Subject: [PATCH 4/7] Consolidate xxx_frag_create()
Posted by Pavel Emelianov on Tue, 16 Oct 2007 14:00:10 GMT
View Forum Message <> Reply to Message

This one uses the xxx_frag_intern() and xxx_frag_alloc()
routines, which are already consolidated, so remove them
from protocol code (as promised).

The ->constructor callback is used to init the rest of
the frag queue and it is the same for netfilter and ipv6.

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

```
diff --git a/include/net/inet_frag.h b/include/net/inet_frag.h
index 412b858..e33072b 100644
--- a/include/net/inet_frag.h
+++ b/include/net/inet_frag.h
@@ -39,6 +39,8 @@ struct inet_frags {
  struct inet_frags_ctl *ctl;

  unsigned int  (*hashfn)(struct inet_frag_queue *);
+ void   (*constructor)(struct inet_frag_queue *q,
+       void *arg);
  void   (*destructor)(struct inet_frag_queue *);
  void   (*skb_free)(struct sk_buff *);
  int   (*equal)(struct inet_frag_queue *q1,
@@ -53,9 +55,8 @@ 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);
 int inet_frag_evictor(struct inet_frags *f);
-struct inet_frag_queue *inet_frag_intern(struct inet_frag_queue *q,
-  struct inet_frags *f, unsigned int hash);
-struct inet_frag_queue *inet_frag_alloc(struct inet_frags *f);
+struct inet_frag_queue *inet_frag_create(struct inet_frags *f,
+  void *create_arg, unsigned int hash);

 static inline void inet_frag_put(struct inet_frag_queue *q, struct inet_frags *f)
 {
diff --git a/include/net/ipv6.h b/include/net/ipv6.h
index ff12697..9dc99bf 100644
--- a/include/net/ipv6.h
+++ b/include/net/ipv6.h
@@ -380,6 +380,14 @@ static inline int ipv6_prefix_equal(const struct in6_addr *a1,
 struct inet_frag_queue;
 int ip6_frag_equal(struct inet_frag_queue *q1, struct inet_frag_queue *q2);
```

```
+struct ip6_create_arg {
+ __be32 id;
+ struct in6_addr *src;
+ struct in6_addr *dst;
+};
+
+void ip6_frag_init(struct inet_frag_queue *q, void *a);
+
 static inline int ipv6_addr_any(const struct in6_addr *a)
 {
  return ((a->s6_addr32[0] | a->s6_addr32[1] |
diff --git a/net/ipv4/inet_fragment.c b/net/ipv4/inet_fragment.c
index 22539fb..0124885 100644
--- a/net/ipv4/inet_fragment.c
+++ b/net/ipv4/inet_fragment.c
@@ -173,7 +173,7 @@ int inet_frag_evictor(struct inet_frags *f)
 }
 EXPORT_SYMBOL(inet_frag_evictor);

-struct inet_frag_queue *inet_frag_intern(struct inet_frag_queue *qp_in,
+static struct inet_frag_queue *inet_frag_intern(struct inet_frag_queue *qp_in,
   struct inet_frags *f, unsigned int hash)
 {
  struct inet_frag_queue *qp;
@@ -208,9 +208,8 @@ struct inet_frag_queue *inet_frag_intern(struct inet_frag_queue *qp_in,
  write_unlock(&f->lock);
  return qp;
 }
-EXPORT_SYMBOL(inet_frag_intern);

-struct inet_frag_queue *inet_frag_alloc(struct inet_frags *f)
+static struct inet_frag_queue *inet_frag_alloc(struct inet_frags *f, void *arg)
 {
  struct inet_frag_queue *q;

@@ -218,6 +217,7 @@ struct inet_frag_queue *inet_frag_alloc(struct inet_frags *f)
  if (q == NULL)
   return NULL;

+ f->constructor(q, arg);
  atomic_add(f->qsize, &f->mem);
  setup_timer(&q->timer, f->frag_expire, (unsigned long)q);
  spin_lock_init(&q->lock);
@@ -225,4 +225,16 @@ struct inet_frag_queue *inet_frag_alloc(struct inet_frags *f)

  return q;
 }
-EXPORT_SYMBOL(inet_frag_alloc);
```

```
+
+struct inet_frag_queue *inet_frag_create(struct inet_frags *f, void *arg,
+  unsigned int hash)
+{
+ struct inet_frag_queue *q;
+
+ q = inet_frag_alloc(f, arg);
+ if (q == NULL)
+  return NULL;
+
+ return inet_frag_intern(q, f, hash);
+}
+EXPORT_SYMBOL(inet_frag_create);
diff --git a/net/ipv4/ip_fragment.c b/net/ipv4/ip_fragment.c
index 2cf8cdc..db29c3c 100644
--- a/net/ipv4/ip_fragment.c
+++ b/net/ipv4/ip_fragment.c
@@ -108,6 +108,11 @@ int ip_frag_mem(void)
 static int ip_frag_reasm(struct ipq *qp, struct sk_buff *prev,
     struct net_device *dev);

+struct ip4_create_arg {
+ struct iphdr *iph;
+ u32 user;
+};
+
 static unsigned int ipqhashfn(__be16 id, __be32 saddr, __be32 daddr, u8 prot)
 {
  return jhash_3words((__force u32)id << 16 | prot,
@@ -146,6 +151,20 @@ static __inline__ void frag_kfree_skb(struct sk_buff *skb, int *work)
  kfree_skb(skb);
 }

+static void ip4_frag_init(struct inet_frag_queue *q, void *a)
+{
+ struct ipq *qp = container_of(q, struct ipq, q);
+ struct ip4_create_arg *arg = a;
+
+ qp->protocol = arg->iph->protocol;
+ qp->id = arg->iph->id;
+ qp->saddr = arg->iph->saddr;
+ qp->daddr = arg->iph->daddr;
+ qp->user = arg->user;
+ qp->peer = sysctl_ipfrag_max_dist ?
+  inet_getpeer(arg->iph->saddr, 1) : NULL;
+}
+
 static __inline__ void ip4_frag_free(struct inet_frag_queue *q)
```

```
 {
  struct ipq *qp;
@@ -156,14 +175,6 @@ static __inline__ void ip4_frag_free(struct inet_frag_queue *q)
  kfree(qp);
 }

-static __inline__ struct ipq *frag_alloc_queue(void)
-{
- struct inet_frag_queue *q;
-
- q = inet_frag_alloc(&ip4_frags);
- return q ? container_of(q, struct ipq, q) : NULL;
-}
-

 /* Destruction primitives. */

@@ -226,30 +237,20 @@ out:

 /* Creation primitives. */

-static struct ipq *ip_frag_intern(struct ipq *qp_in, unsigned int hash)
-{
- struct inet_frag_queue *q;
-
- q = inet_frag_intern(&qp_in->q, &ip4_frags, hash);
- return container_of(q, struct ipq, q);
-}
-
 /* Add an entry to the 'ipq' queue for a newly received IP datagram. */
 static struct ipq *ip_frag_create(struct iphdr *iph, u32 user, unsigned int h)
 {
- struct ipq *qp;
+ struct inet_frag_queue *q;
+ struct ip4_create_arg arg;

- if ((qp = frag_alloc_queue()) == NULL)
-  goto out_nomem;
+ arg.iph = iph;
+ arg.user = user;

- qp->protocol = iph->protocol;
- qp->id = iph->id;
- qp->saddr = iph->saddr;
- qp->daddr = iph->daddr;
- qp->user = user;
- qp->peer = sysctl_ipfrag_max_dist ? inet_getpeer(iph->saddr, 1) : NULL;
+ q = inet_frag_create(&ip4_frags, &arg, h);
```

```
+ if (q == NULL)
+  goto out_nomem;

- return ip_frag_intern(qp, h);
+ return container_of(q, struct ipq, q);

 out_nomem:
  LIMIT_NETDEBUG(KERN_ERR "ip_frag_create: no memory left !\n");
@@ -642,6 +643,7 @@ void __init ipfrag_init(void)
 {
  ip4_frags.ctl = &ip4_frags_ctl;
  ip4_frags.hashfn = ip4_hashfn;
+ ip4_frags.constructor = ip4_frag_init;
  ip4_frags.destructor = ip4_frag_free;
  ip4_frags.skb_free = NULL;
  ip4_frags.qsize = sizeof(struct ipq);
diff --git a/net/ipv6/netfilter/nf_conntrack_reasm.c b/net/ipv6/netfilter/nf_conntrack_reasm.c
index 29a42d2..72451e2 100644
--- a/net/ipv6/netfilter/nf_conntrack_reasm.c
+++ b/net/ipv6/netfilter/nf_conntrack_reasm.c
@@ -135,14 +135,6 @@ static void nf_frag_free(struct inet_frag_queue *q)
  kfree(container_of(q, struct nf_ct_frag6_queue, q));
 }

-static inline struct nf_ct_frag6_queue *frag_alloc_queue(void)
-{
- struct inet_frag_queue *q;
-
- q = inet_frag_alloc(&nf_frags);
- return q ? container_of(q, struct nf_ct_frag6_queue, q) : NULL;
-}
-
 /* Destruction primitives. */

 static __inline__ void fq_put(struct nf_ct_frag6_queue *fq)
@@ -184,33 +176,25 @@ out:

 /* Creation primitives. */

-static struct nf_ct_frag6_queue *nf_ct_frag6_intern(unsigned int hash,
-       struct nf_ct_frag6_queue *fq_in)
+static struct nf_ct_frag6_queue *
+nf_ct_frag6_create(unsigned int hash, __be32 id, struct in6_addr *src,
+  struct in6_addr *dst)
 {
  struct inet_frag_queue *q;
+ struct ip6_create_arg arg;
```

```
- q = inet_frag_intern(&fq_in->q, &nf_frags, hash);
- return container_of(q, struct nf_ct_frag6_queue, q);
-}
-
-
-static struct nf_ct_frag6_queue *
-nf_ct_frag6_create(unsigned int hash, __be32 id, struct in6_addr *src,       struct in6_addr *dst)
-{
- struct nf_ct_frag6_queue *fq;
+ arg.id = id;
+ arg.src = src;
+ arg.dst = dst;

- if ((fq = frag_alloc_queue()) == NULL) {
-  pr_debug("Can't alloc new queue\n");
+ q = inet_frag_create(&nf_frags, &arg, hash);
+ if (q == NULL)
   goto oom;
- }

- fq->id = id;
- ipv6_addr_copy(&fq->saddr, src);
- ipv6_addr_copy(&fq->daddr, dst);
-
- return nf_ct_frag6_intern(hash, fq);
+ return container_of(q, struct nf_ct_frag6_queue, q);

 oom:
+ pr_debug("Can't alloc new queue\n");
  return NULL;
 }

@@ -718,6 +702,7 @@ int nf_ct_frag6_init(void)
 {
  nf_frags.ctl = &nf_frags_ctl;
  nf_frags.hashfn = nf_hashfn;
+ nf_frags.constructor = ip6_frag_init;
  nf_frags.destructor = nf_frag_free;
  nf_frags.skb_free = nf_skb_free;
  nf_frags.qsize = sizeof(struct nf_ct_frag6_queue);
diff --git a/net/ipv6/reassembly.c b/net/ipv6/reassembly.c
index 21913c7..ce87340 100644
--- a/net/ipv6/reassembly.c
+++ b/net/ipv6/reassembly.c
@@ -164,17 +164,20 @@ static inline void frag_kfree_skb(struct sk_buff *skb, int *work)
  kfree_skb(skb);
 }
```

```
-static void ip6_frag_free(struct inet_frag_queue *fq)
+void ip6_frag_init(struct inet_frag_queue *q, void *a)
 {
- kfree(container_of(fq, struct frag_queue, q));
+ struct frag_queue *fq = container_of(q, struct frag_queue, q);
+ struct ip6_create_arg *arg = a;
+
+ fq->id = arg->id;
+ ipv6_addr_copy(&fq->saddr, arg->src);
+ ipv6_addr_copy(&fq->daddr, arg->dst);
 }
+EXPORT_SYMBOL(ip6_frag_init);

-static inline struct frag_queue *frag_alloc_queue(void)
+static void ip6_frag_free(struct inet_frag_queue *fq)
 {
- struct inet_frag_queue *q;
-
- q = inet_frag_alloc(&ip6_frags);
- return q ? container_of(q, struct frag_queue, q) : NULL;
+ kfree(container_of(fq, struct frag_queue, q));
 }

 /* Destruction primitives. */
@@ -244,31 +247,22 @@ out:

 /* Creation primitives. */

-
-static struct frag_queue *ip6_frag_intern(struct frag_queue *fq_in,
-  unsigned int hash)
-{
- struct inet_frag_queue *q;
-
- q = inet_frag_intern(&fq_in->q, &ip6_frags, hash);
- return container_of(q, struct frag_queue, q);
-}
-
-
 static struct frag_queue *
 ip6_frag_create(__be32 id, struct in6_addr *src, struct in6_addr *dst,
   struct inet6_dev *idev, unsigned int hash)
 {
- struct frag_queue *fq;
+ struct inet_frag_queue *q;
+ struct ip6_create_arg arg;

- if ((fq = frag_alloc_queue()) == NULL)
```

```
-  goto oom;
+ arg.id = id;
+ arg.src = src;
+ arg.dst = dst;

- fq->id = id;
- ipv6_addr_copy(&fq->saddr, src);
- ipv6_addr_copy(&fq->daddr, dst);
+ q = inet_frag_create(&ip6_frags, &arg, hash);
+ if (q == NULL)
+  goto oom;

- return ip6_frag_intern(fq, hash);
+ return container_of(q, struct frag_queue, q);

 oom:
  IP6_INC_STATS_BH(idev, IPSTATS_MIB_REASMFAILS);
@@ -675,6 +669,7 @@ void __init ipv6_frag_init(void)

  ip6_frags.ctl = &ip6_frags_ctl;
  ip6_frags.hashfn = ip6_hashfn;
+ ip6_frags.constructor = ip6_frag_init;
  ip6_frags.destructor = ip6_frag_free;
  ip6_frags.skb_free = NULL;
  ip6_frags.qsize = sizeof(struct frag_queue);
--
1.5.3.4
```