
Subject: [PATCH 1/3] [RAW]: Family check in the /proc/net/raw[6] is extra.
Posted by [den](#) on Thu, 31 Jan 2008 11:34:52 GMT

[View Forum Message](#) <> [Reply to Message](#)

Different hashtables are used for IPv6 and IPv4 raw sockets, so no need to check the socket family in the iterator over hashtables. Clean this out.

Signed-off-by: Denis V. Lunev <den@openvz.org>

```
include/net/raw.h | 4 +---
net/ipv4/raw.c    | 12 +++++-----
net/ipv6/raw.c    | 2 +-
3 files changed, 6 insertions(+), 12 deletions(-)
```

```
diff --git a/include/net/raw.h b/include/net/raw.h
```

```
index cca81d8..c7ea7a2 100644
```

```
--- a/include/net/raw.h
```

```
+++ b/include/net/raw.h
```

```
@@ -41,7 +41,6 @@ extern void raw_proc_exit(void);
```

```
struct raw_iter_state {
    struct seq_net_private p;
    int bucket;
```

```
- unsigned short family;
    struct raw_hashinfo *h;
};
```

```
@@ -49,8 +48,7 @@ struct raw_iter_state {
```

```
void *raw_seq_start(struct seq_file *seq, loff_t *pos);
```

```
void *raw_seq_next(struct seq_file *seq, void *v, loff_t *pos);
```

```
void raw_seq_stop(struct seq_file *seq, void *v);
```

```
-int raw_seq_open(struct inode *ino, struct file *file, struct raw_hashinfo *h,
```

```
- unsigned short family);
```

```
+int raw_seq_open(struct inode *ino, struct file *file, struct raw_hashinfo *h);
```

```
#endif
```

```
diff --git a/net/ipv4/raw.c b/net/ipv4/raw.c
```

```
index f863c3d..507cbfe 100644
```

```
--- a/net/ipv4/raw.c
```

```
+++ b/net/ipv4/raw.c
```

```
@@ -862,8 +862,7 @@ static struct sock *raw_get_first(struct seq_file *seq)
    struct hlist_node *node;
```

```
    sk_for_each(sk, node, &state->h->ht[state->bucket])
```

```
- if (sk->sk_net == state->p.net &&
```

```
-    sk->sk_family == state->family)
```

```
+ if (sk->sk_net == state->p.net)
```

```
    goto found;
```

```

}
sk = NULL;
@@ -879,8 +878,7 @@ static struct sock *raw_get_next(struct seq_file *seq, struct sock *sk)
    sk = sk_next(sk);
try_again:
;
- } while (sk && sk->sk_net != state->p.net &&
- sk->sk_family != state->family);
+ } while (sk && sk->sk_net != state->p.net);

    if (!sk && ++state->bucket < RAW_HTABLE_SIZE) {
        sk = sk_head(&state->h->ht[state->bucket]);
@@ -974,8 +972,7 @@ static const struct seq_operations raw_seq_ops = {
    .show = raw_seq_show,
};

-int raw_seq_open(struct inode *ino, struct file *file, struct raw_hashinfo *h,
- unsigned short family)
+int raw_seq_open(struct inode *ino, struct file *file, struct raw_hashinfo *h)
{
    int err;
    struct raw_iter_state *i;
@@ -987,14 +984,13 @@ int raw_seq_open(struct inode *ino, struct file *file, struct raw_hashinfo
*h,

    i = raw_seq_private((struct seq_file *)file->private_data);
    i->h = h;
- i->family = family;
    return 0;
}
EXPORT_SYMBOL_GPL(raw_seq_open);

static int raw_v4_seq_open(struct inode *inode, struct file *file)
{
- return raw_seq_open(inode, file, &raw_v4_hashinfo, PF_INET);
+ return raw_seq_open(inode, file, &raw_v4_hashinfo);
}

static const struct file_operations raw_seq_fops = {
diff --git a/net/ipv6/raw.c b/net/ipv6/raw.c
index d61c63d..a2cf499 100644
--- a/net/ipv6/raw.c
+++ b/net/ipv6/raw.c
@@ -1262,7 +1262,7 @@ static const struct seq_operations raw6_seq_ops = {

static int raw6_seq_open(struct inode *inode, struct file *file)
{
- return raw_seq_open(inode, file, &raw_v6_hashinfo, PF_INET6);

```

```
+ return raw_seq_open(inode, file, &raw_v6_hashinfo);  
}
```

```
static const struct file_operations raw6_seq_fops = {
```

```
--
```

```
1.5.3.rc5
```
