Subject: [PATCH COMMIT] diff-smp-nmi-show-regs
Posted by xemul on Wed, 05 Apr 2006 10:30:26 GMT
View Forum Message <> Reply to Message

Added to 026test008

Patch from OpenVZ team <devel@openvz.org>
This patch adds dumping of calltraces on _all_ CPUs on AltSysRq-P and NMI
LOCKUP. It does this via sending NMI IPI interrupts to the cpus.
Taken back from -mm tree.

Signed-off-by: Kirill Korotaev <dev@sw.ru>
Signed-off-by: Pavel Emelianov <xemul@sw.ru>
Signed-off-by: Andrew Morton <akpm@osdl.org>

```
diff -puN  arch/i386/kernel/nmi.c~nmi-lockup-and-altsysrq-p-dumping-cal ltraces-on-_all_-cpus
arch/i386/kernel/nmi.c
---  devel/arch/i386/kernel/nmi.c~nmi-lockup-and-altsysrq-p-dumpi
ng-calltraces-on-_all_-cpus 2006-03-18 00:55:33.000000000 -0800
+++ devel-akpm/arch/i386/kernel/nmi.c 2006-03-18 01:11:52.000000000 -0800
@@ -523,6 +523,21 @@ void touch_nmi_watchdog (void)

 extern void die_nmi(struct pt_regs *, const char *msg);

+static spinlock_t show_regs_lock = SPIN_LOCK_UNLOCKED;
+
+void smp_show_regs(struct pt_regs *regs, void *info)
+{
+ if (regs == NULL)
+  return;
+
+ bust_spinlocks(1);
+ spin_lock(&show_regs_lock);
+ printk("----------- IPI show regs -----------");
+ show_regs(regs);
+ spin_unlock(&show_regs_lock);
+ bust_spinlocks(0);
+}
+
 void nmi_watchdog_tick (struct pt_regs * regs)
 {

diff -puN  arch/i386/kernel/smp.c~nmi-lockup-and-altsysrq-p-dumping-cal ltraces-on-_all_-cpus
arch/i386/kernel/smp.c
---  devel/arch/i386/kernel/smp.c~nmi-lockup-and-altsysrq-p-dumpi
ng-calltraces-on-_all_-cpus 2006-03-18 00:55:33.000000000 -0800
+++ devel-akpm/arch/i386/kernel/smp.c 2006-03-18 00:55:33.000000000 -0800
```

```
@@ -21,6 +21,7 @@
 #include <linux/cpu.h>
 #include <linux/module.h>

+#include <asm/nmi.h>
 #include <asm/mtrr.h>
 #include <asm/tlbflush.h>
 #include <mach_apic.h>
@@ -562,6 +563,89 @@ int smp_call_function (void (*func) (voi
 }
 EXPORT_SYMBOL(smp_call_function);

+static spinlock_t nmi_call_lock = SPIN_LOCK_UNLOCKED;
+static struct nmi_call_data_struct {
+ smp_nmi_function func;
+ void *info;
+ atomic_t started;
+ atomic_t finished;
+ cpumask_t cpus_called;
+ int wait;
+} *nmi_call_data;
+
+static int smp_nmi_callback(struct pt_regs * regs, int cpu)
+{
+ smp_nmi_function func;
+ void *info;
+ int wait;
+
+ func = nmi_call_data->func;
+ info = nmi_call_data->info;
+ wait = nmi_call_data->wait;
+ ack_APIC_irq();
+ /* prevent from calling func() multiple times */
+ if (cpu_test_and_set(cpu, nmi_call_data->cpus_called))
+  return 0;
+ /*
+  * notify initiating CPU that I've grabbed the data and am
+  * about to execute the function
+  */
+ mb();
+ atomic_inc(&nmi_call_data->started);
+ /* at this point the nmi_call_data structure is out of scope */
+ irq_enter();
+ func(regs, info);
+ irq_exit();
+ if (wait)
+  atomic_inc(&nmi_call_data->finished);
+
```

```
+ return 0;
+}
+
+/*
+ * This function tries to call func(regs, info) on each cpu.
+ * Func must be fast and non-blocking.
+ * May be called with disabled interrupts and from any context.
+ */
+int smp_nmi_call_function(smp_nmi_function func, void *info, int wait)
+{
+ struct nmi_call_data_struct data;
+ int cpus;
+
+ cpus = num_online_cpus() - 1;
+ if (!cpus)
+  return 0;
+
+ data.func = func;
+ data.info = info;
+ data.wait = wait;
+ atomic_set(&data.started, 0);
+ atomic_set(&data.finished, 0);
+ cpus_clear(data.cpus_called);
+ /* prevent this cpu from calling func if NMI happens */
+ cpu_set(smp_processor_id(), data.cpus_called);
+
+ if (!spin_trylock(&nmi_call_lock))
+  return -1;
+
+ nmi_call_data = &data;
+ set_nmi_ipi_callback(smp_nmi_callback);
+ mb();
+
+ /* Send a message to all other CPUs and wait for them to respond */
+ send_IPI_allbutself(APIC_DM_NMI);
+ while (atomic_read(&data.started) != cpus)
+  barrier();
+
+ unset_nmi_ipi_callback();
+ if (wait)
+  while (atomic_read(&data.finished) != cpus)
+   barrier();
+ spin_unlock(&nmi_call_lock);
+
+ return 0;
+}
+
 static void stop_this_cpu (void * dummy)
```

```
 {
  /*
diff -puN  arch/i386/kernel/traps.c~nmi-lockup-and-altsysrq-p-dumping-c alltraces-on-_all_-cpus
arch/i386/kernel/traps.c
---  devel/arch/i386/kernel/traps.c~nmi-lockup-and-altsysrq-p-dum
ping-calltraces-on-_all_-cpus 2006-03-18 00:55:33.000000000 -0800
+++ devel-akpm/arch/i386/kernel/traps.c 2006-03-18 01:11:52.000000000 -0800
@@ -689,6 +689,8 @@ void die_nmi (struct pt_regs *regs, cons
  printk(" on CPU%d, eip %08lx, registers:\n",
   smp_processor_id(), regs->eip);
  show_registers(regs);
+ smp_nmi_call_function(smp_show_regs, NULL, 1);
+ bust_spinlocks(1);
  printk(KERN_EMERG "console shuts up ...\n");
  console_silent();
  spin_unlock(&nmi_print_lock);
@@ -705,6 +707,14 @@ void die_nmi (struct pt_regs *regs, cons
  do_exit(SIGSEGV);
 }

+static int dummy_nmi_callback(struct pt_regs * regs, int cpu)
+{
+ return 0;
+}
+
+static nmi_callback_t nmi_callback = dummy_nmi_callback;
+static nmi_callback_t nmi_ipi_callback = dummy_nmi_callback;
+
 static void default_do_nmi(struct pt_regs * regs)
 {
  unsigned char reason = 0;
@@ -727,6 +737,9 @@ static void default_do_nmi(struct pt_reg
    return;
   }
 #endif
+  if (nmi_ipi_callback != dummy_nmi_callback)
+   return;
+
   unknown_nmi_error(reason, regs);
   return;
  }
@@ -743,13 +756,6 @@ static void default_do_nmi(struct pt_reg
  reassert_nmi();
 }

-static int dummy_nmi_callback(struct pt_regs * regs, int cpu)
-{
- return 0;
```

```
-}
-
-static nmi_callback_t nmi_callback = dummy_nmi_callback;
-
 fastcall void do_nmi(struct pt_regs * regs, long error_code)
 {
  int cpu;
@@ -763,9 +769,20 @@ fastcall void do_nmi(struct pt_regs * re
  if (!rcu_dereference(nmi_callback)(regs, cpu))
   default_do_nmi(regs);

+ nmi_ipi_callback(regs, cpu);
  nmi_exit();
 }

+void set_nmi_ipi_callback(nmi_callback_t callback)
+{
+ nmi_ipi_callback = callback;
+}
+
+void unset_nmi_ipi_callback(void)
+{
+ nmi_ipi_callback = dummy_nmi_callback;
+}
+
 void set_nmi_callback(nmi_callback_t callback)
 {
  rcu_assign_pointer(nmi_callback, callback);
diff -puN  drivers/char/sysrq.c~nmi-lockup-and-altsysrq-p-dumping-callt races-on-_all_-cpus
drivers/char/sysrq.c
---  devel/drivers/char/sysrq.c~nmi-lockup-and-altsysrq-p-dumping
-calltraces-on-_all_-cpus 2006-03-18 00:55:33.000000000 -0800
+++ devel-akpm/drivers/char/sysrq.c 2006-03-18 00:55:33.000000000 -0800
@@ -183,8 +183,13 @@ static struct sysrq_key_op sysrq_showloc
 static void sysrq_handle_showregs(int key, struct pt_regs *pt_regs,
       struct tty_struct *tty)
 {
+ bust_spinlocks(1);
  if (pt_regs)
   show_regs(pt_regs);
+ bust_spinlocks(0);
+#ifdef __i386__
+ smp_nmi_call_function(smp_show_regs, NULL, 0);
+#endif
 }
 static struct sysrq_key_op sysrq_showregs_op = {
  .handler = sysrq_handle_showregs,
diff -puN  include/asm-i386/nmi.h~nmi-lockup-and-altsysrq-p-dumping-cal ltraces-on-_all_-cpus
```

```
include/asm-i386/nmi.h
---  devel/include/asm-i386/nmi.h~nmi-lockup-and-altsysrq-p-dumpi
ng-calltraces-on-_all_-cpus 2006-03-18 00:55:33.000000000 -0800
+++ devel-akpm/include/asm-i386/nmi.h 2006-03-18 00:55:33.000000000 -0800
@@ -17,6 +17,7 @@ typedef int (*nmi_callback_t)(struct pt_
  * set. Return 1 if the NMI was handled.
  */
 void set_nmi_callback(nmi_callback_t callback);
+void set_nmi_ipi_callback(nmi_callback_t callback);

 /**
  * unset_nmi_callback
@@ -24,5 +25,6 @@ void set_nmi_callback(nmi_callback_t cal
  * Remove the handler previously set.
  */
 void unset_nmi_callback(void);
+void unset_nmi_ipi_callback(void);

 #endif /* ASM_NMI_H */
diff -puN  include/linux/sched.h~nmi-lockup-and-altsysrq-p-dumping-call traces-on-_all_-cpus
include/linux/sched.h
---  devel/include/linux/sched.h~nmi-lockup-and-altsysrq-p-dumpin
g-calltraces-on-_all_-cpus 2006-03-18 00:55:33.000000000 -0800
+++ devel-akpm/include/linux/sched.h 2006-03-18 00:55:33.000000000 -0800
@@ -193,6 +193,7 @@ extern cpumask_t nohz_cpu_mask;

 extern void show_state(void);
 extern void show_regs(struct pt_regs *);
+extern void smp_show_regs(struct pt_regs *, void *);

 /*
  * TASK is a pointer to the task whose backtrace we want to see (or NULL for current
diff -puN  include/linux/smp.h~nmi-lockup-and-altsysrq-p-dumping-calltr aces-on-_all_-cpus
include/linux/smp.h
---  devel/include/linux/smp.h~nmi-lockup-and-altsysrq-p-dumping-
calltraces-on-_all_-cpus 2006-03-18 00:55:33.000000000 -0800
+++ devel-akpm/include/linux/smp.h 2006-03-18 01:12:41.000000000 -0800
@@ -10,6 +10,9 @@

 extern void cpu_idle(void);

+struct pt_regs;
+typedef void (*smp_nmi_function)(struct pt_regs *regs, void *info);
+
 #ifdef CONFIG_SMP

 #include <linux/preempt.h>
@@ -49,6 +52,8 @@ extern int __cpu_up(unsigned int cpunum)
```

```
 */
 extern void smp_cpus_done(unsigned int max_cpus);

+extern int smp_nmi_call_function(smp_nmi_function func, void *info, int wait);
+
 /*
  * Call a function on all other processors
  */
@@ -94,6 +99,12 @@ static inline void smp_send_reschedule(i
 #define num_booting_cpus()   1
 #define smp_prepare_boot_cpu()   do {} while (0)

+static inline int smp_nmi_call_function(smp_nmi_function func,
+       void *info, int wait)
+{
+ return 0;
+}
+
 #endif /* !SMP */

 /*
```

File Attachments

1) diff-smp-nmi-show-regs, downloaded 483 times