

## Re: [PATCH/RFC] remove irq\_disabled warning from local\_bh\_enable

---

*Source:* <http://linux.derkeiler.com/Mailing-Lists/Kernel/2008-06/msg06888.html>

---

- *From:* Johannes Berg <[johannes@xxxxxxxxxxxxxxxxxxx](mailto:johannes@xxxxxxxxxxxxxxxxxxx)>
  - *Date:* Wed, 18 Jun 2008 09:29:37 +0200
- 

On Tue, 2008-06-17 at 16:55 -0700, Linus Torvalds wrote:

This warning has started to trigger with mac80211 because it can, under some circumstances, use spin\_lock\_bh() protected sections within irq-disabled sections. Is that a bug?

Yes, it's a bug.

You know, that actually makes sense to me.

Why? Not because of the "spin\_lock\_bh()" itself, but because of the \_unlock\_, which does a "local\_bh\_enable\_ip()", which in turn will check the whole "do\_softirq()" if it was the last softirq\_count.

Yeah, that makes sense to me as well. In fact, I pondered a bit over the code until posting this.

And you must not do softirq's when hard-irq's were disabled!

So it should in theory be ok (but perhaps a bit odd) to do something like

```
spin_lock_irq(&irq_lock);
..do something..
spin_lock_bh(&bh_lock);
spin_unlock_irq(&irq_lock);
.. do something else ..
spin_unlock_bh(&bh_lock);
```

where the "spin\_lock\_bh()" itself is in an irq-locked context – as long as the "spin\_unlock\_bh()" is \*not\*.

See?

Re: [PATCH/RFC] remove irqs\_disabled warning from local\_bh\_enable

Well, yes, from an API POV I do see that. And it's actually what I thought up front. The thing that started to confuse me is that local\_bh\_enable\_ip reads like this:

```
void local_bh_enable_ip(unsigned long ip)
{
#ifdef CONFIG_TRACE_IRQFLAGS
unsigned long flags;

WARN_ON_ONCE(in_irq());

local_irq_save(flags);
#endif

[...]
if (unlikely(!in_interrupt() && local_softirq_pending()))
do_softirq();

dec_preempt_count();
#ifdef CONFIG_TRACE_IRQFLAGS
local_irq_restore(flags);
#endif
```

So doesn't it actually run do\_softirq() when interrupts are still disabled? If I look deeper into the code that appears to be a rather odd trick to avoid locking and \_\_do\_softirq will unconditionally enable IRQs, so I guess here's my answer...

How about the patch below?

johannes

From: Johannes Berg <johannes@xxxxxxxxxxxxxxxxxxx>  
Subject: clean up and comment local\_bh\_enable code

There's no need to use local\_irq\_save() over local\_irq\_disable() in the local\_bh\_enable code since it is a bug to call it with irqs disabled and do\_softirq will enable irqs if there is any pending work. Consolidate the code from local\_bh\_enable and ...\_ip to avoid having a disconnect between them in the warnings they trigger that is currently there. Also always trigger the warning on in\_irq(), not just in the trace-irqflags case.

Signed-off-by: Johannes Berg <johannes@xxxxxxxxxxxxxxxxxxx>

---  
I made it a static inline so that runtime behaviour isn't affected by another function call, but that can be argued.

Maybe local\_bh\_enable() should just be a static inline somewhere in a header file, that would get rid of one export\_symbol as well.

Re: [PATCH/RFC] remove irqs\_disabled warning from local\_bh\_enable

Re: [PATCH/RFC] remove irq\_disabled warning from local\_bh\_enable

kernel/softirq.c | 48 ++++++++-----  
1 file changed, 11 insertions(+), 37 deletions(-)

--- everything.orig/kernel/softirq.c 2008-06-17 23:48:25.000000000 +0200  
+++ everything/kernel/softirq.c 2008-06-18 09:13:21.000000000 +0200  
@@ -131,23 +131,17 @@ void \_local\_bh\_enable(void)

EXPORT\_SYMBOL(\_local\_bh\_enable);

```
-void local_bh_enable(void)
+static inline void _local_bh_enable_ip(unsigned long ip)
{
+ WARN_ON_ONCE(in_irq() || irq_disabled());
#ifdef CONFIG_TRACE_IRQFLAGS
- unsigned long flags;
-
- WARN_ON_ONCE(in_irq());
-#endif
- WARN_ON_ONCE(irq_disabled());
-
-#ifdef CONFIG_TRACE_IRQFLAGS
- local_irq_save(flags);
+ local_irq_disable();
#endif
/*
* Are softirqs going to be turned on now:
*/
if (softirq_count() == SOFTIRQ_OFFSET)
- trace_softirqs_on((unsigned long)__builtin_return_address(0));
+ trace_softirqs_on(ip);
/*
* Keep preemption disabled until we are done with
* softirq processing:
@@ -159,40 +153,20 @@ void local_bh_enable(void)
```

```
dec_preempt_count();
#ifdef CONFIG_TRACE_IRQFLAGS
- local_irq_restore(flags);
+ local_irq_enable();
#endif
preempt_check_resched();
}
+
+void local_bh_enable(void)
+{
+ _local_bh_enable_ip((unsigned long)__builtin_return_address(0));
+}
EXPORT_SYMBOL(local_bh_enable);
```

```
void local_bh_enable_ip(unsigned long ip)
```

Re: [PATCH/RFC] remove irq\_disabled warning from local\_bh\_enable

```
{
-#ifdef CONFIG_TRACE_IRQFLAGS
- unsigned long flags;
-
- WARN_ON_ONCE(in_irq());
-
- local_irq_save(flags);
-#endif
- /*
- * Are softirqs going to be turned on now:
- */
- if (softirq_count() == SOFTIRQ_OFFSET)
- trace_softirqs_on(ip);
- /*
- * Keep preemption disabled until we are done with
- * softirq processing:
- */
- sub_preempt_count(SOFTIRQ_OFFSET - 1);
-
- if (unlikely(!in_interrupt() && local_softirq_pending()))
- do_softirq();
-
- dec_preempt_count();
-#ifdef CONFIG_TRACE_IRQFLAGS
- local_irq_restore(flags);
-#endif
- preempt_check_resched();
+ _local_bh_enable_ip(ip);
}
EXPORT_SYMBOL(local_bh_enable_ip);
```

--  
To unsubscribe from this list: send the line "unsubscribe linux-kernel" in  
the body of a message to majordomo@xxxxxxxxxxxxxxxxx  
More majordomo info at <http://vger.kernel.org/majordomo-info.html>  
Please read the FAQ at <http://www.tux.org/lkml/>

Re: [PATCH/RFC] remove irq\_disabled warning from local\_bh\_enable