Subject: [PATCH 8/8] Forget the zero_it argument of sk_alloc()
Posted by Pavel Emelianov on Wed, 31 Oct 2007 12:57:18 GMT
View Forum Message <> Reply to Message

Finally, the zero_it argument can be completely removed from
the callers and from the function prototype.

Besides, fix the checkpatch.pl warnings about using the
assignments inside if-s.

This patch is rather big, and it is a part of the previous one.
I splitted it wishing to make the patches more readable. Hope
this particular split helped.

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

```
diff --git a/drivers/net/pppoe.c b/drivers/net/pppoe.c
index 8936ed3..a005d8f 100644
--- a/drivers/net/pppoe.c
+++ b/drivers/net/pppoe.c
@@ -491,7 +491,7 @@ static int pppoe_create(struct net *net, struct socket *sock)
  int error = -ENOMEM;
  struct sock *sk;

- sk = sk_alloc(net, PF_PPPOX, GFP_KERNEL, &pppoe_sk_proto, 1);
+ sk = sk_alloc(net, PF_PPPOX, GFP_KERNEL, &pppoe_sk_proto);
  if (!sk)
   goto out;

diff --git a/drivers/net/pppol2tp.c b/drivers/net/pppol2tp.c
index 921d4ef..f8904fd 100644
--- a/drivers/net/pppol2tp.c
+++ b/drivers/net/pppol2tp.c
@@ -1416,7 +1416,7 @@ static int pppol2tp_create(struct net *net, struct socket *sock)
  int error = -ENOMEM;
  struct sock *sk;

- sk = sk_alloc(net, PF_PPPOX, GFP_KERNEL, &pppol2tp_sk_proto, 1);
+ sk = sk_alloc(net, PF_PPPOX, GFP_KERNEL, &pppol2tp_sk_proto);
  if (!sk)
   goto out;

diff --git a/include/net/sock.h b/include/net/sock.h
index ecad7b4..20de3fa 100644
--- a/include/net/sock.h
+++ b/include/net/sock.h
```

```
@@ -779,7 +779,7 @@ extern void FASTCALL(release_sock(struct sock *sk));

 extern struct sock  *sk_alloc(struct net *net, int family,
         gfp_t priority,
-        struct proto *prot, int zero_it);
+        struct proto *prot);
 extern void    sk_free(struct sock *sk);
 extern struct sock  *sk_clone(const struct sock *sk,
         const gfp_t priority);
diff --git a/net/appletalk/ddp.c b/net/appletalk/ddp.c
index 7c0b515..e0d37d6 100644
--- a/net/appletalk/ddp.c
+++ b/net/appletalk/ddp.c
@@ -1044,7 +1044,7 @@ static int atalk_create(struct net *net, struct socket *sock, int protocol)
  if (sock->type != SOCK_RAW && sock->type != SOCK_DGRAM)
   goto out;
  rc = -ENOMEM;
- sk = sk_alloc(net, PF_APPLETALK, GFP_KERNEL, &ddp_proto, 1);
+ sk = sk_alloc(net, PF_APPLETALK, GFP_KERNEL, &ddp_proto);
  if (!sk)
   goto out;
  rc = 0;
diff --git a/net/atm/common.c b/net/atm/common.c
index e166d9e..eba09a0 100644
--- a/net/atm/common.c
+++ b/net/atm/common.c
@@ -133,7 +133,7 @@ int vcc_create(struct net *net, struct socket *sock, int protocol, int family)
  sock->sk = NULL;
  if (sock->type == SOCK_STREAM)
   return -EINVAL;
- sk = sk_alloc(net, family, GFP_KERNEL, &vcc_proto, 1);
+ sk = sk_alloc(net, family, GFP_KERNEL, &vcc_proto);
  if (!sk)
   return -ENOMEM;
  sock_init_data(sock, sk);
diff --git a/net/ax25/af_ax25.c b/net/ax25/af_ax25.c
index 993e5c7..8378afd 100644
--- a/net/ax25/af_ax25.c
+++ b/net/ax25/af_ax25.c
@@ -836,7 +836,8 @@ static int ax25_create(struct net *net, struct socket *sock, int protocol)
   return -ESOCKTNOSUPPORT;
  }

- if ((sk = sk_alloc(net, PF_AX25, GFP_ATOMIC, &ax25_proto, 1)) == NULL)
+ sk = sk_alloc(net, PF_AX25, GFP_ATOMIC, &ax25_proto);
+ if (sk == NULL)
   return -ENOMEM;
```

```
  ax25 = sk->sk_protinfo = ax25_create_cb();
@@ -861,7 +862,8 @@ struct sock *ax25_make_new(struct sock *osk, struct ax25_dev
*ax25_dev)
  struct sock *sk;
  ax25_cb *ax25, *oax25;

- if ((sk = sk_alloc(osk->sk_net, PF_AX25, GFP_ATOMIC, osk->sk_prot, 1)) == NULL)
+ sk = sk_alloc(osk->sk_net, PF_AX25, GFP_ATOMIC, osk->sk_prot);
+ if (sk == NULL)
   return NULL;

  if ((ax25 = ax25_create_cb()) == NULL) {
diff --git a/net/bluetooth/bnep/sock.c b/net/bluetooth/bnep/sock.c
index f718965..9ebd3c6 100644
--- a/net/bluetooth/bnep/sock.c
+++ b/net/bluetooth/bnep/sock.c
@@ -213,7 +213,7 @@ static int bnep_sock_create(struct net *net, struct socket *sock, int
protocol)
  if (sock->type != SOCK_RAW)
   return -ESOCKTNOSUPPORT;

- sk = sk_alloc(net, PF_BLUETOOTH, GFP_ATOMIC, &bnep_proto, 1);
+ sk = sk_alloc(net, PF_BLUETOOTH, GFP_ATOMIC, &bnep_proto);
  if (!sk)
   return -ENOMEM;

diff --git a/net/bluetooth/cmtp/sock.c b/net/bluetooth/cmtp/sock.c
index cf700c2..783edab 100644
--- a/net/bluetooth/cmtp/sock.c
+++ b/net/bluetooth/cmtp/sock.c
@@ -204,7 +204,7 @@ static int cmtp_sock_create(struct net *net, struct socket *sock, int
protocol)
  if (sock->type != SOCK_RAW)
   return -ESOCKTNOSUPPORT;

- sk = sk_alloc(net, PF_BLUETOOTH, GFP_ATOMIC, &cmtp_proto, 1);
+ sk = sk_alloc(net, PF_BLUETOOTH, GFP_ATOMIC, &cmtp_proto);
  if (!sk)
   return -ENOMEM;

diff --git a/net/bluetooth/hci_sock.c b/net/bluetooth/hci_sock.c
index 8825102..1499132 100644
--- a/net/bluetooth/hci_sock.c
+++ b/net/bluetooth/hci_sock.c
@@ -645,7 +645,7 @@ static int hci_sock_create(struct net *net, struct socket *sock, int protocol)

  sock->ops = &hci_sock_ops;
```

```
- sk = sk_alloc(net, PF_BLUETOOTH, GFP_ATOMIC, &hci_sk_proto, 1);
+ sk = sk_alloc(net, PF_BLUETOOTH, GFP_ATOMIC, &hci_sk_proto);
  if (!sk)
   return -ENOMEM;

diff --git a/net/bluetooth/hidp/sock.c b/net/bluetooth/hidp/sock.c
index 1de2b6f..3292b95 100644
--- a/net/bluetooth/hidp/sock.c
+++ b/net/bluetooth/hidp/sock.c
@@ -255,7 +255,7 @@ static int hidp_sock_create(struct net *net, struct socket *sock, int
protocol)
  if (sock->type != SOCK_RAW)
   return -ESOCKTNOSUPPORT;

- sk = sk_alloc(net, PF_BLUETOOTH, GFP_ATOMIC, &hidp_proto, 1);
+ sk = sk_alloc(net, PF_BLUETOOTH, GFP_ATOMIC, &hidp_proto);
  if (!sk)
   return -ENOMEM;

diff --git a/net/bluetooth/l2cap.c b/net/bluetooth/l2cap.c
index 6fbbae7..477e052 100644
--- a/net/bluetooth/l2cap.c
+++ b/net/bluetooth/l2cap.c
@@ -607,7 +607,7 @@ static struct sock *l2cap_sock_alloc(struct net *net, struct socket *sock,
int p
 {
  struct sock *sk;

- sk = sk_alloc(net, PF_BLUETOOTH, prio, &l2cap_proto, 1);
+ sk = sk_alloc(net, PF_BLUETOOTH, prio, &l2cap_proto);
  if (!sk)
   return NULL;

diff --git a/net/bluetooth/rfcomm/sock.c b/net/bluetooth/rfcomm/sock.c
index 266b697..c46d510 100644
--- a/net/bluetooth/rfcomm/sock.c
+++ b/net/bluetooth/rfcomm/sock.c
@@ -287,7 +287,7 @@ static struct sock *rfcomm_sock_alloc(struct net *net, struct socket *sock,
int
  struct rfcomm_dlc *d;
  struct sock *sk;

- sk = sk_alloc(net, PF_BLUETOOTH, prio, &rfcomm_proto, 1);
+ sk = sk_alloc(net, PF_BLUETOOTH, prio, &rfcomm_proto);
  if (!sk)
   return NULL;

diff --git a/net/bluetooth/sco.c b/net/bluetooth/sco.c
```

```
index 82d0dfd..93ad1aa 100644
--- a/net/bluetooth/sco.c
+++ b/net/bluetooth/sco.c
@@ -421,7 +421,7 @@ static struct sock *sco_sock_alloc(struct net *net, struct socket *sock, int
pro
 {
  struct sock *sk;

- sk = sk_alloc(net, PF_BLUETOOTH, prio, &sco_proto, 1);
+ sk = sk_alloc(net, PF_BLUETOOTH, prio, &sco_proto);
  if (!sk)
   return NULL;

diff --git a/net/core/sock.c b/net/core/sock.c
index 6046fc6..12ad206 100644
--- a/net/core/sock.c
+++ b/net/core/sock.c
@@ -927,7 +927,7 @@ static void sk_prot_free(struct proto *prot, struct sock *sk)
  * @zero_it: if we should zero the newly allocated sock
  */
 struct sock *sk_alloc(struct net *net, int family, gfp_t priority,
-        struct proto *prot, int zero_it)
+        struct proto *prot)
 {
  struct sock *sk;

diff --git a/net/decnet/af_decnet.c b/net/decnet/af_decnet.c
index aabe98d..57d5749 100644
--- a/net/decnet/af_decnet.c
+++ b/net/decnet/af_decnet.c
@@ -474,7 +474,7 @@ static struct proto dn_proto = {
 static struct sock *dn_alloc_sock(struct net *net, struct socket *sock, gfp_t gfp)
 {
  struct dn_scp *scp;
- struct sock *sk = sk_alloc(net, PF_DECnet, gfp, &dn_proto, 1);
+ struct sock *sk = sk_alloc(net, PF_DECnet, gfp, &dn_proto);

  if  (!sk)
   goto out;
diff --git a/net/econet/af_econet.c b/net/econet/af_econet.c
index 9cae16b..f70df07 100644
--- a/net/econet/af_econet.c
+++ b/net/econet/af_econet.c
@@ -624,7 +624,7 @@ static int econet_create(struct net *net, struct socket *sock, int protocol)
  sock->state = SS_UNCONNECTED;

  err = -ENOBUFS;
- sk = sk_alloc(net, PF_ECONET, GFP_KERNEL, &econet_proto, 1);
```

```
+ sk = sk_alloc(net, PF_ECONET, GFP_KERNEL, &econet_proto);
  if (sk == NULL)
   goto out;

diff --git a/net/ipv4/af_inet.c b/net/ipv4/af_inet.c
index 621b128..d2f22e7 100644
--- a/net/ipv4/af_inet.c
+++ b/net/ipv4/af_inet.c
@@ -323,7 +323,7 @@ lookup_protocol:
  BUG_TRAP(answer_prot->slab != NULL);

  err = -ENOBUFS;
- sk = sk_alloc(net, PF_INET, GFP_KERNEL, answer_prot, 1);
+ sk = sk_alloc(net, PF_INET, GFP_KERNEL, answer_prot);
  if (sk == NULL)
   goto out;

diff --git a/net/ipv6/af_inet6.c b/net/ipv6/af_inet6.c
index 1b1caf3..ecbd388 100644
--- a/net/ipv6/af_inet6.c
+++ b/net/ipv6/af_inet6.c
@@ -162,7 +162,7 @@ lookup_protocol:
  BUG_TRAP(answer_prot->slab != NULL);

  err = -ENOBUFS;
- sk = sk_alloc(net, PF_INET6, GFP_KERNEL, answer_prot, 1);
+ sk = sk_alloc(net, PF_INET6, GFP_KERNEL, answer_prot);
  if (sk == NULL)
   goto out;

diff --git a/net/ipx/af_ipx.c b/net/ipx/af_ipx.c
index 29b063d..a195a66 100644
--- a/net/ipx/af_ipx.c
+++ b/net/ipx/af_ipx.c
@@ -1381,7 +1381,7 @@ static int ipx_create(struct net *net, struct socket *sock, int protocol)
   goto out;

  rc = -ENOMEM;
- sk = sk_alloc(net, PF_IPX, GFP_KERNEL, &ipx_proto, 1);
+ sk = sk_alloc(net, PF_IPX, GFP_KERNEL, &ipx_proto);
  if (!sk)
   goto out;
 #ifdef IPX_REFCNT_DEBUG
diff --git a/net/irda/af_irda.c b/net/irda/af_irda.c
index 0328ae2..48ce59a 100644
--- a/net/irda/af_irda.c
+++ b/net/irda/af_irda.c
@@ -1078,7 +1078,7 @@ static int irda_create(struct net *net, struct socket *sock, int protocol)
```

```
  }

  /* Allocate networking socket */
- sk = sk_alloc(net, PF_IRDA, GFP_ATOMIC, &irda_proto, 1);
+ sk = sk_alloc(net, PF_IRDA, GFP_ATOMIC, &irda_proto);
  if (sk == NULL)
   return -ENOMEM;

diff --git a/net/iucv/af_iucv.c b/net/iucv/af_iucv.c
index 43e01c8..aef6645 100644
--- a/net/iucv/af_iucv.c
+++ b/net/iucv/af_iucv.c
@@ -216,7 +216,7 @@ static struct sock *iucv_sock_alloc(struct socket *sock, int proto, gfp_t
prio)
 {
  struct sock *sk;

- sk = sk_alloc(&init_net, PF_IUCV, prio, &iucv_proto, 1);
+ sk = sk_alloc(&init_net, PF_IUCV, prio, &iucv_proto);
  if (!sk)
   return NULL;

diff --git a/net/key/af_key.c b/net/key/af_key.c
index 266f112..10c89d4 100644
--- a/net/key/af_key.c
+++ b/net/key/af_key.c
@@ -152,7 +152,7 @@ static int pfkey_create(struct net *net, struct socket *sock, int protocol)
   return -EPROTONOSUPPORT;

  err = -ENOMEM;
- sk = sk_alloc(net, PF_KEY, GFP_KERNEL, &key_proto, 1);
+ sk = sk_alloc(net, PF_KEY, GFP_KERNEL, &key_proto);
  if (sk == NULL)
   goto out;

diff --git a/net/llc/llc_conn.c b/net/llc/llc_conn.c
index 8ebc276..5c0b484 100644
--- a/net/llc/llc_conn.c
+++ b/net/llc/llc_conn.c
@@ -869,7 +869,7 @@ static void llc_sk_init(struct sock* sk)
  */
 struct sock *llc_sk_alloc(struct net *net, int family, gfp_t priority, struct proto *prot)
 {
- struct sock *sk = sk_alloc(net, family, priority, prot, 1);
+ struct sock *sk = sk_alloc(net, family, priority, prot);

  if (!sk)
   goto out;
```

```
diff --git a/net/netlink/af_netlink.c b/net/netlink/af_netlink.c
index 4f994c0..2601712 100644
--- a/net/netlink/af_netlink.c
+++ b/net/netlink/af_netlink.c
@@ -396,7 +396,7 @@ static int __netlink_create(struct net *net, struct socket *sock,

  sock->ops = &netlink_ops;

- sk = sk_alloc(net, PF_NETLINK, GFP_KERNEL, &netlink_proto, 1);
+ sk = sk_alloc(net, PF_NETLINK, GFP_KERNEL, &netlink_proto);
  if (!sk)
   return -ENOMEM;

diff --git a/net/netrom/af_netrom.c b/net/netrom/af_netrom.c
index 3a4d479..972250c 100644
--- a/net/netrom/af_netrom.c
+++ b/net/netrom/af_netrom.c
@@ -423,7 +423,8 @@ static int nr_create(struct net *net, struct socket *sock, int protocol)
  if (sock->type != SOCK_SEQPACKET || protocol != 0)
   return -ESOCKTNOSUPPORT;

- if ((sk = sk_alloc(net, PF_NETROM, GFP_ATOMIC, &nr_proto, 1)) == NULL)
+ sk = sk_alloc(net, PF_NETROM, GFP_ATOMIC, &nr_proto);
+ if (sk  == NULL)
   return -ENOMEM;

  nr = nr_sk(sk);
@@ -465,7 +466,8 @@ static struct sock *nr_make_new(struct sock *osk)
  if (osk->sk_type != SOCK_SEQPACKET)
   return NULL;

- if ((sk = sk_alloc(osk->sk_net, PF_NETROM, GFP_ATOMIC, osk->sk_prot, 1)) == NULL)
+ sk = sk_alloc(osk->sk_net, PF_NETROM, GFP_ATOMIC, osk->sk_prot);
+ if (sk == NULL)
   return NULL;

  nr = nr_sk(sk);
diff --git a/net/packet/af_packet.c b/net/packet/af_packet.c
index d093650..4cb2dfb 100644
--- a/net/packet/af_packet.c
+++ b/net/packet/af_packet.c
@@ -995,7 +995,7 @@ static int packet_create(struct net *net, struct socket *sock, int protocol)
  sock->state = SS_UNCONNECTED;

  err = -ENOBUFS;
- sk = sk_alloc(net, PF_PACKET, GFP_KERNEL, &packet_proto, 1);
+ sk = sk_alloc(net, PF_PACKET, GFP_KERNEL, &packet_proto);
  if (sk == NULL)
```

```
   goto out;

diff --git a/net/rose/af_rose.c b/net/rose/af_rose.c
index 509defe..ed2d65c 100644
--- a/net/rose/af_rose.c
+++ b/net/rose/af_rose.c
@@ -513,7 +513,8 @@ static int rose_create(struct net *net, struct socket *sock, int protocol)
  if (sock->type != SOCK_SEQPACKET || protocol != 0)
   return -ESOCKTNOSUPPORT;

- if ((sk = sk_alloc(net, PF_ROSE, GFP_ATOMIC, &rose_proto, 1)) == NULL)
+ sk = sk_alloc(net, PF_ROSE, GFP_ATOMIC, &rose_proto);
+ if (sk == NULL)
   return -ENOMEM;

  rose = rose_sk(sk);
@@ -551,7 +552,8 @@ static struct sock *rose_make_new(struct sock *osk)
  if (osk->sk_type != SOCK_SEQPACKET)
   return NULL;

- if ((sk = sk_alloc(osk->sk_net, PF_ROSE, GFP_ATOMIC, &rose_proto, 1)) == NULL)
+ sk = sk_alloc(osk->sk_net, PF_ROSE, GFP_ATOMIC, &rose_proto);
+ if (sk == NULL)
   return NULL;

  rose = rose_sk(sk);
diff --git a/net/rxrpc/af_rxrpc.c b/net/rxrpc/af_rxrpc.c
index c680017..d638945 100644
--- a/net/rxrpc/af_rxrpc.c
+++ b/net/rxrpc/af_rxrpc.c
@@ -627,7 +627,7 @@ static int rxrpc_create(struct net *net, struct socket *sock, int protocol)
  sock->ops = &rxrpc_rpc_ops;
  sock->state = SS_UNCONNECTED;

- sk = sk_alloc(net, PF_RXRPC, GFP_KERNEL, &rxrpc_proto, 1);
+ sk = sk_alloc(net, PF_RXRPC, GFP_KERNEL, &rxrpc_proto);
  if (!sk)
   return -ENOMEM;

diff --git a/net/sctp/ipv6.c b/net/sctp/ipv6.c
index eb4deaf..7f31ff6 100644
--- a/net/sctp/ipv6.c
+++ b/net/sctp/ipv6.c
@@ -631,7 +631,7 @@ static struct sock *sctp_v6_create_accept_sk(struct sock *sk,
  struct ipv6_pinfo *newnp, *np = inet6_sk(sk);
  struct sctp6_sock *newsctp6sk;

- newsk = sk_alloc(sk->sk_net, PF_INET6, GFP_KERNEL, sk->sk_prot, 1);
```

```
+ newsk = sk_alloc(sk->sk_net, PF_INET6, GFP_KERNEL, sk->sk_prot);
  if (!newsk)
   goto out;

diff --git a/net/sctp/protocol.c b/net/sctp/protocol.c
index f5cd96f..40c1a47 100644
--- a/net/sctp/protocol.c
+++ b/net/sctp/protocol.c
@@ -552,7 +552,8 @@ static struct sock *sctp_v4_create_accept_sk(struct sock *sk,
 {
  struct inet_sock *inet = inet_sk(sk);
  struct inet_sock *newinet;
- struct sock *newsk = sk_alloc(sk->sk_net, PF_INET, GFP_KERNEL, sk->sk_prot, 1);
+ struct sock *newsk = sk_alloc(sk->sk_net, PF_INET, GFP_KERNEL,
+   sk->sk_prot);

  if (!newsk)
   goto out;
diff --git a/net/tipc/socket.c b/net/tipc/socket.c
index e36b4b5..6b79226 100644
--- a/net/tipc/socket.c
+++ b/net/tipc/socket.c
@@ -201,7 +201,7 @@ static int tipc_create(struct net *net, struct socket *sock, int protocol)
   return -EPROTOTYPE;
  }

- sk = sk_alloc(net, AF_TIPC, GFP_KERNEL, &tipc_proto, 1);
+ sk = sk_alloc(net, AF_TIPC, GFP_KERNEL, &tipc_proto);
  if (!sk) {
   tipc_deleteport(ref);
   return -ENOMEM;
diff --git a/net/unix/af_unix.c b/net/unix/af_unix.c
index 9163ec5..515e7a6 100644
--- a/net/unix/af_unix.c
+++ b/net/unix/af_unix.c
@@ -602,7 +602,7 @@ static struct sock * unix_create1(struct net *net, struct socket *sock)
  if (atomic_read(&unix_nr_socks) >= 2*get_max_files())
   goto out;

- sk = sk_alloc(net, PF_UNIX, GFP_KERNEL, &unix_proto, 1);
+ sk = sk_alloc(net, PF_UNIX, GFP_KERNEL, &unix_proto);
  if (!sk)
   goto out;

diff --git a/net/x25/af_x25.c b/net/x25/af_x25.c
index fc416f9..92cfe8e 100644
--- a/net/x25/af_x25.c
+++ b/net/x25/af_x25.c
```

```
@@ -472,7 +472,7 @@ static struct proto x25_proto = {
 static struct sock *x25_alloc_socket(struct net *net)
 {
  struct x25_sock *x25;
- struct sock *sk = sk_alloc(net, AF_X25, GFP_ATOMIC, &x25_proto, 1);
+ struct sock *sk = sk_alloc(net, AF_X25, GFP_ATOMIC, &x25_proto);

  if (!sk)
   goto out;
```