Subject: [PATCH 2/2] Eliminate unused argument from sk_stream_alloc_pskb
Posted by Pavel Emelianov on Mon, 26 Nov 2007 17:17:27 GMT
View Forum Message <> Reply to Message

The 3rd argument is always zero (according to grep :) Eliminate
it and merge the function with sk_stream_alloc_skb.

This saves 44 more bytes, and together with the previous patch
we have:

```
add/remove: 1/0 grow/shrink: 0/8 up/down: 183/-751 (-568)
function                     old     new   delta
sk_stream_alloc_skb            -     183    +183
ip_rt_init                   529     525      -4
arp_ignore                   112     107      -5
__inet_lookup_listener       284     274     -10
tcp_sendmsg                 2583    2481    -102
tcp_sendpage                1449    1300    -149
tso_fragment                 417     258    -159
tcp_fragment                1149     988    -161
__tcp_push_pending_frames   1998    1837    -161
```

Question: is this 2.6.24 material (good space saving) or should I
rework this against 2.6.25 (it applies with fuzzes, but seems to
compile)?

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

```
diff --git a/include/net/sock.h b/include/net/sock.h
index 492dc4a..a469ed8 100644
--- a/include/net/sock.h
+++ b/include/net/sock.h
@@ -1230,15 +1230,7 @@ static inline void sk_stream_moderate_sndbuf(struct sock *sk)
  }
 }

-struct sk_buff *sk_stream_alloc_pskb(struct sock *sk,
-  int size, int mem, gfp_t gfp);
-
-static inline struct sk_buff *sk_stream_alloc_skb(struct sock *sk,
-       int size,
-       gfp_t gfp)
-{
- return sk_stream_alloc_pskb(sk, size, 0, gfp);
-}
+struct sk_buff *sk_stream_alloc_skb(struct sock *sk, int size, gfp_t gfp);
```

```
 static inline struct page *sk_stream_alloc_page(struct sock *sk)
 {
diff --git a/net/ipv4/tcp.c b/net/ipv4/tcp.c
index 0dfda20..1965c37 100644
--- a/net/ipv4/tcp.c
+++ b/net/ipv4/tcp.c
@@ -501,8 +501,7 @@ static inline void tcp_push(struct sock *sk, int flags, int mss_now,
  }
 }

-struct sk_buff *sk_stream_alloc_pskb(struct sock *sk,
-  int size, int mem, gfp_t gfp)
+struct sk_buff *sk_stream_alloc_skb(struct sock *sk, int size, gfp_t gfp)
 {
  struct sk_buff *skb;

@@ -511,7 +510,6 @@ struct sk_buff *sk_stream_alloc_pskb(struct sock *sk,

  skb = alloc_skb_fclone(size + sk->sk_prot->max_header, gfp);
  if (skb) {
-  skb->truesize += mem;
   if (sk_stream_wmem_schedule(sk, skb->truesize)) {
    /*
     * Make sure that we have exactly size bytes
@@ -564,8 +562,7 @@ new_segment:
    if (!sk_stream_memory_free(sk))
     goto wait_for_sndbuf;

-   skb = sk_stream_alloc_pskb(sk, 0, 0,
-         sk->sk_allocation);
+   skb = sk_stream_alloc_skb(sk, 0, sk->sk_allocation);
    if (!skb)
     goto wait_for_memory;

@@ -745,8 +742,8 @@ new_segment:
     if (!sk_stream_memory_free(sk))
      goto wait_for_sndbuf;

-    skb = sk_stream_alloc_pskb(sk, select_size(sk),
-          0, sk->sk_allocation);
+    skb = sk_stream_alloc_skb(sk, select_size(sk),
+      sk->sk_allocation);
     if (!skb)
      goto wait_for_memory;

diff --git a/net/ipv4/tcp_output.c b/net/ipv4/tcp_output.c
index e5130a7..132e16b 100644
```

```
--- a/net/ipv4/tcp_output.c
+++ b/net/ipv4/tcp_output.c
@@ -1183,7 +1183,7 @@ static int tso_fragment(struct sock *sk, struct sk_buff *skb, unsigned int
len,
  if (skb->len != skb->data_len)
   return tcp_fragment(sk, skb, len, mss_now);

- buff = sk_stream_alloc_pskb(sk, 0, 0, GFP_ATOMIC);
+ buff = sk_stream_alloc_skb(sk, 0, GFP_ATOMIC);
  if (unlikely(buff == NULL))
   return -ENOMEM;

--
1.5.3.4
```