
Subject: [PATCH 12/12] L2 network namespace (v3): L3 network namespace intro
Posted by [Mishin Dmitry](#) on Wed, 17 Jan 2007 16:18:25 GMT
[View Forum Message](#) <> [Reply to Message](#)

Introduce two kind of network namespaces - level 2 and level 3. First one is namespace with full set of networking objects, while second one - socket-level with restricted set.

Signed-off-by: Dmitry Mishin <dim@openvz.org>

```
---
include/linux/net_namespace.h | 3 +++
net/core/net_namespace.c      | 40 ++++++-----
2 files changed, 31 insertions(+), 12 deletions(-)

--- linux-2.6.20-rc4-mm1.net_ns.orig/include/linux/net_namespace.h
+++ linux-2.6.20-rc4-mm1.net_ns/include/linux/net_namespace.h
@@ -24,6 +24,9 @@ struct net_namespace {
    int fib4_trie_last_dflt;
#endif
    unsigned int hash;
+#define NET_NS_LEVEL2 1
+#define NET_NS_LEVEL3 2
+ unsigned int level;
};

extern struct net_namespace init_net_ns;
--- linux-2.6.20-rc4-mm1.net_ns.orig/net/core/net_namespace.c
+++ linux-2.6.20-rc4-mm1.net_ns/net/core/net_namespace.c
@@ -30,13 +30,19 @@ EXPORT_PER_CPU_SYMBOL_GPL(exec_net_ns);

/*
 * Clone a new ns copying an original net ns, setting refcount to 1
+ * @level: level of namespace to create
 * @old_ns: namespace to clone
- * Return NULL on error (failure to kcalloc), new ns otherwise
+ * Return ERR_PTR on error, new ns otherwise
 */
-static struct net_namespace *clone_net_ns(struct net_namespace *old_ns)
+static struct net_namespace *clone_net_ns(unsigned int level,
+ struct net_namespace *old_ns)
{
    struct net_namespace *ns;

+ /* level 3 namespaces are incomplete in order to have childs */
+ if (current_net_ns->level == NET_NS_LEVEL3)
+ return ERR_PTR(-EPERM);
+
+

```

```

ns = kzalloc(sizeof(struct net_namespace), GFP_KERNEL);
if (!ns)
    return NULL;
@@ -48,20 +54,25 @@ static struct net_namespace *clone_net_n

if ((push_net_ns(ns)) != old_ns)
    BUG();
+ if (level == NET_NS_LEVEL2) {
#ifdef CONFIG_IP_MULTIPLE_TABLES
- INIT_LIST_HEAD(&ns->fib_rules_ops_list);
+ INIT_LIST_HEAD(&ns->fib_rules_ops_list);
#endif
- if (ip_fib_struct_init())
- goto out_fib4;
+ if (ip_fib_struct_init())
+ goto out_fib4;
+ }
+ ns->level = level;
if (loopback_init())
    goto out_loopback;
pop_net_ns(old_ns);
- printk(KERN_DEBUG "NET_NS: created new netcontext %p for %s "
- "(pid=%d)\n", ns, current->comm, current->tgid);
+ printk(KERN_DEBUG "NET_NS: created new netcontext %p, level %u, "
+ "for %s (pid=%d)\n", ns, (ns->level == NET_NS_LEVEL2) ?
+ 2 : 3, current->comm, current->tgid);
return ns;

out_loopback:
- ip_fib_struct_cleanup(ns);
+ if (level == NET_NS_LEVEL2)
+ ip_fib_struct_cleanup(ns);
out_fib4:
    pop_net_ns(old_ns);
    BUG_ON(atomic_read(&ns->kref.refcount) != 1);
@@ -75,13 +86,17 @@ out_fib4:
int unshare_net_ns(unsigned long unshare_flags,
    struct net_namespace **new_net)
{
+ unsigned int level;
+
if (unshare_flags & (CLONE_NEWNET2|CLONE_NEWNET3)) {
    if (!capable(CAP_SYS_ADMIN))
        return -EPERM;

- *new_net = clone_net_ns(current->nsproxy->net_ns);
- if (!*new_net)
- return -ENOMEM;

```

```

+ level = (unshare_flags & CLONE_NEWNET2) ? NET_NS_LEVEL2 :
+   NET_NS_LEVEL3;
+ *new_net = clone_net_ns(level, current->nsproxy->net_ns);
+ if (IS_ERR(*new_net))
+   return PTR_ERR(*new_net);
+ }

return 0;
@@ -110,7 +125,8 @@ void free_net_ns(struct kref *kref)
    ns, atomic_read(&ns->kref.refcount));
return;
}
- ip_fib_struct_cleanup(ns);
+ if (ns->level == NET_NS_LEVEL2)
+ ip_fib_struct_cleanup(ns);
    printk(KERN_DEBUG "NET_NS: net namespace %p destroyed\n", ns);
    kfree(ns);
}

```

Containers mailing list

Containers@lists.osdl.org

<https://lists.osdl.org/mailman/listinfo/containers>
