Subject: [PATCH] iptables compat code error way & module refcounting fix
Posted by Mishin Dmitry on Mon, 30 Oct 2006 10:31:32 GMT
View Forum Message <> Reply to Message

This patch fixes bug in iptables modules refcounting on compat error way.

As we are getting modules in check_compat_entry_size_and_hooks(), in case of
later error, we should put them all in translate_compat_table(), not in the
compat_copy_entry_from_user() or compat_copy_match_from_user(), as it is now.

Signed-off-by: Dmitry Mishin <dim@openvz.org>
Acked-by: Vasily Averin <vvs@openvz.org>
Acked-by: Kirill Korotaev <dev@openvz.org>

```
 ip_tables.c |   36 ++++++++++++-------------------------
 1 file changed, 11 insertions(+), 25 deletions(-)

---
diff --git a/net/ipv4/netfilter/ip_tables.c b/net/ipv4/netfilter/ip_tables.c
index 4b90927..128ba23 100644
--- a/net/ipv4/netfilter/ip_tables.c
+++ b/net/ipv4/netfilter/ip_tables.c
@@ -1513,7 +1513,7 @@ cleanup_matches:

 static inline int compat_copy_match_from_user(struct ipt_entry_match *m,
   void **dstptr, compat_uint_t *size, const char *name,
- const struct ipt_ip *ip, unsigned int hookmask, int *i)
+ const struct ipt_ip *ip, unsigned int hookmask)
 {
   struct ipt_entry_match *dm;
   struct ipt_match *match;
@@ -1526,22 +1526,13 @@ static inline int compat_copy_match_from
   ret = xt_check_match(match, AF_INET, dm->u.match_size - sizeof(*dm),
         name, hookmask, ip->proto,
         ip->invflags & IPT_INV_PROTO);
- if (ret)
-  goto err;
-
- if (m->u.kernel.match->checkentry
+ if (!ret && m->u.kernel.match->checkentry
      && !m->u.kernel.match->checkentry(name, ip, match, dm->data,
           hookmask)) {
    duprintf("ip_tables: check failed for `%s'.\n",
      m->u.kernel.match->name);
    ret = -EINVAL;
-  goto err;
   }
- (*i)++;
```

```
- return 0;
-
-err:
- module_put(m->u.kernel.match->me);
  return ret;
 }

@@ -1553,19 +1544,18 @@ static int compat_copy_entry_from_user(s
  struct ipt_target *target;
  struct ipt_entry *de;
  unsigned int origsize;
- int ret, h, j;
+ int ret, h;

  ret = 0;
  origsize = *size;
  de = (struct ipt_entry *)*dstptr;
  memcpy(de, e, sizeof(struct ipt_entry));

- j = 0;
  *dstptr += sizeof(struct compat_ipt_entry);
  ret = IPT_MATCH_ITERATE(e, compat_copy_match_from_user, dstptr, size,
-   name, &de->ip, de->comefrom, &j);
+   name, &de->ip, de->comefrom);
  if (ret)
-  goto cleanup_matches;
+  goto err;
  de->target_offset = e->target_offset - (origsize - *size);
  t = ipt_get_target(e);
  target = t->u.kernel.target;
@@ -1599,12 +1589,7 @@ static int compat_copy_entry_from_user(s
   goto err;
  }
  ret = 0;
- return ret;
-
 err:
- module_put(t->u.kernel.target->me);
-cleanup_matches:
- IPT_MATCH_ITERATE(e, cleanup_match, &j);
  return ret;
 }

@@ -1618,7 +1603,7 @@ translate_compat_table(const char *name,
   unsigned int *hook_entries,
   unsigned int *underflows)
 {
- unsigned int i;
```

```
+ unsigned int i, j;
  struct xt_table_info *newinfo, *info;
  void *pos, *entry0, *entry1;
  unsigned int size;
@@ -1636,21 +1621,21 @@ translate_compat_table(const char *name,
  }

  duprintf("translate_compat_table: size %u\n", info->size);
- i = 0;
+ j = 0;
  xt_compat_lock(AF_INET);
  /* Walk through entries, checking offsets. */
  ret = IPT_ENTRY_ITERATE(entry0, total_size,
      check_compat_entry_size_and_hooks,
      info, &size, entry0,
      entry0 + total_size,
-     hook_entries, underflows, &i, name);
+     hook_entries, underflows, &j, name);
  if (ret != 0)
   goto out_unlock;

  ret = -EINVAL;
- if (i != number) {
+ if (j != number) {
   duprintf("translate_compat_table: %u not %u entries\n",
-     i, number);
+     j, number);
   goto out_unlock;
  }

@@ -1709,6 +1694,7 @@ translate_compat_table(const char *name,
 free_newinfo:
  xt_free_table_info(newinfo);
 out:
+ IPT_ENTRY_ITERATE(entry0, total_size, cleanup_entry, &j);
  return ret;
 out_unlock:
  xt_compat_unlock(AF_INET);
```