Subject: [PATCH 5/6][NETNS]: Tcp-v6 sockets per-net lookup.
Posted by Pavel Emelianov on Thu, 31 Jan 2008 12:40:16 GMT
View Forum Message <> Reply to Message

Add a net argument to inet6_lookup and propagate it further.
Actually, this is tcp-v6 implementation of what was done for
tcp-v4 sockets in a previous patch.

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

---
 include/linux/ipv6.h | 8 ++++----
 include/net/inet6_hashtables.h | 17 ++++++++++-------
 net/dccp/ipv6.c | 8 ++++----
 net/ipv4/inet_diag.c | 2 +-
 net/ipv6/inet6_hashtables.c | 25 ++++++++++++++++---------
 net/ipv6/tcp_ipv6.c | 19 ++++++++++---------
 6 files changed, 43 insertions(+), 36 deletions(-)

diff --git a/include/linux/ipv6.h b/include/linux/ipv6.h
index c347860..4aaefc3 100644
--- a/include/linux/ipv6.h
+++ b/include/linux/ipv6.h
@@ -457,16 +457,16 @@ static inline struct raw6_sock *raw6_sk(const struct sock *sk)
 #define inet_v6_ipv6only(__sk) 0
 #endif /* defined(CONFIG_IPV6) || defined(CONFIG_IPV6_MODULE) */

-#define INET6_MATCH(__sk, __hash, __saddr, __daddr, __ports, __dif)\
- (((__sk)->sk_hash == (__hash))   && \
+#define INET6_MATCH(__sk, __net, __hash, __saddr, __daddr, __ports, __dif)\
+ (((__sk)->sk_hash == (__hash)) && ((__sk)->sk_net == (__net)) && \
   ((*((__portpair *)&(inet_sk(__sk)->dport))) == (__ports))   && \
   ((__sk)->sk_family == AF_INET6) && \
   ipv6_addr_equal(&inet6_sk(__sk)->daddr, (__saddr)) && \
   ipv6_addr_equal(&inet6_sk(__sk)->rcv_saddr, (__daddr)) && \
   (!((__sk)->sk_bound_dev_if) || ((__sk)->sk_bound_dev_if == (__dif))))

-#define INET6_TW_MATCH(__sk, __hash, __saddr, __daddr, __ports, __dif) \
- (((__sk)->sk_hash == (__hash))   && \
+#define INET6_TW_MATCH(__sk, __net, __hash, __saddr, __daddr, __ports, __dif) \
+ (((__sk)->sk_hash == (__hash)) && ((__sk)->sk_net == (__net)) && \
   (*((__portpair *)&(inet_twsk(__sk)->tw_dport)) == (__ports)) && \
   ((__sk)->sk_family   == PF_INET6)   && \
   (ipv6_addr_equal(&inet6_twsk(__sk)->tw_v6_daddr, (__saddr))) && \
diff --git a/include/net/inet6_hashtables.h b/include/net/inet6_hashtables.h
index 668056b..fdff630 100644
--- a/include/net/inet6_hashtables.h
+++ b/include/net/inet6_hashtables.h

```
@@ -57,34 +57,37 @@ extern void __inet6_hash(struct inet_hashinfo *hashinfo, struct sock *sk);
  *
  * The sockhash lock must be held as a reader here.
  */
-extern struct sock *__inet6_lookup_established(struct inet_hashinfo *hashinfo,
+extern struct sock *__inet6_lookup_established(struct net *net,
+        struct inet_hashinfo *hashinfo,
         const struct in6_addr *saddr,
         const __be16 sport,
         const struct in6_addr *daddr,
         const u16 hnum,
         const int dif);

-extern struct sock *inet6_lookup_listener(struct inet_hashinfo *hashinfo,
+extern struct sock *inet6_lookup_listener(struct net *net,
+       struct inet_hashinfo *hashinfo,
        const struct in6_addr *daddr,
        const unsigned short hnum,
        const int dif);

-static inline struct sock *__inet6_lookup(struct inet_hashinfo *hashinfo,
+static inline struct sock *__inet6_lookup(struct net *net,
+       struct inet_hashinfo *hashinfo,
        const struct in6_addr *saddr,
        const __be16 sport,
        const struct in6_addr *daddr,
        const u16 hnum,
        const int dif)
 {
- struct sock *sk = __inet6_lookup_established(hashinfo, saddr, sport,
-           daddr, hnum, dif);
+ struct sock *sk = __inet6_lookup_established(net, hashinfo, saddr,
+      sport, daddr, hnum, dif);
  if (sk)
   return sk;

- return inet6_lookup_listener(hashinfo, daddr, hnum, dif);
+ return inet6_lookup_listener(net, hashinfo, daddr, hnum, dif);
 }

-extern struct sock *inet6_lookup(struct inet_hashinfo *hashinfo,
+extern struct sock *inet6_lookup(struct net *net, struct inet_hashinfo *hashinfo,
      const struct in6_addr *saddr, const __be16 sport,
      const struct in6_addr *daddr, const __be16 dport,
      const int dif);
diff --git a/net/dccp/ipv6.c b/net/dccp/ipv6.c
index f42b75c..ed0a005 100644
--- a/net/dccp/ipv6.c
```

```
+++ b/net/dccp/ipv6.c
@@ -101,8 +101,8 @@ static void dccp_v6_err(struct sk_buff *skb, struct inet6_skb_parm *opt,
  int err;
  __u64 seq;

- sk = inet6_lookup(&dccp_hashinfo, &hdr->daddr, dh->dccph_dport,
-     &hdr->saddr, dh->dccph_sport, inet6_iif(skb));
+ sk = inet6_lookup(&init_net, &dccp_hashinfo, &hdr->daddr, dh->dccph_dport,
+   &hdr->saddr, dh->dccph_sport, inet6_iif(skb));

  if (sk == NULL) {
   ICMP6_INC_STATS_BH(__in6_dev_get(skb->dev), ICMP6_MIB_INERRORS);
@@ -366,7 +366,7 @@ static struct sock *dccp_v6_hnd_req(struct sock *sk,struct sk_buff *skb)
  if (req != NULL)
   return dccp_check_req(sk, skb, req, prev);

- nsk = __inet6_lookup_established(&dccp_hashinfo,
+ nsk = __inet6_lookup_established(&init_net, &dccp_hashinfo,
       &iph->saddr, dh->dccph_sport,
       &iph->daddr, ntohs(dh->dccph_dport),
       inet6_iif(skb));
@@ -797,7 +797,7 @@ static int dccp_v6_rcv(struct sk_buff *skb)

  /* Step 2:
   * Look up flow ID in table and get corresponding socket */
- sk = __inet6_lookup(&dccp_hashinfo, &ipv6_hdr(skb)->saddr,
+ sk = __inet6_lookup(&init_net, &dccp_hashinfo, &ipv6_hdr(skb)->saddr,
        dh->dccph_sport,
        &ipv6_hdr(skb)->daddr, ntohs(dh->dccph_dport),
        inet6_iif(skb));
diff --git a/net/ipv4/inet_diag.c b/net/ipv4/inet_diag.c
index 95c9f14..da97695 100644
--- a/net/ipv4/inet_diag.c
+++ b/net/ipv4/inet_diag.c
@@ -274,7 +274,7 @@ static int inet_diag_get_exact(struct sk_buff *in_skb,
  }
 #if defined(CONFIG_IPV6) || defined (CONFIG_IPV6_MODULE)
  else if (req->idiag_family == AF_INET6) {
-  sk = inet6_lookup(hashinfo,
+  sk = inet6_lookup(&init_net, hashinfo,
       (struct in6_addr *)req->id.idiag_dst,
       req->id.idiag_dport,
       (struct in6_addr *)req->id.idiag_src,
diff --git a/net/ipv6/inet6_hashtables.c b/net/ipv6/inet6_hashtables.c
index ece6d0e..d325a99 100644
--- a/net/ipv6/inet6_hashtables.c
+++ b/net/ipv6/inet6_hashtables.c
@@ -54,7 +54,8 @@ EXPORT_SYMBOL(__inet6_hash);
```

```
 *
 * The sockhash lock must be held as a reader here.
 */
-struct sock *__inet6_lookup_established(struct inet_hashinfo *hashinfo,
+struct sock *__inet6_lookup_established(struct net *net,
+     struct inet_hashinfo *hashinfo,
        const struct in6_addr *saddr,
        const __be16 sport,
        const struct in6_addr *daddr,
@@ -75,12 +76,12 @@ struct sock *__inet6_lookup_established(struct inet_hashinfo *hashinfo,
 read_lock(lock);
 sk_for_each(sk, node, &head->chain) {
  /* For IPV6 do the cheaper port and family tests first. */
-  if (INET6_MATCH(sk, hash, saddr, daddr, ports, dif))
+  if (INET6_MATCH(sk, net, hash, saddr, daddr, ports, dif))
    goto hit; /* You sunk my battleship! */
 }
 /* Must check for a TIME_WAIT'er before going to listener hash. */
 sk_for_each(sk, node, &head->twchain) {
-  if (INET6_TW_MATCH(sk, hash, saddr, daddr, ports, dif))
+  if (INET6_TW_MATCH(sk, net, hash, saddr, daddr, ports, dif))
    goto hit;
 }
 read_unlock(lock);
@@ -93,9 +94,9 @@ hit:
 }
 EXPORT_SYMBOL(__inet6_lookup_established);

-struct sock *inet6_lookup_listener(struct inet_hashinfo *hashinfo,
-       const struct in6_addr *daddr,
-       const unsigned short hnum, const int dif)
+struct sock *inet6_lookup_listener(struct net *net,
+  struct inet_hashinfo *hashinfo, const struct in6_addr *daddr,
+  const unsigned short hnum, const int dif)
 {
 struct sock *sk;
 const struct hlist_node *node;
@@ -104,7 +105,8 @@ struct sock *inet6_lookup_listener(struct inet_hashinfo *hashinfo,

 read_lock(&hashinfo->lhash_lock);
 sk_for_each(sk, node, &hashinfo->listening_hash[inet_lhashfn(hnum)]) {
-  if (inet_sk(sk)->num == hnum && sk->sk_family == PF_INET6) {
+  if (sk->sk_net == net && inet_sk(sk)->num == hnum &&
+    sk->sk_family == PF_INET6) {
    const struct ipv6_pinfo *np = inet6_sk(sk);

    score = 1;
@@ -136,7 +138,7 @@ struct sock *inet6_lookup_listener(struct inet_hashinfo *hashinfo,
```

```
 EXPORT_SYMBOL_GPL(inet6_lookup_listener);

-struct sock *inet6_lookup(struct inet_hashinfo *hashinfo,
+struct sock *inet6_lookup(struct net *net, struct inet_hashinfo *hashinfo,
       const struct in6_addr *saddr, const __be16 sport,
       const struct in6_addr *daddr, const __be16 dport,
       const int dif)
@@ -144,7 +146,7 @@ struct sock *inet6_lookup(struct inet_hashinfo *hashinfo,
  struct sock *sk;

  local_bh_disable();
- sk = __inet6_lookup(hashinfo, saddr, sport, daddr, ntohs(dport), dif);
+ sk = __inet6_lookup(net, hashinfo, saddr, sport, daddr, ntohs(dport), dif);
  local_bh_enable();

  return sk;
@@ -170,6 +172,7 @@ static int __inet6_check_established(struct inet_timewait_death_row
*death_row,
  struct sock *sk2;
  const struct hlist_node *node;
  struct inet_timewait_sock *tw;
+ struct net *net = sk->sk_net;

  prefetch(head->chain.first);
  write_lock(lock);
@@ -178,7 +181,7 @@ static int __inet6_check_established(struct inet_timewait_death_row
*death_row,
  sk_for_each(sk2, node, &head->twchain) {
   tw = inet_twsk(sk2);

-  if (INET6_TW_MATCH(sk2, hash, saddr, daddr, ports, dif)) {
+  if (INET6_TW_MATCH(sk2, net, hash, saddr, daddr, ports, dif)) {
    if (twsk_unique(sk, sk2, twp))
     goto unique;
    else
@@ -189,7 +192,7 @@ static int __inet6_check_established(struct inet_timewait_death_row
*death_row,

  /* And established part... */
  sk_for_each(sk2, node, &head->chain) {
-  if (INET6_MATCH(sk2, hash, saddr, daddr, ports, dif))
+  if (INET6_MATCH(sk2, net, hash, saddr, daddr, ports, dif))
    goto not_unique;
  }

diff --git a/net/ipv6/tcp_ipv6.c b/net/ipv6/tcp_ipv6.c
index 00c0839..59d0029 100644
```

```
--- a/net/ipv6/tcp_ipv6.c
+++ b/net/ipv6/tcp_ipv6.c
@@ -330,8 +330,8 @@ static void tcp_v6_err(struct sk_buff *skb, struct inet6_skb_parm *opt,
  struct tcp_sock *tp;
  __u32 seq;

- sk = inet6_lookup(&tcp_hashinfo, &hdr->daddr, th->dest, &hdr->saddr,
-     th->source, skb->dev->ifindex);
+ sk = inet6_lookup(skb->dev->nd_net, &tcp_hashinfo, &hdr->daddr,
+    th->dest, &hdr->saddr, th->source, skb->dev->ifindex);

  if (sk == NULL) {
   ICMP6_INC_STATS_BH(__in6_dev_get(skb->dev), ICMP6_MIB_INERRORS);
@@ -1208,9 +1208,9 @@ static struct sock *tcp_v6_hnd_req(struct sock *sk,struct sk_buff *skb)
  if (req)
   return tcp_check_req(sk, skb, req, prev);

- nsk = __inet6_lookup_established(&tcp_hashinfo, &ipv6_hdr(skb)->saddr,
-      th->source, &ipv6_hdr(skb)->daddr,
-      ntohs(th->dest), inet6_iif(skb));
+ nsk = __inet6_lookup_established(sk->sk_net, &tcp_hashinfo,
+    &ipv6_hdr(skb)->saddr, th->source,
+    &ipv6_hdr(skb)->daddr, ntohs(th->dest), inet6_iif(skb));

  if (nsk) {
   if (nsk->sk_state != TCP_TIME_WAIT) {
@@ -1710,9 +1710,10 @@ static int tcp_v6_rcv(struct sk_buff *skb)
  TCP_SKB_CB(skb)->flags = ipv6_get_dsfield(ipv6_hdr(skb));
  TCP_SKB_CB(skb)->sacked = 0;

- sk = __inet6_lookup(&tcp_hashinfo, &ipv6_hdr(skb)->saddr, th->source,
-       &ipv6_hdr(skb)->daddr, ntohs(th->dest),
-       inet6_iif(skb));
+ sk = __inet6_lookup(skb->dev->nd_net, &tcp_hashinfo,
+    &ipv6_hdr(skb)->saddr, th->source,
+    &ipv6_hdr(skb)->daddr, ntohs(th->dest),
+    inet6_iif(skb));

  if (!sk)
   goto no_tcp_socket;
@@ -1792,7 +1793,7 @@ do_time_wait:
  {
   struct sock *sk2;

-  sk2 = inet6_lookup_listener(&tcp_hashinfo,
+  sk2 = inet6_lookup_listener(skb->dev->nd_net, &tcp_hashinfo,
         &ipv6_hdr(skb)->daddr,
         ntohs(th->dest), inet6_iif(skb));
```

if (sk2 != NULL) {
--
1.5.3.4

Subject: Re: [PATCH 5/6][NETNS]: Tcp-v6 sockets per-net lookup.
Posted by davem on Thu, 31 Jan 2008 13:07:20 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Thu, 31 Jan 2008 15:40:16 +0300

> Add a net argument to inet6_lookup and propagate it further.
> Actually, this is tcp-v6 implementation of what was done for
> tcp-v4 sockets in a previous patch.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Applied.
