Subject: [PATCH net-2.6.25 8/19] [NETNS] Add netns parameter to fib_get_table/fib_new_table.
Posted by den on Wed, 19 Dec 2007 15:24:32 GMT
View Forum Message <> Reply to Message

This is a central entry point to the fib storage after the previous patch.

Acked-by: Benjamin Thery <benjamin.thery@bull.net>
Signed-off-by: Denis V. Lunev <den@openvz.org>
---
 include/net/ip_fib.h | 16 ++++++++--------
 net/ipv4/fib_frontend.c | 24 ++++++++++++++----------
 net/ipv4/fib_hash.c | 4 ++--
 net/ipv4/fib_rules.c | 12 ++++++------
 net/ipv4/fib_trie.c | 8 ++++----
 5 files changed, 32 insertions(+), 32 deletions(-)

diff --git a/include/net/ip_fib.h b/include/net/ip_fib.h
index 06e6a33..1faad1f 100644
--- a/include/net/ip_fib.h
+++ b/include/net/ip_fib.h
@@ -165,7 +165,7 @@ struct fib_table {
 #define TABLE_LOCAL_INDEX 0
 #define TABLE_MAIN_INDEX 1

-static inline struct fib_table *fib_get_table(u32 id)
+static inline struct fib_table *fib_get_table(struct net *net, u32 id)
 {
  struct hlist_head *ptr;

@@ -175,20 +175,20 @@ static inline struct fib_table *fib_get_table(u32 id)
  return hlist_entry(ptr->first, struct fib_table, tb_hlist);
 }

-static inline struct fib_table *fib_new_table(u32 id)
+static inline struct fib_table *fib_new_table(struct net *net, u32 id)
 {
- return fib_get_table(id);
+ return fib_get_table(net, id);
 }

 static inline int fib_lookup(const struct flowi *flp, struct fib_result *res)
 {
  struct fib_table *table;

- table = fib_get_table(RT_TABLE_LOCAL);
+ table = fib_get_table(&init_net, RT_TABLE_LOCAL);
  if (!table->tb_lookup(table, flp, res))

```
   return 0;

- table = fib_get_table(RT_TABLE_MAIN);
+ table = fib_get_table(&init_net, RT_TABLE_MAIN);
  if (!table->tb_lookup(table, flp, res))
   return 0;
  return -ENETUNREACH;
@@ -197,7 +197,7 @@ static inline int fib_lookup(const struct flowi *flp, struct fib_result *res)
 static inline void fib_select_default(const struct flowi *flp,
          struct fib_result *res)
 {
- struct fib_table *table = fib_get_table(RT_TABLE_MAIN);
+ struct fib_table *table = fib_get_table(&init_net, RT_TABLE_MAIN);
  if (FIB_RES_GW(*res) && FIB_RES_NH(*res).nh_scope == RT_SCOPE_LINK)
   table->tb_select_default(table, flp, res);
 }
@@ -212,8 +212,8 @@ extern u32 fib_rules_tclass(struct fib_result *res);

 extern int fib_lookup(struct flowi *flp, struct fib_result *res);

-extern struct fib_table *fib_new_table(u32 id);
-extern struct fib_table *fib_get_table(u32 id);
+extern struct fib_table *fib_new_table(struct net *net, u32 id);
+extern struct fib_table *fib_get_table(struct net *net, u32 id);
 extern void fib_select_default(const struct flowi *flp, struct fib_result *res);

 #endif /* CONFIG_IP_MULTIPLE_TABLES */
diff --git a/net/ipv4/fib_frontend.c b/net/ipv4/fib_frontend.c
index df9716f..2c20908 100644
--- a/net/ipv4/fib_frontend.c
+++ b/net/ipv4/fib_frontend.c
@@ -78,14 +78,14 @@ fail:
 }
 #else

-struct fib_table *fib_new_table(u32 id)
+struct fib_table *fib_new_table(struct net *net, u32 id)
 {
  struct fib_table *tb;
  unsigned int h;

  if (id == 0)
   id = RT_TABLE_MAIN;
- tb = fib_get_table(id);
+ tb = fib_get_table(net, id);
  if (tb)
   return tb;
  tb = fib_hash_init(id);
```

```
@@ -96,7 +96,7 @@ struct fib_table *fib_new_table(u32 id)
  return tb;
 }

-struct fib_table *fib_get_table(u32 id)
+struct fib_table *fib_get_table(struct net *net, u32 id)
 {
  struct fib_table *tb;
  struct hlist_node *node;
@@ -148,7 +148,7 @@ struct net_device * ip_dev_find(__be32 addr)
  res.r = NULL;
 #endif

- local_table = fib_get_table(RT_TABLE_LOCAL);
+ local_table = fib_get_table(&init_net, RT_TABLE_LOCAL);
  if (!local_table || local_table->tb_lookup(local_table, &fl, &res))
   return NULL;
  if (res.type != RTN_LOCAL)
@@ -183,7 +183,7 @@ static inline unsigned __inet_dev_addr_type(const struct net_device
*dev,
  res.r = NULL;
 #endif

- local_table = fib_get_table(RT_TABLE_LOCAL);
+ local_table = fib_get_table(&init_net, RT_TABLE_LOCAL);
  if (local_table) {
   ret = RTN_UNICAST;
   if (!local_table->tb_lookup(local_table, &fl, &res)) {
@@ -453,13 +453,13 @@ int ip_rt_ioctl(unsigned int cmd, void __user *arg)
    struct fib_table *tb;

    if (cmd == SIOCDELRT) {
-    tb = fib_get_table(cfg.fc_table);
+    tb = fib_get_table(&init_net, cfg.fc_table);
     if (tb)
      err = tb->tb_delete(tb, &cfg);
     else
      err = -ESRCH;
    } else {
-    tb = fib_new_table(cfg.fc_table);
+    tb = fib_new_table(&init_net, cfg.fc_table);
     if (tb)
      err = tb->tb_insert(tb, &cfg);
     else
@@ -573,7 +573,7 @@ static int inet_rtm_delroute(struct sk_buff *skb, struct nlmsghdr* nlh, void
*ar
  if (err < 0)
   goto errout;
```

```
- tb = fib_get_table(cfg.fc_table);
+ tb = fib_get_table(net, cfg.fc_table);
  if (tb == NULL) {
   err = -ESRCH;
   goto errout;
@@ -598,7 +598,7 @@ static int inet_rtm_newroute(struct sk_buff *skb, struct nlmsghdr* nlh,
void *ar
  if (err < 0)
   goto errout;

- tb = fib_new_table(cfg.fc_table);
+ tb = fib_new_table(&init_net, cfg.fc_table);
  if (tb == NULL) {
   err = -ENOBUFS;
   goto errout;
@@ -671,9 +671,9 @@ static void fib_magic(int cmd, int type, __be32 dst, int dst_len, struct
in_ifad
  };

  if (type == RTN_UNICAST)
-  tb = fib_new_table(RT_TABLE_MAIN);
+  tb = fib_new_table(&init_net, RT_TABLE_MAIN);
  else
-  tb = fib_new_table(RT_TABLE_LOCAL);
+  tb = fib_new_table(&init_net, RT_TABLE_LOCAL);

  if (tb == NULL)
   return;
@@ -845,7 +845,7 @@ static void nl_fib_input(struct sk_buff *skb)
  }

  frn = (struct fib_result_nl *) NLMSG_DATA(nlh);
- tb = fib_get_table(frn->tb_id_in);
+ tb = fib_get_table(&init_net, frn->tb_id_in);

  nl_fib_lookup(frn, tb);

diff --git a/net/ipv4/fib_hash.c b/net/ipv4/fib_hash.c
index 11277f6..75c1bde 100644
--- a/net/ipv4/fib_hash.c
+++ b/net/ipv4/fib_hash.c
@@ -795,7 +795,7 @@ struct fib_iter_state {
 static struct fib_alias *fib_get_first(struct seq_file *seq)
 {
  struct fib_iter_state *iter = seq->private;
- struct fib_table *main_table = fib_get_table(RT_TABLE_MAIN);
+ struct fib_table *main_table = fib_get_table(&init_net, RT_TABLE_MAIN);
```

```
  struct fn_hash *table = (struct fn_hash *)main_table->tb_data;

  iter->bucket    = 0;
@@ -935,7 +935,7 @@ static void *fib_seq_start(struct seq_file *seq, loff_t *pos)
  void *v = NULL;

  read_lock(&fib_hash_lock);
- if (fib_get_table(RT_TABLE_MAIN))
+ if (fib_get_table(&init_net, RT_TABLE_MAIN))
    v = *pos ? fib_get_idx(seq, *pos - 1) : SEQ_START_TOKEN;
  return v;
 }
diff --git a/net/ipv4/fib_rules.c b/net/ipv4/fib_rules.c
index 1aae61c..49819fe 100644
--- a/net/ipv4/fib_rules.c
+++ b/net/ipv4/fib_rules.c
@@ -93,7 +93,7 @@ static int fib4_rule_action(struct fib_rule *rule, struct flowi *flp,
    goto errout;
  }

- if ((tbl = fib_get_table(rule->table)) == NULL)
+ if ((tbl = fib_get_table(&init_net, rule->table)) == NULL)
    goto errout;

  err = tbl->tb_lookup(tbl, flp, (struct fib_result *) arg->result);
@@ -109,7 +109,7 @@ void fib_select_default(const struct flowi *flp, struct fib_result *res)
  if (res->r && res->r->action == FR_ACT_TO_TBL &&
      FIB_RES_GW(*res) && FIB_RES_NH(*res).nh_scope == RT_SCOPE_LINK) {
    struct fib_table *tb;
-   if ((tb = fib_get_table(res->r->table)) != NULL)
+   if ((tb = fib_get_table(&init_net, res->r->table)) != NULL)
      tb->tb_select_default(tb, flp, res);
  }
 }
@@ -130,13 +130,13 @@ static int fib4_rule_match(struct fib_rule *rule, struct flowi *fl, int flags)
  return 1;
 }

-static struct fib_table *fib_empty_table(void)
+static struct fib_table *fib_empty_table(struct net *net)
 {
  u32 id;

  for (id = 1; id <= RT_TABLE_MAX; id++)
-   if (fib_get_table(id) == NULL)
-     return fib_new_table(id);
+   if (fib_get_table(net, id) == NULL)
+     return fib_new_table(net, id);
```

```
  return NULL;
 }

@@ -159,7 +159,7 @@ static int fib4_rule_configure(struct fib_rule *rule, struct sk_buff *skb,
    if (rule->action == FR_ACT_TO_TBL) {
     struct fib_table *table;

-    table = fib_empty_table();
+    table = fib_empty_table(&init_net);
     if (table == NULL) {
      err = -ENOBUFS;
      goto errout;
diff --git a/net/ipv4/fib_trie.c b/net/ipv4/fib_trie.c
index 43b6e94..9526758 100644
--- a/net/ipv4/fib_trie.c
+++ b/net/ipv4/fib_trie.c
@@ -2164,12 +2164,12 @@ static int fib_triestat_seq_show(struct seq_file *seq, void *v)
  struct fib_table *tb;

  trie_local = NULL;
- tb = fib_get_table(RT_TABLE_LOCAL);
+ tb = fib_get_table(&init_net, RT_TABLE_LOCAL);
  if (tb)
   trie_local = (struct trie *) tb->tb_data;

  trie_main = NULL;
- tb = fib_get_table(RT_TABLE_MAIN);
+ tb = fib_get_table(&init_net, RT_TABLE_MAIN);
  if (tb)
   trie_main = (struct trie *) tb->tb_data;

@@ -2236,12 +2236,12 @@ static void *fib_trie_seq_start(struct seq_file *seq, loff_t *pos)
  struct fib_table *tb;

  if (!iter->trie_local) {
-  tb = fib_get_table(RT_TABLE_LOCAL);
+  tb = fib_get_table(&init_net, RT_TABLE_LOCAL);
   if (tb)
    iter->trie_local = (struct trie *) tb->tb_data;
  }
  if (!iter->trie_main) {
-  tb = fib_get_table(RT_TABLE_MAIN);
+  tb = fib_get_table(&init_net, RT_TABLE_MAIN);
   if (tb)
    iter->trie_main = (struct trie *) tb->tb_data;
  }
--
1.5.3.rc5
```