Subject: [PATCH v3] Race between cat /proc/kallsyms and rmmod
Posted by Alexey Dobriyan on Mon, 19 Mar 2007 14:25:43 GMT
View Forum Message <> Reply to Message

Iterating code of /proc/kallsyms calls module_get_kallsym() which grabs
and drops module_mutex internally and returns "struct module *",
module is removed, aforementioned "struct module *" is used in non-trivial
way.

Steps to reproduce:

```
 modprobe/rmmod loop
 cat /proc/kallsyms >/dev/null loop
```

Copy all needed info under module_mutex.

NOTE: this patch keeps module_mutex static.

Signed-off-by: Alexey Dobriyan <adobriyan@sw.ru>

```
 include/linux/module.h |   13 +++++++++----
 kernel/kallsyms.c      |   32 ++++++++++++++++++++++++++----------
 kernel/module.c        |   12 ++++++++----
 3 files changed, 34 insertions(+), 23 deletions(-)

--- a/include/linux/module.h
+++ b/include/linux/module.h
@@ -370,10 +370,12 @@ 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
+/* Returns 0 and fills in value, defined and namebuf, or -ERANGE if
    symnum out of range. */
-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, size_t namelen,
+   char *module_name, size_t module_namelen,
+   int *exported);

 /* Look for this name: can be of form module:name. */
 unsigned long module_kallsyms_lookup_name(const char *name);
@@ -530,7 +532,10 @@ static inline const char *module_address
 static inline struct module *module_get_kallsym(unsigned int symnum,
       unsigned long *value,
       char *type, char *name,
```

```
-       size_t namelen)
+       size_t namelen,
+       char *module_name,
+       size_t module_namelen,
+       int *exported)
 {
  return NULL;
 }
--- a/kernel/kallsyms.c
+++ b/kernel/kallsyms.c
@@ -295,25 +295,22 @@ 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];
+ int exported;
 };

 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)
+ if (module_get_kallsym(iter->pos - kallsyms_num_syms,
+       &iter->value, &iter->type,
+       iter->name, sizeof(iter->name),
+       iter->module_name, sizeof(iter->module_name),
+       &iter->exported) < 0)
   return 0;
-
- /* 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);
-
  return 1;
 }

@@ -322,7 +319,7 @@ 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);
@@ -386,12 +383,17 @@ static int s_show(struct seq_file *m, vo
  if (!iter->name[0])
   return 0;

- if (iter->owner)
+ if (iter->module_name[0]) {
+  char type;
+
+  /* Label it "global" if it is exported,
+   * "local" if not exported. */
+  type = iter->exported ? toupper(iter->type) :
+     tolower(iter->type);
   seq_printf(m, "%0*lx %c %s\t[%s]\n",
       (int)(2*sizeof(void*)),
-      iter->value, iter->type, iter->name,
-      module_name(iter->owner));
- else
+      iter->value, type, iter->name, iter->module_name);
+ } else
   seq_printf(m, "%0*lx %c %s\n",
       (int)(2*sizeof(void*)),
       iter->value, iter->type, iter->name);
--- a/kernel/module.c
+++ b/kernel/module.c
@@ -2119,8 +2119,10 @@ const char *module_address_lookup(unsign
  return NULL;
 }

-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, size_t namelen,
+   char *module_name, size_t module_namelen,
+   int *exported)
 {
  struct module *mod;

@@ -2131,13 +2133,15 @@ struct module *module_get_kallsym(unsign
    *type = mod->symtab[symnum].st_info;
    strlcpy(name, mod->strtab + mod->symtab[symnum].st_name,
     namelen);
+   strlcpy(module_name, mod->name, module_namelen);
+   *exported = is_exported(name, mod);
    mutex_unlock(&module_mutex);
```

```
-   return mod;
+   return 0;
    }
    symnum -= mod->num_symtab;
   }
   mutex_unlock(&module_mutex);
- return NULL;
+ return -ERANGE;
 }

 static unsigned long mod_find_symname(struct module *mod, const char *name)
```