

Overview of SRCU

Kernel Summit 2013

Lai Jiangshan <laijs@cn.fujitsu.com>

What is SRCU

SRCU is a RCU variant which permits sleeping/blocking within read side critical sections

SRCU History

- Oct 2006, original version by Paul.
 - Fast read site and slow update site.
 - Update site needs 3 `synchronize_sched()` and waiting the sum of percpu counters to zero
- Feb/Mar 2012, Re-implemented by Paul & Lai(inspired by Peter Zijlstra).
 - Faster update site(no `synchronize_sched()`)
 - Adds `call_srcu()`.

When to Choose

- readers will need to block
- reader may be on some contexts which are not watched by RCU.
(idle loop or offlined CPU)
- need a separated RCU domain
 - SRCU isolates grace-period detection

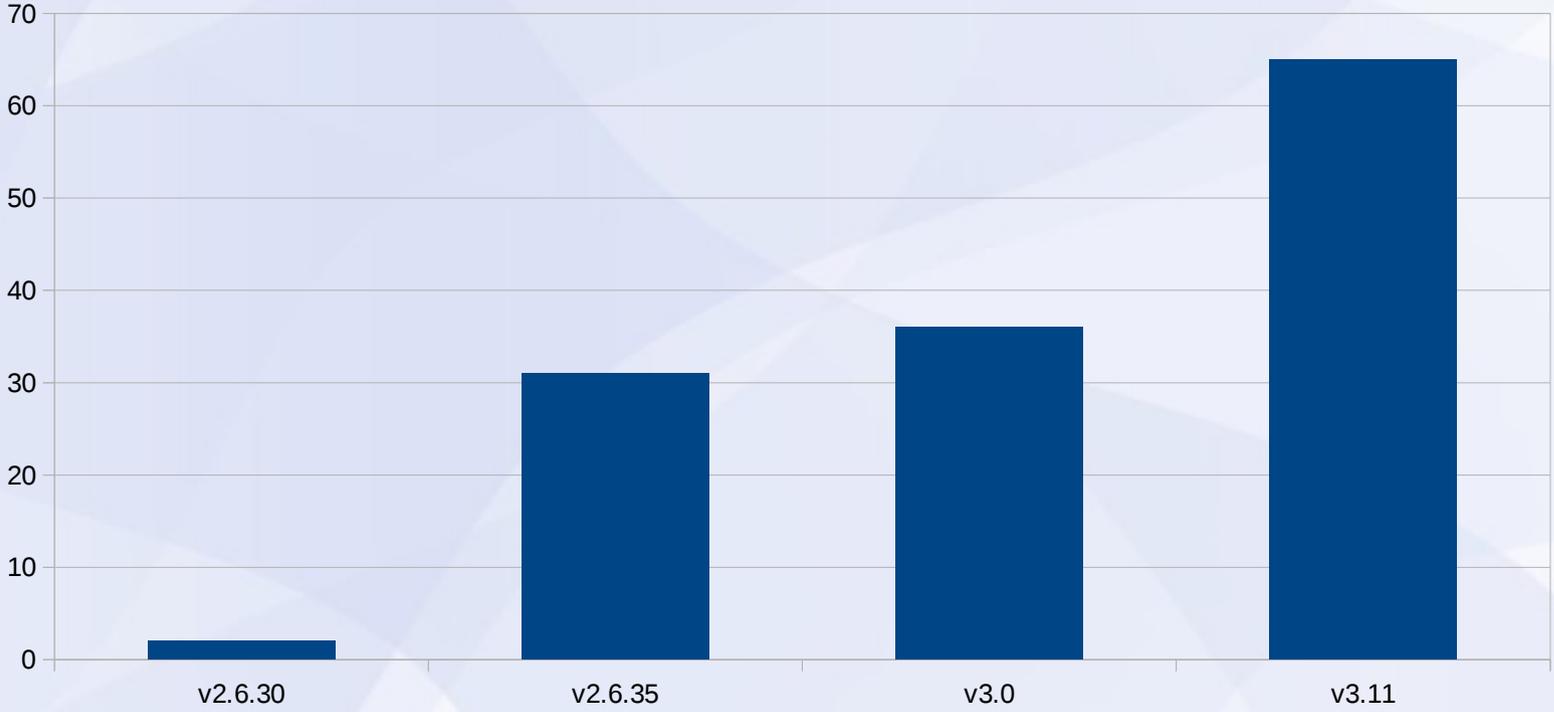
SRCU APIs

SRCU APIs	similar RCU APIs
init_srcu_struct()	
DEFINE_SRCU()	
DEFINE_STATIC_SRCU()	
cleanup_srcu_struct()	
srcu_read_lock()	rcu_read_lock()
srcu_read_unlock()	rcu_read_unlock()
srcu_dereference()	rcu_dereference()
synchronize_srcu()	synchronize_rcu()
synchronize_srcu_expedited()	synchronize_rcu_expedited()
call_srcu()	call_rcu()
srcu_barrier()	rcu_barrier()

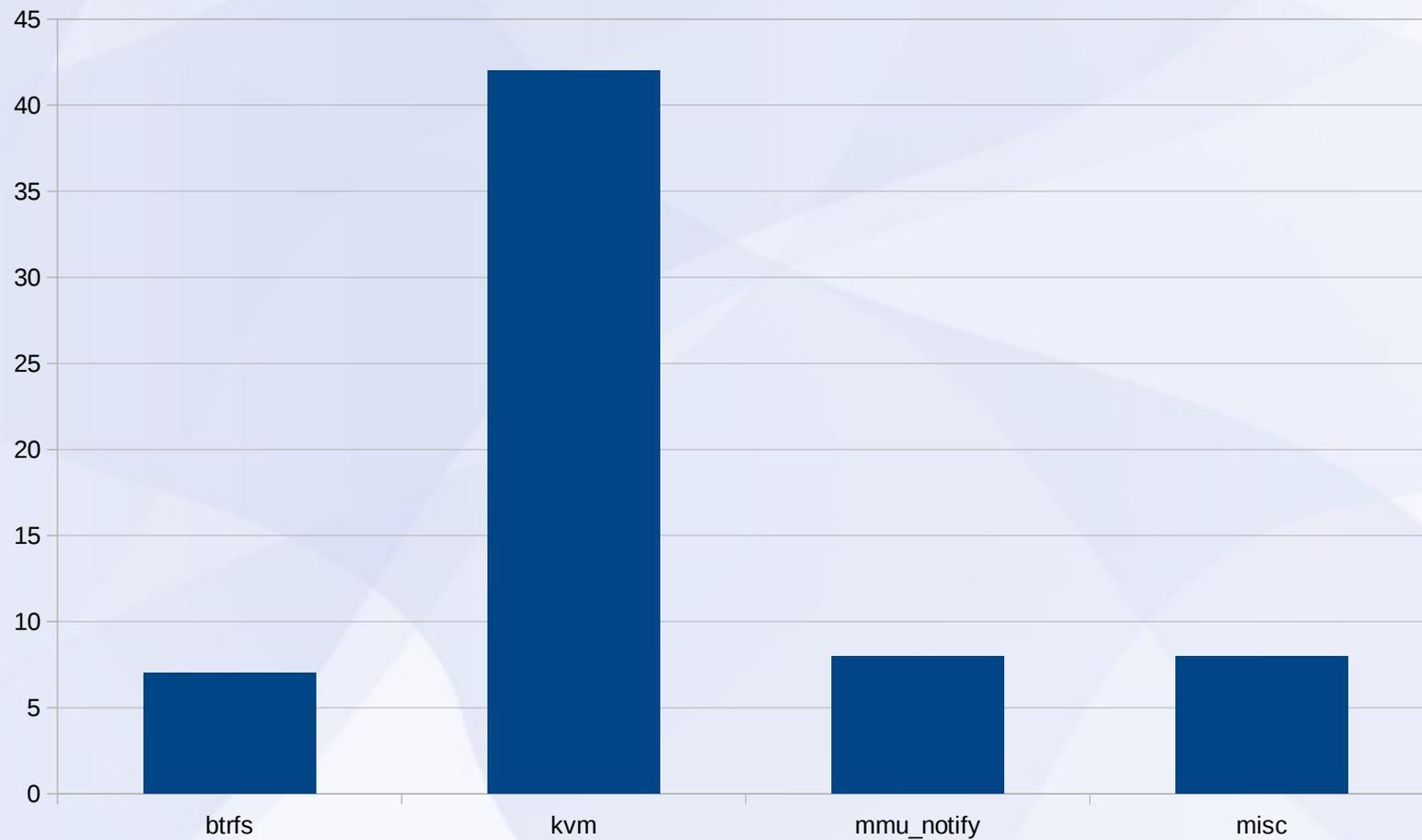
Usage

init	<pre>struct srcu_struct srcu; init_srcu_struct(&srcu);</pre>	<pre>DEFINE_SRCU(srcu);</pre>
reader	<pre>idx = srcu_read_lock(&srcu); tmp = srcu_dereference(gptr, &srcu); do_something_with(tmp); srcu_read_unlock(&srcu, idx);</pre>	
update	<pre>spin_lock(&update_lock); old = rcu_dereference_protected(gptr,1); rcu_assign_pointer(gptr, new); spin_unlock(&update_lock); synchronize_srcu(&srcu); kfree(old);</pre>	<pre>spin_lock(&update_lock); old = rcu_dereference_protected(gptr,1); rcu_assign_pointer(gptr, new); spin_unlock(&update_lock); call_srcu(&srcu, &old->head, free_after_srcu);</pre>

Using



Who is using SRCU



call_srcu()

- Invoke the func after srcu-GP
 - Don't implement call_srcu() by yourself

```
static int fsnotify_mark_destroy(void *ignored)
{
    struct fsnotify_mark *mark, *next;
    LIST_HEAD(private_destroy_list);

    for (;;) {
        spin_lock(&destroy_lock);
        /* exchange the list head */
        list_replace_init(&destroy_list, &private_destroy_list);
        spin_unlock(&destroy_lock);

        synchronize_srcu(&fsnotify_mark_srcu);

        list_for_each_entry_safe(mark, next, &private_destroy_list, destroy_list) {
            list_del_init(&mark->destroy_list);
            fsnotify_put_mark(mark);
        }

        wait_event_interruptible(destroy_waitq, !list_empty(&destroy_list));
    }

    return 0;
}
```

SRCU substitutions

- RCU/RCU_SCHED/RCU_BH
- reader/writer lock
- Reference
 - (RCU itself is a bulk reference mechanism)
 - Atomic/Kref
 - Scalable Rcuref (NEW)

Thanks

Any problem/suggestion are welcome!