Subject: [PATCH net-2.6.25 6/10][NETNS][FRAGS]: Make the net.ipv4.ipfrag_timeout work in namespaces.
Posted by Pavel Emelianov on Tue, 22 Jan 2008 14:02:17 GMT
View Forum Message <> Reply to Message

Move it to the netns_frags, adjust the usage and
make the appropriate ctl table writable.

Now fragment, that live in different namespaces can
live for different times.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

---
 include/net/inet_frag.h | 4 +++-
 net/ipv4/inet_fragment.c | 2 +-
 net/ipv4/ip_fragment.c | 20 ++++++++++----------
 net/ipv6/netfilter/nf_conntrack_reasm.c | 4 ++--
 net/ipv6/reassembly.c | 6 +++---
 5 files changed, 19 insertions(+), 17 deletions(-)

diff --git a/include/net/inet_frag.h b/include/net/inet_frag.h
index 6edce7b..f56e296 100644
--- a/include/net/inet_frag.h
+++ b/include/net/inet_frag.h
@@ -4,6 +4,9 @@
 struct netns_frags {
  int nqueues;
  atomic_t mem;
+
+ /* sysctls */
+ int timeout;
 };

 struct inet_frag_queue {
@@ -29,7 +32,6 @@ struct inet_frag_queue {
 struct inet_frags_ctl {
  int high_thresh;
  int low_thresh;
- int timeout;
  int secret_interval;
 };

diff --git a/net/ipv4/inet_fragment.c b/net/ipv4/inet_fragment.c
index ad79ae0..9da9679 100644
--- a/net/ipv4/inet_fragment.c
+++ b/net/ipv4/inet_fragment.c
@@ -206,7 +206,7 @@ static struct inet_frag_queue *inet_frag_intern(struct netns_frags *nf,

```
  }
 #endif
  qp = qp_in;
- if (!mod_timer(&qp->timer, jiffies + f->ctl->timeout))
+ if (!mod_timer(&qp->timer, jiffies + nf->timeout))
    atomic_inc(&qp->refcnt);

  atomic_inc(&qp->refcnt);
diff --git a/net/ipv4/ip_fragment.c b/net/ipv4/ip_fragment.c
index c51e1a1..70d241c 100644
--- a/net/ipv4/ip_fragment.c
+++ b/net/ipv4/ip_fragment.c
@@ -83,13 +83,6 @@ static struct inet_frags_ctl ip4_frags_ctl __read_mostly = {
    */
  .high_thresh  = 256 * 1024,
  .low_thresh  = 192 * 1024,
-
- /*
-  * Important NOTE! Fragment queue must be destroyed before MSL expires.
-  * RFC791 is wrong proposing to prolongate timer each fragment arrival
-  * by TTL.
-  */
- .timeout  = IP_FRAG_TIME,
  .secret_interval = 10 * 60 * HZ,
 };

@@ -287,7 +280,7 @@ static int ip_frag_reinit(struct ipq *qp)
 {
  struct sk_buff *fp;

- if (!mod_timer(&qp->q.timer, jiffies + ip4_frags_ctl.timeout)) {
+ if (!mod_timer(&qp->q.timer, jiffies + qp->q.net->timeout)) {
    atomic_inc(&qp->q.refcnt);
    return -ETIMEDOUT;
  }
@@ -633,7 +626,7 @@ static struct ctl_table ip4_frags_ctl_table[] = {
  {
    .ctl_name = NET_IPV4_IPFRAG_TIME,
    .procname = "ipfrag_time",
-   .data  = &ip4_frags_ctl.timeout,
+   .data  = &init_net.ipv4.frags.timeout,
    .maxlen  = sizeof(int),
    .mode  = 0644,
    .proc_handler = &proc_dointvec_jiffies,
@@ -672,7 +665,7 @@ static int ip4_frags_ctl_register(struct net *net)

    table[0].mode &= ~0222;
    table[1].mode &= ~0222;
```

```
-  table[2].mode &= ~0222;
+  table[2].data = &net->ipv4.frags.timeout;
   table[3].mode &= ~0222;
   table[4].mode &= ~0222;
  }
@@ -712,6 +705,13 @@ static inline void ip4_frags_ctl_unregister(struct net *net)

 static int ipv4_frags_init_net(struct net *net)
 {
+ /*
+  * Important NOTE! Fragment queue must be destroyed before MSL expires.
+  * RFC791 is wrong proposing to prolongate timer each fragment arrival
+  * by TTL.
+  */
+ net->ipv4.frags.timeout = IP_FRAG_TIME;
+
  inet_frags_init_net(&net->ipv4.frags);

  return ip4_frags_ctl_register(net);
diff --git a/net/ipv6/netfilter/nf_conntrack_reasm.c b/net/ipv6/netfilter/nf_conntrack_reasm.c
index cb826be..92a311f 100644
--- a/net/ipv6/netfilter/nf_conntrack_reasm.c
+++ b/net/ipv6/netfilter/nf_conntrack_reasm.c
@@ -73,7 +73,6 @@ struct nf_ct_frag6_queue
 static struct inet_frags_ctl nf_frags_ctl __read_mostly = {
  .high_thresh  = 256 * 1024,
  .low_thresh  = 192 * 1024,
- .timeout  = IPV6_FRAG_TIMEOUT,
  .secret_interval = 10 * 60 * HZ,
 };

@@ -84,7 +83,7 @@ static struct netns_frags nf_init_frags;
 struct ctl_table nf_ct_ipv6_sysctl_table[] = {
  {
   .procname = "nf_conntrack_frag6_timeout",
-  .data  = &nf_frags_ctl.timeout,
+  .data  = &nf_init_frags.timeout,
   .maxlen  = sizeof(unsigned int),
   .mode  = 0644,
   .proc_handler = &proc_dointvec_jiffies,
@@ -712,6 +711,7 @@ int nf_ct_frag6_init(void)
  nf_frags.qsize = sizeof(struct nf_ct_frag6_queue);
  nf_frags.match = ip6_frag_match;
  nf_frags.frag_expire = nf_ct_frag6_expire;
+ nf_init_frags.timeout = IPV6_FRAG_TIMEOUT;
  inet_frags_init_net(&nf_init_frags);
  inet_frags_init(&nf_frags);
```

```
diff --git a/net/ipv6/reassembly.c b/net/ipv6/reassembly.c
index 0300dcb..9176136 100644
--- a/net/ipv6/reassembly.c
+++ b/net/ipv6/reassembly.c
@@ -650,7 +650,7 @@ static struct ctl_table ip6_frags_ctl_table[] = {
  {
   .ctl_name = NET_IPV6_IP6FRAG_TIME,
   .procname = "ip6frag_time",
-  .data  = &init_net.ipv6.sysctl.frags.timeout,
+  .data  = &init_net.ipv6.frags.timeout,
   .maxlen  = sizeof(int),
   .mode  = 0644,
   .proc_handler = &proc_dointvec_jiffies,
@@ -681,7 +681,7 @@ static int ip6_frags_sysctl_register(struct net *net)

   table[0].mode &= ~0222;
   table[1].mode &= ~0222;
-  table[2].mode &= ~0222;
+  table[2].data = &net->ipv6.frags.timeout;
   table[3].mode &= ~0222;
  }

@@ -724,7 +724,7 @@ static int ipv6_frags_init_net(struct net *net)

  net->ipv6.sysctl.frags.high_thresh = 256 * 1024;
  net->ipv6.sysctl.frags.low_thresh = 192 * 1024;
- net->ipv6.sysctl.frags.timeout = IPV6_FRAG_TIMEOUT;
+ net->ipv6.frags.timeout = IPV6_FRAG_TIMEOUT;
  net->ipv6.sysctl.frags.secret_interval = 10 * 60 * HZ;

  inet_frags_init_net(&net->ipv6.frags);
--
1.5.3.4
```