
Subject: [PATCH COMMIT] diff-merge-2.6.16.9-20060424

Posted by [xemul](#) on Mon, 24 Apr 2006 14:44:09 GMT

[View Forum Message](#) <> [Reply to Message](#)

Added to 026test010

Patch from OpenVZ team <devel@openvz.org>

Merge 2.6.16.9 from /linux/kernel/git/stable/linux-2.6.16.y

From: OpenVZ team <devel@openvz.org>

Date: Mon, 24 Apr 2006 14:41:18 +0000 (+0400)

Subject: Merge 2.6.16.9 from /linux/kernel/git/stable/linux-2.6.16.y

X-Git-Url:

<http://10.11.12.17/cgi-bin/gitweb.cgi?p=kernel;a=commitdiff;h=35ceb3e3ef2337a0d9639297f708a588c1073bd2>

Merge 2.6.16.9 from /linux/kernel/git/stable/linux-2.6.16.y

--- a/arch/i386/kernel/cpu/amd.c

+++ b/arch/i386/kernel/cpu/amd.c

@@ -207,6 +207,8 @@ static void __init init_amd(struct cpuin

set_bit(X86_FEATURE_K7, c->x86_capability);

break;

}

+ if (c->x86 >= 6)

+ set_bit(X86_FEATURE_FXSAVE_LEAK, c->x86_capability);

display_cacheinfo(c);

--- a/arch/x86_64/kernel/process.c

+++ b/arch/x86_64/kernel/process.c

@@ -548,8 +548,6 @@ __switch_to(struct task_struct *prev_p,

int cpu = smp_processor_id());

struct tss_struct *tss = &per_cpu(init_tss, cpu);

- unlazy_fpu(prev_p);

-

/*

* Reload esp0, LDT and the page table pointer:

*/

@@ -612,6 +610,12 @@ __switch_to(struct task_struct *prev_p,

prev->useresp = read_pda(oldrsp);

write_pda(oldrsp, next->useresp);

write_pda(pcurrent, next_p);

+

+ /* This must be here to ensure both math_state_restore() and

```

+ kernel_fpu_begin() work consistently.
+ And the AMD workaround requires it to be after DS reload. */
+ unlazy_fpu(prev_p);
+
+ write_pda(kernelstack,
+   task_stack_page(next_p) + THREAD_SIZE - PDA_STACKOFFSET);

--- a/arch/x86_64/kernel/setup.c
+++ b/arch/x86_64/kernel/setup.c
@@ -909,6 +909,10 @@ static int __init init_amd(struct cpuinfo
+ if (c->x86 == 15 && ((level >= 0x0f48 && level < 0x0f50) || level >= 0x0f58))
+   set_bit(X86_FEATURE_REP_GOOD, &c->x86_capability);

+ /* Enable workaround for FXSAVE leak */
+ if (c->x86 >= 6)
+   set_bit(X86_FEATURE_FXSAVE_LEAK, &c->x86_capability);
+
+   r = get_model_name(c);
+   if (!r) {
+     switch (c->x86) {
--- a/include/asm-i386/cpufeature.h
+++ b/include/asm-i386/cpufeature.h
@@ -70,6 +70,7 @@
+ #define X86_FEATURE_P3 (3*32+ 6) /* P3 */
+ #define X86_FEATURE_P4 (3*32+ 7) /* P4 */
+ #define X86_FEATURE_CONSTANT_TSC (3*32+ 8) /* TSC ticks at a constant rate */
+ #define X86_FEATURE_FXSAVE_LEAK (3*32+10) /* FXSAVE leaks FOP/FIP/FOP */

/* Intel-defined CPU features, CPUID level 0x00000001 (ecx), word 4 */
+ #define X86_FEATURE_XMM3 (4*32+ 0) /* Streaming SIMD Extensions-3 */
--- a/include/asm-i386/i387.h
+++ b/include/asm-i386/i387.h
@@ -13,6 +13,7 @@

+ #include <linux/sched.h>
+ #include <linux/init.h>
+ #include <linux/kernel_stat.h>
+ #include <asm/processor.h>
+ #include <asm/sigcontext.h>
+ #include <asm/user.h>
@@ -38,17 +39,38 @@ extern void init_fpu(struct task_struct
+ extern void kernel_fpu_begin(void);
+ #define kernel_fpu_end() do { stts(); preempt_enable(); } while(0)

+ /* We need a safe address that is cheap to find and that is already
+  + in L1 during context switch. The best choices are unfortunately
+  + different for UP and SMP */
+ #ifdef CONFIG_SMP

```

```

+#define safe_address (__per_cpu_offset[0])
+#else
+#define safe_address (kstat_cpu(0).cpustat.user)
+#endif
+
+/*
+ * These must be called with preempt disabled
+ */
static inline void __save_init_fpu( struct task_struct *tsk )
{
+ /* Use more nops than strictly needed in case the compiler
+  varies code */
  alternative_input(
- "fnsave %1 ; fwait ;" GENERIC_NOP2,
- "fxsave %1 ; fnclex",
+ "fnsave %[fx] ; fwait;" GENERIC_NOP8 GENERIC_NOP4,
+ "fxsave %[fx]\n"
+ "bt $7,%[fsw] ; jc 1f ; fnclex\n1:",
  X86_FEATURE_FXSR,
- "m" (tsk->thread.i387.fxsave)
- : "memory");
+ [fx] "m" (tsk->thread.i387.fxsave),
+ [fsw] "m" (tsk->thread.i387.fxsave.swd) : "memory");
+ /* AMD K7/K8 CPUs don't save/restore FDP/FIP/FOP unless an exception
+  is pending. Clear the x87 state here by setting it to fixed
+  values. __per_cpu_offset[0] is a random variable that should be in L1 */
+ alternative_input(
+ GENERIC_NOP8 GENERIC_NOP2,
+ "emms\n\t" /* clear stack tags */
+ "fildl %[addr]", /* set F?P to defined value */
+ X86_FEATURE_FXSAVE_LEAK,
+ [addr] "m" (safe_address));
  task_thread_info(tsk)->status &= ~TS_USEDFFPU;
}

--- a/include/asm-x86_64/cpufeature.h
+++ b/include/asm-x86_64/cpufeature.h
@@ -64,6 +64,7 @@
#define X86_FEATURE_REP_GOOD (3*32+ 4) /* rep microcode works well on this CPU */
#define X86_FEATURE_CONSTANT_TSC (3*32+5) /* TSC runs at constant rate */
#define X86_FEATURE_SYNC_RDTSC (3*32+6) /* RDTSC syncs CPU core */
+#define X86_FEATURE_FXSAVE_LEAK (3*32+7) /* FIP/FOP/FDP leaks through FXSAVE */

/* Intel-defined CPU features, CPUID level 0x00000001 (ecx), word 4 */
#define X86_FEATURE_XMM3 (4*32+ 0) /* Streaming SIMD Extensions-3 */
--- a/include/asm-x86_64/i387.h
+++ b/include/asm-x86_64/i387.h
@@ -72,6 +72,23 @@ extern int set_fpregs(struct task_struct

```

```

#define set_fpu_swd(t,val) ((t)->thread.i387.fxsave.swd = (val))
#define set_fpu_fxr_twd(t,val) ((t)->thread.i387.fxsave.twd = (val))

+#define X87_FSW_ES (1 << 7) /* Exception Summary */
+
+/* AMD CPUs don't save/restore FDP/FIP/FOP unless an exception
+ is pending. Clear the x87 state here by setting it to fixed
+ values. The kernel data segment can be sometimes 0 and sometimes
+ new user value. Both should be ok.
+ Use the PDA as safe address because it should be already in L1. */
+static inline void clear_fpu_state(struct i387_fxsave_struct *fx)
+{
+ if (unlikely(fx->swd & X87_FSW_ES))
+  asm volatile("fnclex");
+ alternative_input(ASM_NOP8 ASM_NOP2,
+   " emms\n" /* clear stack tags */
+   " fildl %%gs:0", /* load to clear state */
+   X86_FEATURE_FXSAVE_LEAK);
+}
+
+static inline int restore_fpu_checking(struct i387_fxsave_struct *fx)
+{
+ int err;
@@ -119,6 +136,7 @@ static inline int save_i387_checking(str
#endif
+ if (unlikely(err))
+  __clear_user(fx, sizeof(struct i387_fxsave_struct));
+ /* No need to clear here because the caller clears USED_MATH */
+ return err;
+}

@@ -149,7 +167,7 @@ static inline void __fxsave_clear(struct
+ "i" (offsetof(__typeof__(*tsk),
+   thread.i387.fxsave)));
#endif
- __asm__ __volatile__("fnclex");
+ clear_fpu_state(&tsk->thread.i387.fxsave);
+}

static inline void kernel_fpu_begin(void)

```

File Attachments

1) [diff-merge-2.6.16.9-20060424](#), downloaded 420 times
