Subject: [PATCH 3/9] Collect common sysctl variables together
Posted by Pavel Emelianov on Fri, 12 Oct 2007 13:10:05 GMT
View Forum Message <> Reply to Message

Some sysctl variables are used to tune the frag queues
management and it will be useful to work with them in
a common way in the future, so move them into one
structure, moreover they are the same for all the frag
management codes.

I don't place them in the existing inet_frags object,
introduced in the previous patch for two reasons:

 1. to keep them in the __read_mostly section;
 2. not to export the whole inet_frags objects outside.

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

```
diff --git a/include/net/inet_frag.h b/include/net/inet_frag.h
index d51f238..ada03ba 100644
--- a/include/net/inet_frag.h
+++ b/include/net/inet_frag.h
@@ -20,6 +20,13 @@ struct inet_frag_queue {

 #define INETFRAGS_HASHSZ  64

+struct inet_frags_ctl {
+ int high_thresh;
+ int low_thresh;
+ int timeout;
+ int secret_interval;
+};
+
 struct inet_frags {
  struct list_head lru_list;
  struct hlist_head hash[INETFRAGS_HASHSZ];
@@ -28,6 +35,7 @@ struct inet_frags {
  int   nqueues;
  atomic_t  mem;
  struct timer_list secret_timer;
+ struct inet_frags_ctl *ctl;
 };

 void inet_frags_init(struct inet_frags *);
diff --git a/include/net/ip.h b/include/net/ip.h
index a18dcec..e7b0feb 100644
```

```
--- a/include/net/ip.h
+++ b/include/net/ip.h
@@ -177,10 +177,8 @@ extern int sysctl_ip_default_ttl;
 extern int sysctl_ip_nonlocal_bind;

 /* From ip_fragment.c */
-extern int sysctl_ipfrag_high_thresh;
-extern int sysctl_ipfrag_low_thresh;
-extern int sysctl_ipfrag_time;
-extern int sysctl_ipfrag_secret_interval;
+struct inet_frags_ctl;
+extern struct inet_frags_ctl ip4_frags_ctl;
 extern int sysctl_ipfrag_max_dist;

 /* From inetpeer.c */
diff --git a/include/net/ipv6.h b/include/net/ipv6.h
index 77cdab3..b29d76c 100644
--- a/include/net/ipv6.h
+++ b/include/net/ipv6.h
@@ -565,10 +565,8 @@ extern int inet6_hash_connect(struct inet_timewait_death_row
*death_row,
 /*
  * reassembly.c
  */
-extern int sysctl_ip6frag_high_thresh;
-extern int sysctl_ip6frag_low_thresh;
-extern int sysctl_ip6frag_time;
-extern int sysctl_ip6frag_secret_interval;
+struct inet_frags_ctl;
+extern struct inet_frags_ctl ip6_frags_ctl;

 extern const struct proto_ops inet6_stream_ops;
 extern const struct proto_ops inet6_dgram_ops;
diff --git a/include/net/netfilter/ipv6/nf_conntrack_ipv6.h
b/include/net/netfilter/ipv6/nf_conntrack_ipv6.h
index 070d12c..f703533 100644
--- a/include/net/netfilter/ipv6/nf_conntrack_ipv6.h
+++ b/include/net/netfilter/ipv6/nf_conntrack_ipv6.h
@@ -15,8 +15,7 @@ extern void nf_ct_frag6_output(unsigned int hooknum, struct sk_buff *skb,
          struct net_device *out,
          int (*okfn)(struct sk_buff *));

-extern unsigned int nf_ct_frag6_timeout;
-extern unsigned int nf_ct_frag6_low_thresh;
-extern unsigned int nf_ct_frag6_high_thresh;
+struct inet_frags_ctl;
+extern struct inet_frags_ctl nf_frags_ctl;
```

```
 #endif /* _NF_CONNTRACK_IPV6_H*/
diff --git a/net/ipv4/ip_fragment.c b/net/ipv4/ip_fragment.c
index 5e1667e..61035a8 100644
--- a/net/ipv4/ip_fragment.c
+++ b/net/ipv4/ip_fragment.c
@@ -50,21 +50,8 @@
  * as well. Or notify me, at least. --ANK
  */

-/* Fragment cache limits. We will commit 256K at one time. Should we
- * cross that limit we will prune down to 192K. This should cope with
- * even the most extreme cases without allowing an attacker to measurably
- * harm machine performance.
- */
-int sysctl_ipfrag_high_thresh __read_mostly = 256*1024;
-int sysctl_ipfrag_low_thresh __read_mostly = 192*1024;
-
 int sysctl_ipfrag_max_dist __read_mostly = 64;

-/* Important NOTE! Fragment queue must be destroyed before MSL expires.
- * RFC791 is wrong proposing to prolongate timer each fragment arrival by TTL.
- */
-int sysctl_ipfrag_time __read_mostly = IP_FRAG_TIME;
-
 struct ipfrag_skb_cb
 {
  struct inet_skb_parm h;
@@ -87,6 +74,25 @@ struct ipq {
  struct inet_peer *peer;
 };

+struct inet_frags_ctl ip4_frags_ctl __read_mostly = {
+ /*
+  * Fragment cache limits. We will commit 256K at one time. Should we
+  * cross that limit we will prune down to 192K. This should cope with
+  * even the most extreme cases without allowing an attacker to
+  * measurably harm machine performance.
+  */
+ .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,
```

```
+};
+
 static struct inet_frags ip4_frags;

 int ip_frag_nqueues(void)
@@ -120,8 +126,6 @@ static unsigned int ipqhashfn(__be16 id, __be32 saddr, __be32 daddr,
u8 prot)
        ip4_frags.rnd) & (INETFRAGS_HASHSZ - 1);
 }

-int sysctl_ipfrag_secret_interval __read_mostly = 10 * 60 * HZ;
-
 static void ipfrag_secret_rebuild(unsigned long dummy)
 {
  unsigned long now = jiffies;
@@ -147,7 +151,7 @@ static void ipfrag_secret_rebuild(unsigned long dummy)
  }
  write_unlock(&ip4_frags.lock);

- mod_timer(&ip4_frags.secret_timer, now + sysctl_ipfrag_secret_interval);
+ mod_timer(&ip4_frags.secret_timer, now + ip4_frags_ctl.secret_interval);
 }

 /* Memory Tracking Functions. */
@@ -234,7 +238,7 @@ static void ip_evictor(void)
  struct list_head *tmp;
  int work;

- work = atomic_read(&ip4_frags.mem) - sysctl_ipfrag_low_thresh;
+ work = atomic_read(&ip4_frags.mem) - ip4_frags_ctl.low_thresh;
  if (work <= 0)
   return;

@@ -323,7 +327,7 @@ static struct ipq *ip_frag_intern(struct ipq *qp_in)
 #endif
  qp = qp_in;

- if (!mod_timer(&qp->q.timer, jiffies + sysctl_ipfrag_time))
+ if (!mod_timer(&qp->q.timer, jiffies + ip4_frags_ctl.timeout))
   atomic_inc(&qp->q.refcnt);

  atomic_inc(&qp->q.refcnt);
@@ -429,7 +433,7 @@ static int ip_frag_reinit(struct ipq *qp)
 {
  struct sk_buff *fp;

- if (!mod_timer(&qp->q.timer, jiffies + sysctl_ipfrag_time)) {
+ if (!mod_timer(&qp->q.timer, jiffies + ip4_frags_ctl.timeout)) {
```

```
   atomic_inc(&qp->q.refcnt);
   return -ETIMEDOUT;
  }
@@ -693,7 +697,7 @@ struct sk_buff *ip_defrag(struct sk_buff *skb, u32 user)
  IP_INC_STATS_BH(IPSTATS_MIB_REASMREQDS);

  /* Start by cleaning up the memory. */
- if (atomic_read(&ip4_frags.mem) > sysctl_ipfrag_high_thresh)
+ if (atomic_read(&ip4_frags.mem) > ip4_frags_ctl.high_thresh)
   ip_evictor();

  dev = skb->dev;
@@ -724,9 +728,10 @@ void __init ipfrag_init(void)
 {
  init_timer(&ip4_frags.secret_timer);
  ip4_frags.secret_timer.function = ipfrag_secret_rebuild;
- ip4_frags.secret_timer.expires = jiffies + sysctl_ipfrag_secret_interval;
+ ip4_frags.secret_timer.expires = jiffies + ip4_frags_ctl.secret_interval;
  add_timer(&ip4_frags.secret_timer);

+ ip4_frags.ctl = &ip4_frags_ctl;
  inet_frags_init(&ip4_frags);
 }

diff --git a/net/ipv4/sysctl_net_ipv4.c b/net/ipv4/sysctl_net_ipv4.c
index eb286ab..c98ef16 100644
--- a/net/ipv4/sysctl_net_ipv4.c
+++ b/net/ipv4/sysctl_net_ipv4.c
@@ -19,6 +19,7 @@
 #include <net/route.h>
 #include <net/tcp.h>
 #include <net/cipso_ipv4.h>
+#include <net/inet_frag.h>

 /* From af_inet.c */
 extern int sysctl_ip_nonlocal_bind;
@@ -357,7 +358,7 @@ ctl_table ipv4_table[] = {
  {
   .ctl_name = NET_IPV4_IPFRAG_HIGH_THRESH,
   .procname = "ipfrag_high_thresh",
-  .data  = &sysctl_ipfrag_high_thresh,
+  .data  = &ip4_frags_ctl.high_thresh,
   .maxlen  = sizeof(int),
   .mode  = 0644,
   .proc_handler = &proc_dointvec
@@ -365,7 +366,7 @@ ctl_table ipv4_table[] = {
  {
   .ctl_name = NET_IPV4_IPFRAG_LOW_THRESH,
```

```
   .procname = "ipfrag_low_thresh",
-  .data  = &sysctl_ipfrag_low_thresh,
+  .data  = &ip4_frags_ctl.low_thresh,
   .maxlen  = sizeof(int),
   .mode  = 0644,
   .proc_handler = &proc_dointvec
@@ -381,7 +382,7 @@ ctl_table ipv4_table[] = {
  {
   .ctl_name = NET_IPV4_IPFRAG_TIME,
   .procname = "ipfrag_time",
-  .data  = &sysctl_ipfrag_time,
+  .data  = &ip4_frags_ctl.timeout,
   .maxlen  = sizeof(int),
   .mode  = 0644,
   .proc_handler = &proc_dointvec_jiffies,
@@ -732,7 +733,7 @@ ctl_table ipv4_table[] = {
  {
   .ctl_name = NET_IPV4_IPFRAG_SECRET_INTERVAL,
   .procname = "ipfrag_secret_interval",
-  .data  = &sysctl_ipfrag_secret_interval,
+  .data  = &ip4_frags_ctl.secret_interval,
   .maxlen  = sizeof(int),
   .mode  = 0644,
   .proc_handler = &proc_dointvec_jiffies,
diff --git a/net/ipv6/netfilter/nf_conntrack_l3proto_ipv6.c
b/net/ipv6/netfilter/nf_conntrack_l3proto_ipv6.c
index 37a3db9..572c0bc 100644
--- a/net/ipv6/netfilter/nf_conntrack_l3proto_ipv6.c
+++ b/net/ipv6/netfilter/nf_conntrack_l3proto_ipv6.c
@@ -18,6 +18,7 @@
 #include <linux/icmp.h>
 #include <linux/sysctl.h>
 #include <net/ipv6.h>
+#include <net/inet_frag.h>

 #include <linux/netfilter_ipv6.h>
 #include <net/netfilter/nf_conntrack.h>
@@ -307,7 +308,7 @@ static ctl_table nf_ct_ipv6_sysctl_table[] = {
  {
   .ctl_name = NET_NF_CONNTRACK_FRAG6_TIMEOUT,
   .procname = "nf_conntrack_frag6_timeout",
-  .data  = &nf_ct_frag6_timeout,
+  .data  = &nf_frags_ctl.timeout,
   .maxlen  = sizeof(unsigned int),
   .mode  = 0644,
   .proc_handler = &proc_dointvec_jiffies,
@@ -315,7 +316,7 @@ static ctl_table nf_ct_ipv6_sysctl_table[] = {
  {
```

```
   .ctl_name = NET_NF_CONNTRACK_FRAG6_LOW_THRESH,
   .procname = "nf_conntrack_frag6_low_thresh",
-  .data  = &nf_ct_frag6_low_thresh,
+  .data  = &nf_frags_ctl.low_thresh,
   .maxlen  = sizeof(unsigned int),
   .mode  = 0644,
   .proc_handler = &proc_dointvec,
@@ -323,7 +324,7 @@ static ctl_table nf_ct_ipv6_sysctl_table[] = {
  {
   .ctl_name = NET_NF_CONNTRACK_FRAG6_HIGH_THRESH,
   .procname = "nf_conntrack_frag6_high_thresh",
-  .data  = &nf_ct_frag6_high_thresh,
+  .data  = &nf_frags_ctl.high_thresh,
   .maxlen  = sizeof(unsigned int),
   .mode  = 0644,
   .proc_handler = &proc_dointvec,
diff --git a/net/ipv6/netfilter/nf_conntrack_reasm.c b/net/ipv6/netfilter/nf_conntrack_reasm.c
index eb2ca1b..966a888 100644
--- a/net/ipv6/netfilter/nf_conntrack_reasm.c
+++ b/net/ipv6/netfilter/nf_conntrack_reasm.c
@@ -49,10 +49,6 @@
 #define NF_CT_FRAG6_LOW_THRESH 196608  /* == 192*1024 */
 #define NF_CT_FRAG6_TIMEOUT IPV6_FRAG_TIMEOUT

-unsigned int nf_ct_frag6_high_thresh __read_mostly = 256*1024;
-unsigned int nf_ct_frag6_low_thresh __read_mostly = 192*1024;
-unsigned long nf_ct_frag6_timeout __read_mostly = IPV6_FRAG_TIMEOUT;
-
 struct nf_ct_frag6_skb_cb
 {
  struct inet6_skb_parm h;
@@ -74,6 +70,13 @@ struct nf_ct_frag6_queue
  __u16   nhoffset;
 };

+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,
+};
+
 static struct inet_frags nf_frags;

 static __inline__ void __fq_unlink(struct nf_ct_frag6_queue *fq)
@@ -117,8 +120,6 @@ static unsigned int ip6qhashfn(__be32 id, struct in6_addr *saddr,
  return c & (INETFRAGS_HASHSZ - 1);
 }
```

```
-int nf_ct_frag6_secret_interval = 10 * 60 * HZ;
-
 static void nf_ct_frag6_secret_rebuild(unsigned long dummy)
 {
  unsigned long now = jiffies;
@@ -144,7 +145,7 @@ static void nf_ct_frag6_secret_rebuild(unsigned long dummy)
  }
  write_unlock(&nf_frags.lock);

- mod_timer(&nf_frags.secret_timer, now + nf_ct_frag6_secret_interval);
+ mod_timer(&nf_frags.secret_timer, now + nf_frags_ctl.secret_interval);
 }

 /* Memory Tracking Functions. */
@@ -229,10 +230,10 @@ static void nf_ct_frag6_evictor(void)
  unsigned int work;

  work = atomic_read(&nf_frags.mem);
- if (work <= nf_ct_frag6_low_thresh)
+ if (work <= nf_frags_ctl.low_thresh)
   return;

- work -= nf_ct_frag6_low_thresh;
+ work -= nf_frags_ctl.low_thresh;
  while (work > 0) {
   read_lock(&nf_frags.lock);
   if (list_empty(&nf_frags.lru_list)) {
@@ -296,7 +297,7 @@ static struct nf_ct_frag6_queue *nf_ct_frag6_intern(unsigned int hash,
 #endif
  fq = fq_in;

- if (!mod_timer(&fq->q.timer, jiffies + nf_ct_frag6_timeout))
+ if (!mod_timer(&fq->q.timer, jiffies + nf_frags_ctl.timeout))
   atomic_inc(&fq->q.refcnt);

  atomic_inc(&fq->q.refcnt);
@@ -766,7 +767,7 @@ struct sk_buff *nf_ct_frag6_gather(struct sk_buff *skb)
   goto ret_orig;
  }

- if (atomic_read(&nf_frags.mem) > nf_ct_frag6_high_thresh)
+ if (atomic_read(&nf_frags.mem) > nf_frags_ctl.high_thresh)
   nf_ct_frag6_evictor();

  fq = fq_find(fhdr->identification, &hdr->saddr, &hdr->daddr);
@@ -838,10 +839,10 @@ int nf_ct_frag6_kfree_frags(struct sk_buff *skb)
 int nf_ct_frag6_init(void)
```

```
 {
  setup_timer(&nf_frags.secret_timer, nf_ct_frag6_secret_rebuild, 0);
- nf_frags.secret_timer.expires = jiffies
-         + nf_ct_frag6_secret_interval;
+ nf_frags.secret_timer.expires = jiffies + nf_frags_ctl.secret_interval;
  add_timer(&nf_frags.secret_timer);

+ nf_frags.ctl = &nf_frags_ctl;
  inet_frags_init(&nf_frags);

  return 0;
@@ -852,6 +853,6 @@ void nf_ct_frag6_cleanup(void)
  inet_frags_fini(&nf_frags);

  del_timer(&nf_frags.secret_timer);
- nf_ct_frag6_low_thresh = 0;
+ nf_frags_ctl.low_thresh = 0;
  nf_ct_frag6_evictor();
 }
diff --git a/net/ipv6/reassembly.c b/net/ipv6/reassembly.c
index 7b6315f..f0e22be 100644
--- a/net/ipv6/reassembly.c
+++ b/net/ipv6/reassembly.c
@@ -55,11 +55,6 @@
 #include <net/addrconf.h>
 #include <net/inet_frag.h>

-int sysctl_ip6frag_high_thresh __read_mostly = 256*1024;
-int sysctl_ip6frag_low_thresh __read_mostly = 192*1024;
-
-int sysctl_ip6frag_time __read_mostly = IPV6_FRAG_TIMEOUT;
-
 struct ip6frag_skb_cb
 {
  struct inet6_skb_parm h;
@@ -86,6 +81,13 @@ struct frag_queue
  __u16   nhoffset;
 };

+struct inet_frags_ctl ip6_frags_ctl __read_mostly = {
+ .high_thresh   = 256 * 1024,
+ .low_thresh  = 192 * 1024,
+ .timeout  = IPV6_FRAG_TIMEOUT,
+ .secret_interval = 10 * 60 * HZ,
+};
+
 static struct inet_frags ip6_frags;
```

```
 int ip6_frag_nqueues(void)
@@ -143,8 +145,6 @@ static unsigned int ip6qhashfn(__be32 id, struct in6_addr *saddr,
  return c & (INETFRAGS_HASHSZ - 1);
 }

-int sysctl_ip6frag_secret_interval __read_mostly = 10 * 60 * HZ;
-
 static void ip6_frag_secret_rebuild(unsigned long dummy)
 {
  unsigned long now = jiffies;
@@ -173,7 +173,7 @@ static void ip6_frag_secret_rebuild(unsigned long dummy)
  }
  write_unlock(&ip6_frags.lock);

- mod_timer(&ip6_frags.secret_timer, now + sysctl_ip6frag_secret_interval);
+ mod_timer(&ip6_frags.secret_timer, now + ip6_frags_ctl.secret_interval);
 }

 /* Memory Tracking Functions. */
@@ -252,7 +252,7 @@ static void ip6_evictor(struct inet6_dev *idev)
  struct list_head *tmp;
  int work;

- work = atomic_read(&ip6_frags.mem) - sysctl_ip6frag_low_thresh;
+ work = atomic_read(&ip6_frags.mem) - ip6_frags_ctl.low_thresh;
  if (work <= 0)
   return;

@@ -344,7 +344,7 @@ static struct frag_queue *ip6_frag_intern(struct frag_queue *fq_in)
 #endif
  fq = fq_in;

- if (!mod_timer(&fq->q.timer, jiffies + sysctl_ip6frag_time))
+ if (!mod_timer(&fq->q.timer, jiffies + ip6_frags_ctl.timeout))
   atomic_inc(&fq->q.refcnt);

  atomic_inc(&fq->q.refcnt);
@@ -727,7 +727,7 @@ static int ipv6_frag_rcv(struct sk_buff **skbp)
   return 1;
  }

- if (atomic_read(&ip6_frags.mem) > sysctl_ip6frag_high_thresh)
+ if (atomic_read(&ip6_frags.mem) > ip6_frags_ctl.high_thresh)
   ip6_evictor(ip6_dst_idev(skb->dst));

  if ((fq = fq_find(fhdr->identification, &hdr->saddr, &hdr->daddr,
@@ -765,8 +765,9 @@ void __init ipv6_frag_init(void)
```

```
  init_timer(&ip6_frags.secret_timer);
  ip6_frags.secret_timer.function = ip6_frag_secret_rebuild;
- ip6_frags.secret_timer.expires = jiffies + sysctl_ip6frag_secret_interval;
+ ip6_frags.secret_timer.expires = jiffies + ip6_frags_ctl.secret_interval;
  add_timer(&ip6_frags.secret_timer);

+ ip6_frags.ctl = &ip6_frags_ctl;
  inet_frags_init(&ip6_frags);
 }
diff --git a/net/ipv6/sysctl_net_ipv6.c b/net/ipv6/sysctl_net_ipv6.c
index 3fb4427..97c425f 100644
--- a/net/ipv6/sysctl_net_ipv6.c
+++ b/net/ipv6/sysctl_net_ipv6.c
@@ -12,6 +12,7 @@
 #include <net/ndisc.h>
 #include <net/ipv6.h>
 #include <net/addrconf.h>
+#include <net/inet_frag.h>

 #ifdef CONFIG_SYSCTL

@@ -41,7 +42,7 @@ static ctl_table ipv6_table[] = {
  {
   .ctl_name = NET_IPV6_IP6FRAG_HIGH_THRESH,
   .procname = "ip6frag_high_thresh",
-  .data  = &sysctl_ip6frag_high_thresh,
+  .data  = &ip6_frags_ctl.high_thresh,
   .maxlen  = sizeof(int),
   .mode  = 0644,
   .proc_handler = &proc_dointvec
@@ -49,7 +50,7 @@ static ctl_table ipv6_table[] = {
  {
   .ctl_name = NET_IPV6_IP6FRAG_LOW_THRESH,
   .procname = "ip6frag_low_thresh",
-  .data  = &sysctl_ip6frag_low_thresh,
+  .data  = &ip6_frags_ctl.low_thresh,
   .maxlen  = sizeof(int),
   .mode  = 0644,
   .proc_handler = &proc_dointvec
@@ -57,7 +58,7 @@ static ctl_table ipv6_table[] = {
  {
   .ctl_name = NET_IPV6_IP6FRAG_TIME,
   .procname = "ip6frag_time",
-  .data  = &sysctl_ip6frag_time,
+  .data  = &ip6_frags_ctl.timeout,
   .maxlen  = sizeof(int),
   .mode  = 0644,
   .proc_handler = &proc_dointvec_jiffies,
```

```
@@ -66,7 +67,7 @@ static ctl_table ipv6_table[] = {
  {
   .ctl_name = NET_IPV6_IP6FRAG_SECRET_INTERVAL,
   .procname = "ip6frag_secret_interval",
-  .data  = &sysctl_ip6frag_secret_interval,
+  .data  = &ip6_frags_ctl.secret_interval,
   .maxlen  = sizeof(int),
   .mode  = 0644,
   .proc_handler = &proc_dointvec_jiffies,
--
1.5.3.4
```