# **Kernel Lock Torture Test Operation**

## **CONFIG LOCK TORTURE TEST**

The CONFIG LOCK\_TORTURE\_TEST config option provides a kernel module that runs torture tests on core kernel locking primitives. The kernel module, 'locktorture', may be built after the fact on the running kernel to be tested, if desired. The tests periodically output status messages via printk(), which can be examined via the dmesg (perhaps grepping for "torture"). The test is started when the module is loaded, and stops when the module is unloaded. This program is based on how RCU is tortured, via rcutorture.

This torture test consists of creating a number of kernel threads which acquire the lock and hold it for specific amount of time, thus simulating different critical region behaviors. The amount of contention on the lock can be simulated by either enlarging this critical region hold time and/or creating more kthreads.

#### **Module Parameters**

This module has the following parameters:

#### Locktorture-specific

nwriters stress

Number of kernel threads that will stress exclusive lock ownership (writers). The default value is twice the number of online CPUs.

nreaders stress

Number of kernel threads that will stress shared lock ownership (readers). The default is the same amount of writer locks. If the user did not specify nwriters stress, then both readers and writers be the amount of online CPUs.

torture type

Type of lock to torture. By default, only spinlocks will be tortured. This module can torture the following locks, with string values as follows:

• "lock busted":

Simulates a buggy lock implementation.

"spin lock":

spin\_lock() and spin\_unlock() pairs.

"spin\_lock irq":

spin lock irq() and spin unlock irq() pairs.

'rw\_lock":

read/write lock() and unlock() rwlock pairs.

"rw lock irq":

read/write lock irq() and unlock irq() rwlock pairs.

"mutex lock":

mutex lock() and mutex unlock() pairs.

• "rtmutex lock":

 $rtmutex\_lock() \ and \ rtmutex\_unlock() \ pairs. \ Kernel \ must \ have \ CONFIG\_RT\_MUTEX=y.$ 

• "rwsem lock":

read/write down() and up() semaphore pairs.

#### **Torture-framework (RCU + locking)**

shutdown secs

The number of seconds to run the test before terminating the test and powering off the system. The default is zero, which disables test termination and system shutdown. This capability is useful for automated testing.

onoff interval

The number of seconds between each attempt to execute a randomly selected CPU-hotplug operation. Defaults to zero, which disables CPU hotplugging. In CONFIG\_HOTPLUG\_CPU=n kernels, locktorture will silently refuse to do any CPU-hotplug operations regardless of what value is specified for onoff interval.

onoff\_holdoff

The number of seconds to wait until starting CPU-hotplug operations. This would normally only be used when locktorture was built into the kernel and started automatically at boot time, in which case it is useful in order to avoid confusing boot-time code with CPUs coming and going. This parameter is only useful if CONFIG HOTPLUG CPU is enabled.

stat interval

Number of seconds between statistics-related printk()s. By default, locktorture will report stats every 60 seconds. Setting the interval to zero causes the statistics to be printed -only- when the module is unloaded.

stutter

The length of time to run the test before pausing for this same period of time. Defaults to "stutter=5", so as to run and pause for (roughly) five-second intervals. Specifying "stutter=0" causes the test to run continuously without pausing. shuffle interval

The number of seconds to keep the test threads affinitied to a particular subset of the CPUs, defaults to 3 seconds. Used in conjunction with test no idle hz.

verbose

Enable verbose debugging printing, via printk(). Enabled by default. This extra information is mostly related to high-level errors and reports from the main 'torture' framework.

#### **Statistics**

Statistics are printed in the following format:

### Usage

The following script may be used to torture locks:

```
#!/bin/sh
modprobe locktorture
sleep 3600
rmmod locktorture
dmesg | grep torture:
```

The output can be manually inspected for the error flag of "!!!". One could of course create a more elaborate script that automatically checked for such errors. The "rmmod" command forces a "SUCCESS", "FAILURE", or "RCU\_HOTPLUG" indication to be printk()ed. The first two are self-explanatory, while the last indicates that while there were no locking failures, CPU-hotplug problems were detected.

Also see: Documentation/RCU/torture.rst