Subject: Re: [PATCH RESEND 2/2] Fix some kallsyms_lookup() vs rmmod races
Posted by Paulo Marques on Fri, 16 Mar 2007 16:16:10 GMT
View Forum Message <> Reply to Message

Ingo Molnar wrote:
> * Alexey Dobriyan <adobriyan@sw.ru> wrote:
>
>> [cc'ing folks whose proc files are affected]
>>
>> kallsyms_lookup() can call module_address_lookup() which iterates over
>> modules list without module_mutex taken. Comment at the top of
>> module_address_lookup() says it's for oops resolution so races are
>> irrelevant, but in some cases it's reachable from regular code:
>
> looking at the problem from another angle: wouldnt this be something
> that would benefit from freeze_processes()/unfreeze_processes(), and
> hence no locking would be required?

I also considered this, but it seemed a little too "blunt" to stop
everything (including completely unrelated processes and kernel threads)
just to remove a module.

How about something like this instead? (just for review)

It's a little more intrusive, since the interface for symbol lookup is
changed (and it affects the callers), but it makes the caller aware that
it should set "safe_to_lock" if possible.

This way we avoid exposing module_mutex outside of module.c and avoid
returning any data pointer to some data structure that might disappear
before the caller tries to use it.

Some of these changes are actually cleanups, because the callers where
creating a dummy modname variables that they didn't use afterwards.

The thing I like the less about this patch is the increase of stack
usage on some code paths by about 60 bytes.

Anyway, I don't have very strong feelings about this, so if you think it
would be better to use freeze_processes()/unfreeze_processes(), I can
cook up a patch for that too...

--
Paulo Marques - www.grupopie.com

"Very funny Scotty. Now beam up my clothes."

diffstat:

```
>  arch/parisc/kernel/unwind.c |    3 --
>  arch/powerpc/xmon/xmon.c    |   11 ++++-----
>  arch/ppc/xmon/xmon.c        |    8 +++---
>  arch/sh64/kernel/unwind.c   |    4 +--
>  arch/x86_64/kernel/traps.c  |   10 ++++----
>  fs/proc/base.c              |    3 --
>  include/linux/kallsyms.h    |    6 +++-
>  include/linux/module.h      |   44 ++++++++++++++++++++----------------
>  kernel/kallsyms.c           |   53 ++++++++++++++++++++++++++++++++++++++++++++
>  kernel/kprobes.c            |    6 ++--
>  kernel/lockdep.c            |    3 --
>  kernel/module.c             |   44 ++++++++++++++++++++++++++++++++++++++++
>  kernel/time/timer_list.c    |    3 --
>  kernel/time/timer_stats.c   |    3 --
>  mm/slab.c                   |    6 ++--
>  15 files changed, 114 insertions(+), 93 deletions(-)


diff --git a/arch/parisc/kernel/unwind.c b/arch/parisc/kernel/unwind.c
--- a/arch/parisc/kernel/unwind.c
+++ b/arch/parisc/kernel/unwind.c
@@ -216,11 +216,10 @@ static void unwind_frame_regs(struct unw
   /* Handle some frequent special cases.... */
   {
    char symname[KSYM_NAME_LEN+1];
-   char *modname;
    unsigned long symsize, offset;

    kallsyms_lookup(info->ip, &symsize, &offset,
-     &modname, symname);
+     NULL, symname, 0);

    dbg("info->ip = 0x%lx, name = %s\n", info->ip, symname);

diff --git a/arch/powerpc/xmon/xmon.c b/arch/powerpc/xmon/xmon.c
--- a/arch/powerpc/xmon/xmon.c
+++ b/arch/powerpc/xmon/xmon.c
@@ -1218,7 +1218,6 @@ static void get_function_bounds(unsigned
 {
  unsigned long size, offset;
  const char *name;
- char *modname;

  *startp = *endp = 0;
  if (pc == 0)
@@ -1226,7 +1225,7 @@ static void get_function_bounds(unsigned
  if (setjmp(bus_error_jmp) == 0) {
   catch_memory_errors = 1;
```

```
   sync();
-  name = kallsyms_lookup(pc, &size, &offset, &modname, tmpstr);
+  name = kallsyms_lookup(pc, &size, &offset, NULL, tmpstr, 0);
   if (name != NULL) {
    *startp = pc - offset;
    *endp = pc - offset + size;
@@ -2496,7 +2495,7 @@ symbol_lookup(void)
 static void xmon_print_symbol(unsigned long address, const char *mid,
         const char *after)
 {
- char *modname;
+ char modname[MODULE_NAME_LEN + 1];
  const char *name = NULL;
  unsigned long offset, size;

@@ -2504,8 +2503,8 @@ static void xmon_print_symbol(unsigned l
  if (setjmp(bus_error_jmp) == 0) {
   catch_memory_errors = 1;
   sync();
-  name = kallsyms_lookup(address, &size, &offset, &modname,
-         tmpstr);
+  name = kallsyms_lookup(address, &size, &offset, modname,
+         tmpstr, 0);
   sync();
   /* wait a little while to see if we get a machine check */
   __delay(200);
@@ -2515,7 +2514,7 @@ static void xmon_print_symbol(unsigned l

  if (name) {
   printf("%s%s+%#lx/%#lx", mid, name, offset, size);
-  if (modname)
+  if (modname[0])
    printf(" [%s]", modname);
  }
  printf("%s", after);
diff --git a/arch/ppc/xmon/xmon.c b/arch/ppc/xmon/xmon.c
--- a/arch/ppc/xmon/xmon.c
+++ b/arch/ppc/xmon/xmon.c
@@ -177,7 +177,7 @@ extern inline void __delay(unsigned int
 static void xmon_print_symbol(unsigned long address, const char *mid,
         const char *after)
 {
- char *modname;
+ char modname[MODULE_NAME_LEN+1];
  const char *name = NULL;
  unsigned long offset, size;
  static char tmpstr[128];
@@ -186,8 +186,8 @@ static void xmon_print_symbol(unsigned l
```

```
  if (setjmp(bus_error_jmp) == 0) {
   debugger_fault_handler = handle_fault;
   sync();
-  name = kallsyms_lookup(address, &size, &offset, &modname,
-           tmpstr);
+  name = kallsyms_lookup(address, &size, &offset, modname,
+           tmpstr, 0);
   sync();
   /* wait a little while to see if we get a machine check */
   __delay(200);
@@ -196,7 +196,7 @@ static void xmon_print_symbol(unsigned l

  if (name) {
   printf("%s%s+%#lx/%#lx", mid, name, offset, size);
-  if (modname)
+  if (modname[0])
    printf(" [%s]", modname);
  }
  printf("%s", after);
diff --git a/arch/sh64/kernel/unwind.c b/arch/sh64/kernel/unwind.c
--- a/arch/sh64/kernel/unwind.c
+++ b/arch/sh64/kernel/unwind.c
@@ -46,7 +46,7 @@ static int lookup_prev_stack_frame(unsig
        struct pt_regs *regs)
 {
  const char *sym;
- char *modname, namebuf[128];
+ char namebuf[128];
  unsigned long offset, size;
  unsigned long prologue = 0;
  unsigned long fp_displacement = 0;
@@ -54,7 +54,7 @@ static int lookup_prev_stack_frame(unsig
  unsigned long offset_r14 = 0, offset_r18 = 0;
  int i, found_prologue_end = 0;

- sym = kallsyms_lookup(pc, &size, &offset, &modname, namebuf);
+ sym = kallsyms_lookup(pc, &size, &offset, NULL, namebuf, 0);
  if (!sym)
   return -EINVAL;

diff --git a/arch/x86_64/kernel/traps.c b/arch/x86_64/kernel/traps.c
--- a/arch/x86_64/kernel/traps.c
+++ b/arch/x86_64/kernel/traps.c
@@ -116,18 +116,18 @@ void printk_address(unsigned long addres
 {
  unsigned long offset = 0, symsize;
  const char *symname;
- char *modname;
```

```
+ char modname[MODULE_NAME_LEN+1];
  char *delim = ":";
- char namebuf[128];
+ char namebuf[KSYM_NAME_LEN+1];

  symname = kallsyms_lookup(address, &symsize, &offset,
-     &modname, namebuf);
+     modname, namebuf, 0);
  if (!symname) {
   printk(" [<%016lx>]\n", address);
   return;
  }
- if (!modname)
-  modname = delim = "";
+ if (!modname[0])
+  delim = "";
  printk(" [<%016lx>] %s%s%s%s+0x%lx/0x%lx\n",
   address, delim, modname, delim, symname, offset, symsize);
 }
diff --git a/fs/proc/base.c b/fs/proc/base.c
--- a/fs/proc/base.c
+++ b/fs/proc/base.c
@@ -275,14 +275,13 @@ static int proc_pid_auxv(struct task_str
  */
 static int proc_pid_wchan(struct task_struct *task, char *buffer)
 {
- char *modname;
  const char *sym_name;
  unsigned long wchan, size, offset;
  char namebuf[KSYM_NAME_LEN+1];

  wchan = get_wchan(task);

- sym_name = kallsyms_lookup(wchan, &size, &offset, &modname, namebuf);
+ sym_name = kallsyms_lookup(wchan, &size, &offset, NULL, namebuf, 1);
  if (sym_name)
   return sprintf(buffer, "%s", sym_name);
  return sprintf(buffer, "%lu", wchan);
diff --git a/include/linux/kallsyms.h b/include/linux/kallsyms.h
--- a/include/linux/kallsyms.h
+++ b/include/linux/kallsyms.h
@@ -20,7 +20,8 @@ extern int kallsyms_lookup_size_offset(u
 const char *kallsyms_lookup(unsigned long addr,
        unsigned long *symbolsize,
        unsigned long *offset,
-       char **modname, char *namebuf);
+       char *modname, char *namebuf,
+       int safe_to_lock);
```

```
 /* Replace "%s" in format with address, if found */
 extern void __print_symbol(const char *fmt, unsigned long address);
@@ -42,7 +43,8 @@ static inline int kallsyms_lookup_size_o
 static inline const char *kallsyms_lookup(unsigned long addr,
         unsigned long *symbolsize,
         unsigned long *offset,
-        char **modname, char *namebuf)
+        char *modname, char *namebuf,
+        int safe_to_lock)
 {
  return NULL;
 }
diff --git a/include/linux/module.h b/include/linux/module.h
--- a/include/linux/module.h
+++ b/include/linux/module.h
@@ -370,10 +370,10 @@ struct module *module_text_address(unsig
 struct module *__module_text_address(unsigned long addr);
 int is_module_address(unsigned long addr);

-/* Returns module and fills in value, defined and namebuf, or NULL if
-   symnum out of range. */
-struct module *module_get_kallsym(unsigned int symnum, unsigned long *value,
-    char *type, char *name, size_t namelen);
+/* fills in value, type, name and module_name. returns 0 on success
+   or -ENOENT if symnum out of range. */
+int module_get_kallsym(unsigned int symnum, unsigned long *value,
+         char *type, char *name, char *module_name);

 /* Look for this name: can be of form module:name. */
 unsigned long module_kallsyms_lookup_name(const char *name);
@@ -451,11 +451,15 @@ do {              \
  }            \
 } while(0)

-/* For kallsyms to ask for address resolution.  NULL means not found. */
-const char *module_address_lookup(unsigned long addr,
-      unsigned long *symbolsize,
-      unsigned long *offset,
-      char **modname);
+/* For kallsyms to ask for address resolution. -ENOENT means not found.
+   if modname is NULL it is not filled with the module name
+   if safe_to_lock is 0, the function will not take the module_mutex
+   which can be useful during oops backtrace resolution */
+int module_address_lookup(unsigned long addr,
+     unsigned long *size,
+     unsigned long *offset,
+     char *symname, char *modname,
```

```
+     int safe_to_lock);

 /* For extable.c to search modules' exception tables. */
 const struct exception_table_entry *search_module_extables(unsigned long addr);
@@ -518,21 +522,21 @@ static inline void module_put(struct mod

 #define __unsafe(mod)

-/* For kallsyms to ask for address resolution.  NULL means not found. */
-static inline const char *module_address_lookup(unsigned long addr,
-      unsigned long *symbolsize,
-      unsigned long *offset,
-      char **modname)
+/* For kallsyms to ask for address resolution. -ENOENT means not found. */
+static inline int module_address_lookup(unsigned long addr,
+      unsigned long *size,
+      unsigned long *offset,
+      char *symname, char *modname
+      int safe_to_lock)
 {
- return NULL;
+ return -ENOENT;
 }

-static inline struct module *module_get_kallsym(unsigned int symnum,
-      unsigned long *value,
-      char *type, char *name,
-      size_t namelen)
+static inline int module_get_kallsym(unsigned int symnum,
+         unsigned long *value, char *type,
+         char *name, char *modname)
 {
- return NULL;
+ return -ENOENT;
 }

 static inline unsigned long module_kallsyms_lookup_name(const char *name)
diff --git a/kernel/kallsyms.c b/kernel/kallsyms.c
--- a/kernel/kallsyms.c
+++ b/kernel/kallsyms.c
@@ -229,7 +229,7 @@ int kallsyms_lookup_size_offset(unsigned
  if (is_ksym_addr(addr))
   return !!get_symbol_pos(addr, symbolsize, offset);

- return !!module_address_lookup(addr, symbolsize, offset, NULL);
+ return !module_address_lookup(addr, symbolsize, offset, NULL, NULL, 1);
 }
```

```
 /*
@@ -242,10 +242,9 @@ int kallsyms_lookup_size_offset(unsigned
 const char *kallsyms_lookup(unsigned long addr,
        unsigned long *symbolsize,
        unsigned long *offset,
-       char **modname, char *namebuf)
+       char *modname, char *namebuf,
+       int safe_to_lock)
 {
- const char *msym;
-
  namebuf[KSYM_NAME_LEN] = 0;
  namebuf[0] = 0;

@@ -255,36 +254,36 @@ const char *kallsyms_lookup(unsigned lon
   pos = get_symbol_pos(addr, symbolsize, offset);
   /* Grab name */
   kallsyms_expand_symbol(get_symbol_offset(pos), namebuf);
-  *modname = NULL;
+  if (modname)
+   *modname = '\0';
   return namebuf;
  }

  /* see if it's in a module */
- msym = module_address_lookup(addr, symbolsize, offset, modname);
- if (msym)
-  return strncpy(namebuf, msym, KSYM_NAME_LEN);
+ if (module_address_lookup(addr, symbolsize, offset, namebuf, modname, safe_to_lock) < 0)
+  return NULL;

- return NULL;
+ return namebuf;
 }

 /* Replace "%s" in format with address, or returns -errno. */
 void __print_symbol(const char *fmt, unsigned long address)
 {
- char *modname;
  const char *name;
  unsigned long offset, size;
  char namebuf[KSYM_NAME_LEN+1];
+ char modbuf[MODULE_NAME_LEN+1];
  char buffer[sizeof("%s+%#lx/%#lx [%s]") + KSYM_NAME_LEN +
      2*(BITS_PER_LONG*3/10) + MODULE_NAME_LEN + 1];

- name = kallsyms_lookup(address, &size, &offset, &modname, namebuf);
+ name = kallsyms_lookup(address, &size, &offset, modbuf, namebuf, 0);
```

```
  if (!name)
   sprintf(buffer, "0x%lx", address);
  else {
-  if (modname)
+  if (modbuf[0])
    sprintf(buffer, "%s+%#lx/%#lx [%s]", name, offset,
-    size, modname);
+    size, modbuf);
   else
    sprintf(buffer, "%s+%#lx/%#lx", name, offset, size);
  }
@@ -295,24 +294,24 @@ void __print_symbol(const char *fmt, uns
 struct kallsym_iter
 {
  loff_t pos;
- struct module *owner;
  unsigned long value;
  unsigned int nameoff; /* If iterating in core kernel symbols */
  char type;
  char name[KSYM_NAME_LEN+1];
+ char module_name[MODULE_NAME_LEN+1];
 };

 static int get_ksymbol_mod(struct kallsym_iter *iter)
 {
- iter->owner = module_get_kallsym(iter->pos - kallsyms_num_syms,
-      &iter->value, &iter->type,
-      iter->name, sizeof(iter->name));
- if (iter->owner == NULL)
-  return 0;
+ int err;

- /* Label it "global" if it is exported, "local" if not exported. */
- iter->type = is_exported(iter->name, iter->owner)
-  ? toupper(iter->type) : tolower(iter->type);
+ err = module_get_kallsym(iter->pos - kallsyms_num_syms,
+     &iter->value, &iter->type,
+     iter->name, iter->module_name);
+ if (err <= 0) {
+  iter->module_name[0] = '\0';
+  return 0;
+ }

  return 1;
 }
@@ -322,7 +321,7 @@ static unsigned long get_ksymbol_core(st
 {
```

```
  unsigned off = iter->nameoff;

- iter->owner = NULL;
+ iter->module_name[0] = '\0';
  iter->value = kallsyms_addresses[iter->pos];

  iter->type = kallsyms_get_symbol_type(off);
@@ -347,7 +346,7 @@ static int update_iter(struct kallsym_it
   iter->pos = pos;
   return get_ksymbol_mod(iter);
  }
-
+
  /* If we're not on the desired position, reset to new position. */
  if (pos != iter->pos)
   reset_iter(iter, pos);
@@ -382,15 +381,15 @@ static int s_show(struct seq_file *m, vo
 {
  struct kallsym_iter *iter = m->private;

- /* Some debugging symbols have no name.  Ignore them. */
+ /* Some debugging symbols have no name.  Ignore them. */
  if (!iter->name[0])
   return 0;

- if (iter->owner)
+ if (iter->module_name[0])
   seq_printf(m, "%0*lx %c %s\t[%s]\n",
       (int)(2*sizeof(void*)),
       iter->value, iter->type, iter->name,
-      module_name(iter->owner));
+      iter->module_name);
  else
   seq_printf(m, "%0*lx %c %s\n",
       (int)(2*sizeof(void*)),
diff --git a/kernel/kprobes.c b/kernel/kprobes.c
--- a/kernel/kprobes.c
+++ b/kernel/kprobes.c
@@ -837,7 +837,7 @@ static void __kprobes report_probe(struc
   kprobe_type = "k";
  if (sym)
   seq_printf(pi, "%p  %s  %s+0x%x  %s\n", p->addr, kprobe_type,
-   sym, offset, (modname ? modname : " "));
+   sym, offset, modname);
  else
   seq_printf(pi, "%p  %s  %p\n", p->addr, kprobe_type, p->addr);
 }
@@ -868,13 +868,13 @@ static int __kprobes show_kprobe_addr(st
```

```
  const char *sym = NULL;
  unsigned int i = *(loff_t *) v;
  unsigned long size, offset = 0;
- char *modname, namebuf[128];
+ char modname[MODULE_NAME_LEN+1], namebuf[KSYM_NAME_LEN+1];

  head = &kprobe_table[i];
  preempt_disable();
  hlist_for_each_entry_rcu(p, node, head, hlist) {
   sym = kallsyms_lookup((unsigned long)p->addr, &size,
-     &offset, &modname, namebuf);
+     &offset, modname, namebuf, 1);
   if (p->pre_handler == aggr_pre_handler) {
    list_for_each_entry_rcu(kp, &p->list, list)
     report_probe(pi, kp, sym, offset, modname);
diff --git a/kernel/lockdep.c b/kernel/lockdep.c
--- a/kernel/lockdep.c
+++ b/kernel/lockdep.c
@@ -342,9 +342,8 @@ static const char *usage_str[] =
 const char * __get_key_name(struct lockdep_subclass_key *key, char *str)
 {
  unsigned long offs, size;
- char *modname;

- return kallsyms_lookup((unsigned long)key, &size, &offs, &modname, str);
+ return kallsyms_lookup((unsigned long)key, &size, &offs, NULL, str, 0);
 }

 void
diff --git a/kernel/module.c b/kernel/module.c
--- a/kernel/module.c
+++ b/kernel/module.c
@@ -44,6 +44,8 @@
 #include <asm/semaphore.h>
 #include <asm/cacheflush.h>
 #include <linux/license.h>
+#include <linux/kallsyms.h>
+#include <linux/ctype.h>

 #if 0
 #define DEBUGP printk
@@ -2101,26 +2103,42 @@ static const char *get_ksymbol(struct mo
 /* For kallsyms to ask for address resolution.  NULL means not found.
    We don't lock, as this is used for oops resolution and races are a
    lesser concern. */
-const char *module_address_lookup(unsigned long addr,
+int module_address_lookup(unsigned long addr,
       unsigned long *size,
```

```
       unsigned long *offset,
-      char **modname)
+      char *symname, char *modname,
+      int safe_to_lock)
 {
  struct module *mod;
+ const char *modsym;
+ int ret = -ENOENT;

+ if (safe_to_lock)
+  mutex_lock(&module_mutex);
+
  list_for_each_entry(mod, &modules, list) {
   if (within(addr, mod->module_init, mod->init_size)
       || within(addr, mod->module_core, mod->core_size)) {
    if (modname)
-    *modname = mod->name;
-   return get_ksymbol(mod, addr, size, offset);
+    strlcpy(modname, mod->name, MODULE_NAME_LEN);
+   modsym = get_ksymbol(mod, addr, size, offset);
+   if (modsym) {
+    ret = 0;
+    if (symname)
+     strlcpy(symname, modsym, KSYM_NAME_LEN);
+   }
+   break;
   }
  }
- return NULL;
+
+ if (safe_to_lock)
+  mutex_unlock(&module_mutex);
+
+ return ret;
 }

-struct module *module_get_kallsym(unsigned int symnum, unsigned long *value,
-    char *type, char *name, size_t namelen)
+int module_get_kallsym(unsigned int symnum, unsigned long *value,
+   char *type, char *name, char *module_name)
 {
  struct module *mod;

@@ -2128,16 +2146,20 @@ struct module *module_get_kallsym(unsign
  list_for_each_entry(mod, &modules, list) {
   if (symnum < mod->num_symtab) {
    *value = mod->symtab[symnum].st_value;
-   *type = mod->symtab[symnum].st_info;
```

```
    strlcpy(name, mod->strtab + mod->symtab[symnum].st_name,
-    namelen);
+    KSYM_NAME_LEN);
+   strlcpy(module_name, mod->name, MODULE_NAME_LEN);
+   *type = mod->symtab[symnum].st_info;
+   /* Label it "global" if it is exported, "local" if not exported. */
+   *type = is_exported(name, mod) ? toupper(*type) : tolower(*type);
+
    mutex_unlock(&module_mutex);
-   return mod;
+   return 0;
   }
   symnum -= mod->num_symtab;
  }
  mutex_unlock(&module_mutex);
- return NULL;
+ return -ENOENT;
 }

 static unsigned long mod_find_symname(struct module *mod, const char *name)
diff --git a/kernel/time/timer_list.c b/kernel/time/timer_list.c
--- a/kernel/time/timer_list.c
+++ b/kernel/time/timer_list.c
@@ -42,9 +42,8 @@ static void print_name_offset(struct seq
  char namebuf[KSYM_NAME_LEN+1];
  unsigned long size, offset;
  const char *sym_name;
- char *modname;

- sym_name = kallsyms_lookup(addr, &size, &offset, &modname, namebuf);
+ sym_name = kallsyms_lookup(addr, &size, &offset, NULL, namebuf, 1);
  if (sym_name)
   SEQ_printf(m, "%s", sym_name);
  else
diff --git a/kernel/time/timer_stats.c b/kernel/time/timer_stats.c
--- a/kernel/time/timer_stats.c
+++ b/kernel/time/timer_stats.c
@@ -260,9 +260,8 @@ static void print_name_offset(struct seq
  char namebuf[KSYM_NAME_LEN+1];
  unsigned long size, offset;
  const char *sym_name;
- char *modname;

- sym_name = kallsyms_lookup(addr, &size, &offset, &modname, namebuf);
+ sym_name = kallsyms_lookup(addr, &size, &offset, NULL, namebuf, 1);
  if (sym_name)
   seq_printf(m, "%s", sym_name);
  else
```

```
diff --git a/mm/slab.c b/mm/slab.c
--- a/mm/slab.c
+++ b/mm/slab.c
@@ -4380,16 +4380,16 @@ static void handle_slab(unsigned long *n
 static void show_symbol(struct seq_file *m, unsigned long address)
 {
 #ifdef CONFIG_KALLSYMS
- char *modname;
  const char *name;
  unsigned long offset, size;
  char namebuf[KSYM_NAME_LEN+1];
+ char modname[MODULE_NAME_LEN+1];

- name = kallsyms_lookup(address, &size, &offset, &modname, namebuf);
+ name = kallsyms_lookup(address, &size, &offset, modname, namebuf, 1);

  if (name) {
   seq_printf(m, "%s+%#lx/%#lx", name, offset, size);
-  if (modname)
+  if (modname[0])
    seq_printf(m, " [%s]", modname);
   return;
  }
```