Subject: [PATCH 6/9] Consolidate the xxx_frag_destroy
Posted by Pavel Emelianov on Fri, 12 Oct 2007 13:21:06 GMT
View Forum Message <> Reply to Message

To make in possible we need to know the exact frag queue
size for inet_frags->mem management and two callbacks:

 * to destoy the skb (optional, used in conntracks only)
 * to free the queue itself (mandatory, but later I plan to
   move the allocation and the destruction of frag_queues
   into the common place, so this callback will most likely
   be optional too).

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

```
diff --git a/include/net/inet_frag.h b/include/net/inet_frag.h
index e374412..2dd1cd4 100644
--- a/include/net/inet_frag.h
+++ b/include/net/inet_frag.h
@@ -33,16 +33,21 @@ struct inet_frags {
  rwlock_t  lock;
  u32   rnd;
  int   nqueues;
+ int   qsize;
  atomic_t  mem;
  struct timer_list secret_timer;
  struct inet_frags_ctl *ctl;

  unsigned int  (*hashfn)(struct inet_frag_queue *);
+ void   (*destructor)(struct inet_frag_queue *);
+ void   (*skb_free)(struct sk_buff *);
 };

 void inet_frags_init(struct inet_frags *);
 void inet_frags_fini(struct inet_frags *);

 void inet_frag_kill(struct inet_frag_queue *q, struct inet_frags *f);
+void inet_frag_destroy(struct inet_frag_queue *q,
+    struct inet_frags *f, int *work);

 #endif
diff --git a/net/ipv4/inet_fragment.c b/net/ipv4/inet_fragment.c
index ec10e05..15fb2c4 100644
--- a/net/ipv4/inet_fragment.c
+++ b/net/ipv4/inet_fragment.c
@@ -17,6 +17,8 @@
```

```
 #include <linux/timer.h>
 #include <linux/mm.h>
 #include <linux/random.h>
+#include <linux/skbuff.h>
+#include <linux/rtnetlink.h>

 #include <net/inet_frag.h>

@@ -100,3 +102,41 @@ void inet_frag_kill(struct inet_frag_queue *fq, struct inet_frags *f)
 }

 EXPORT_SYMBOL(inet_frag_kill);
+
+static inline void frag_kfree_skb(struct inet_frags *f, struct sk_buff *skb,
+      int *work)
+{
+ if (work)
+  *work -= skb->truesize;
+
+ atomic_sub(skb->truesize, &f->mem);
+ if (f->skb_free)
+  f->skb_free(skb);
+ kfree_skb(skb);
+}
+
+void inet_frag_destroy(struct inet_frag_queue *q, struct inet_frags *f,
+     int *work)
+{
+ struct sk_buff *fp;
+
+ BUG_TRAP(q->last_in & COMPLETE);
+ BUG_TRAP(del_timer(&q->timer) == 0);
+
+ /* Release all fragment data. */
+ fp = q->fragments;
+ while (fp) {
+  struct sk_buff *xp = fp->next;
+
+  frag_kfree_skb(f, fp, work);
+  fp = xp;
+ }
+
+ if (work)
+  *work -= f->qsize;
+ atomic_sub(f->qsize, &f->mem);
+
+ f->destructor(q);
+
```

```
+}
+EXPORT_SYMBOL(inet_frag_destroy);
diff --git a/net/ipv4/ip_fragment.c b/net/ipv4/ip_fragment.c
index 7aee137..a59ac39 100644
--- a/net/ipv4/ip_fragment.c
+++ b/net/ipv4/ip_fragment.c
@@ -129,11 +129,13 @@ static __inline__ void frag_kfree_skb(struct sk_buff *skb, int *work)
  kfree_skb(skb);
 }

-static __inline__ void frag_free_queue(struct ipq *qp, int *work)
+static __inline__ void ip4_frag_free(struct inet_frag_queue *q)
 {
- if (work)
-  *work -= sizeof(struct ipq);
- atomic_sub(sizeof(struct ipq), &ip4_frags.mem);
+ struct ipq *qp;
+
+ qp = container_of(q, struct ipq, q);
+ if (qp->peer)
+  inet_putpeer(qp->peer);
  kfree(qp);
 }

@@ -150,34 +152,10 @@ static __inline__ struct ipq *frag_alloc_queue(void)

 /* Destruction primitives. */

-/* Complete destruction of ipq. */
-static void ip_frag_destroy(struct ipq *qp, int *work)
-{
- struct sk_buff *fp;
-
- BUG_TRAP(qp->q.last_in&COMPLETE);
- BUG_TRAP(del_timer(&qp->q.timer) == 0);
-
- if (qp->peer)
-  inet_putpeer(qp->peer);
-
- /* Release all fragment data. */
- fp = qp->q.fragments;
- while (fp) {
-  struct sk_buff *xp = fp->next;
-
-  frag_kfree_skb(fp, work);
-  fp = xp;
- }
-
```

```
- /* Finally, release the queue descriptor itself. */
- frag_free_queue(qp, work);
-}
-
 static __inline__ void ipq_put(struct ipq *ipq, int *work)
 {
  if (atomic_dec_and_test(&ipq->q.refcnt))
-  ip_frag_destroy(ipq, work);
+  inet_frag_destroy(&ipq->q, &ip4_frags, work);
 }

 /* Kill ipq entry. It is not destroyed immediately,
@@ -687,6 +665,9 @@ void __init ipfrag_init(void)
 {
  ip4_frags.ctl = &ip4_frags_ctl;
  ip4_frags.hashfn = ip4_hashfn;
+ ip4_frags.destructor = ip4_frag_free;
+ ip4_frags.skb_free = NULL;
+ ip4_frags.qsize = sizeof(struct ipq);
  inet_frags_init(&ip4_frags);
 }

diff --git a/net/ipv6/netfilter/nf_conntrack_reasm.c b/net/ipv6/netfilter/nf_conntrack_reasm.c
index a3aef38..785f5cd 100644
--- a/net/ipv6/netfilter/nf_conntrack_reasm.c
+++ b/net/ipv6/netfilter/nf_conntrack_reasm.c
@@ -114,25 +114,25 @@ static unsigned int nf_hashfn(struct inet_frag_queue *q)
  return ip6qhashfn(nq->id, &nq->saddr, &nq->daddr);
 }

+static void nf_skb_free(struct sk_buff *skb)
+{
+ if (NFCT_FRAG6_CB(skb)->orig)
+  kfree_skb(NFCT_FRAG6_CB(skb)->orig);
+}
+
 /* Memory Tracking Functions. */
 static inline void frag_kfree_skb(struct sk_buff *skb, unsigned int *work)
 {
  if (work)
   *work -= skb->truesize;
  atomic_sub(skb->truesize, &nf_frags.mem);
- if (NFCT_FRAG6_CB(skb)->orig)
-  kfree_skb(NFCT_FRAG6_CB(skb)->orig);
-
+ nf_skb_free(skb);
  kfree_skb(skb);
 }
```

```
-static inline void frag_free_queue(struct nf_ct_frag6_queue *fq,
-       unsigned int *work)
+static void nf_frag_free(struct inet_frag_queue *q)
 {
- if (work)
-  *work -= sizeof(struct nf_ct_frag6_queue);
- atomic_sub(sizeof(struct nf_ct_frag6_queue), &nf_frags.mem);
- kfree(fq);
+ kfree(container_of(q, struct nf_ct_frag6_queue, q));
 }

 static inline struct nf_ct_frag6_queue *frag_alloc_queue(void)
@@ -147,31 +147,10 @@ static inline struct nf_ct_frag6_queue *frag_alloc_queue(void)

 /* Destruction primitives. */

-/* Complete destruction of fq. */
-static void nf_ct_frag6_destroy(struct nf_ct_frag6_queue *fq,
-    unsigned int *work)
-{
- struct sk_buff *fp;
-
- BUG_TRAP(fq->q.last_in&COMPLETE);
- BUG_TRAP(del_timer(&fq->q.timer) == 0);
-
- /* Release all fragment data. */
- fp = fq->q.fragments;
- while (fp) {
-  struct sk_buff *xp = fp->next;
-
-  frag_kfree_skb(fp, work);
-  fp = xp;
- }
-
- frag_free_queue(fq, work);
-}
-
 static __inline__ void fq_put(struct nf_ct_frag6_queue *fq, unsigned int *work)
 {
  if (atomic_dec_and_test(&fq->q.refcnt))
-  nf_ct_frag6_destroy(fq, work);
+  inet_frag_destroy(&fq->q, &nf_frags, work);
 }

 /* Kill fq entry. It is not destroyed immediately,
@@ -799,6 +778,9 @@ int nf_ct_frag6_init(void)
 {
```

```
  nf_frags.ctl = &nf_frags_ctl;
  nf_frags.hashfn = nf_hashfn;
+ nf_frags.destructor = nf_frag_free;
+ nf_frags.skb_free = nf_skb_free;
+ nf_frags.qsize = sizeof(struct nf_ct_frag6_queue);
  inet_frags_init(&nf_frags);

  return 0;
diff --git a/net/ipv6/reassembly.c b/net/ipv6/reassembly.c
index 7a357fd..74b2113 100644
--- a/net/ipv6/reassembly.c
+++ b/net/ipv6/reassembly.c
@@ -148,12 +148,9 @@ static inline void frag_kfree_skb(struct sk_buff *skb, int *work)
  kfree_skb(skb);
 }

-static inline void frag_free_queue(struct frag_queue *fq, int *work)
+static void ip6_frag_free(struct inet_frag_queue *fq)
 {
- if (work)
-  *work -= sizeof(struct frag_queue);
- atomic_sub(sizeof(struct frag_queue), &ip6_frags.mem);
- kfree(fq);
+ kfree(container_of(fq, struct frag_queue, q));
 }

 static inline struct frag_queue *frag_alloc_queue(void)
@@ -168,30 +165,10 @@ static inline struct frag_queue *frag_alloc_queue(void)

 /* Destruction primitives. */

-/* Complete destruction of fq. */
-static void ip6_frag_destroy(struct frag_queue *fq, int *work)
-{
- struct sk_buff *fp;
-
- BUG_TRAP(fq->q.last_in&COMPLETE);
- BUG_TRAP(del_timer(&fq->q.timer) == 0);
-
- /* Release all fragment data. */
- fp = fq->q.fragments;
- while (fp) {
-  struct sk_buff *xp = fp->next;
-
-  frag_kfree_skb(fp, work);
-  fp = xp;
- }
-
```

```
- frag_free_queue(fq, work);
-}
-
 static __inline__ void fq_put(struct frag_queue *fq, int *work)
 {
  if (atomic_dec_and_test(&fq->q.refcnt))
-  ip6_frag_destroy(fq, work);
+  inet_frag_destroy(&fq->q, &ip6_frags, work);
 }

 /* Kill fq entry. It is not destroyed immediately,
@@ -721,5 +698,8 @@ void __init ipv6_frag_init(void)

  ip6_frags.ctl = &ip6_frags_ctl;
  ip6_frags.hashfn = ip6_hashfn;
+ ip6_frags.destructor = ip6_frag_free;
+ ip6_frags.skb_free = NULL;
+ ip6_frags.qsize = sizeof(struct frag_queue);
  inet_frags_init(&ip6_frags);
 }
--
1.5.3.4
```