Subject: [PATCH nf-2.6.22] [netfilter] early_drop imrovement
Posted by vaverin on Sat, 07 Apr 2007 11:45:35 GMT
View Forum Message <> Reply to Message

When the number of conntracks is reached nf_conntrack_max limit, early_drop() is
called and tries to free one of already used conntracks in one of the hash
buckets. If it does not find any conntracks that may be freed, it
leads to transmission errors.
However it is not fair because of current hash bucket may be empty but the
neighbour ones can have the number of conntracks that can be freed. On the other
hand the number of checked conntracks is not limited and it can cause a long delay.
The following patch limits the number of checked conntracks by average number of
conntracks in one hash bucket and allows to search conntracks in other hash buckets.

Signed-off-by: Vasily Averin <vvs@sw.ru>

```
diff --git a/net/netfilter/nf_conntrack_core.c b/net/netfilter/nf_conntrack_core.c
index e132c8a..d0b5794 100644
--- a/net/netfilter/nf_conntrack_core.c
+++ b/net/netfilter/nf_conntrack_core.c
@@ -525,7 +525,7 @@ EXPORT_SYMBOL_GPL(nf_conntrack_tuple_taken);

 /* There's a small race here where we may free a just-assured
    connection.  Too bad: we're in trouble anyway. */
-static int early_drop(struct list_head *chain)
+static int __early_drop(struct list_head *chain, unsigned int *cnt)
 {
  /* Traverse backwards: gives us oldest, which is roughly LRU */
  struct nf_conntrack_tuple_hash *h;
@@ -540,6 +540,10 @@ static int early_drop(struct list_head *chain)
    atomic_inc(&ct->ct_general.use);
    break;
   }
+  if (!--(*cnt)) {
+   dropped = 1;
+   break;
+  }
  }
  read_unlock_bh(&nf_conntrack_lock);

@@ -555,6 +559,21 @@ static int early_drop(struct list_head *chain)
  return dropped;
 }

+static int early_drop(const struct nf_conntrack_tuple *orig)
+{
+ unsigned int i, hash, cnt;
+ int ret = 0;
```

```
+
+ hash = hash_conntrack(orig);
+ cnt = (nf_conntrack_max/nf_conntrack_htable_size) + 1;
+
+ for (i = 0;
+   !ret && i < nf_conntrack_htable_size;
+   ++i, hash = ++hash % nf_conntrack_htable_size)
+  ret = __early_drop(&nf_conntrack_hash[hash], &cnt);
+ return ret;
+}
+
 static struct nf_conn *
 __nf_conntrack_alloc(const struct nf_conntrack_tuple *orig,
       const struct nf_conntrack_tuple *repl,
@@ -574,9 +593,7 @@ __nf_conntrack_alloc(const struct nf_conntrack_tuple *orig,

  if (nf_conntrack_max
     && atomic_read(&nf_conntrack_count) > nf_conntrack_max) {
-  unsigned int hash = hash_conntrack(orig);
-  /* Try dropping from this hash chain. */
-  if (!early_drop(&nf_conntrack_hash[hash])) {
+  if (!early_drop(orig)) {
    atomic_dec(&nf_conntrack_count);
    if (net_ratelimit())
     printk(KERN_WARNING
```