages.)
20  */
21
22  #include <linux/kernel.h>
23  #include <linux/module.h>
24  #include <linux/kprobes.h>
25  #include <linux/ktime.h>
26  #include <linux/limits.h>
27  #include <linux/sched.h>
28
29  static char func_name[NAME_MAX] = "kernel_clone";
30  module_param_string(func, func_name, NAME_MAX, S_IRUGO);
31  MODULE_PARM_DESC(func, "Function to kretprobe; this module will report the"
32                      " function's execution time");
33
34  /* per-instance private data */
35  struct my_data {
36      ktime_t entry_stamp;
37  };
38

```

