

---

Subject: [PATCH] {get|set}sockopt compatibility layer  
Posted by [Mishin Dmitry](#) on Fri, 10 Mar 2006 11:21:10 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

This patch extends {get|set}sockopt compatibility layer in order to move protocol specific parts to their place and avoid huge universal net/compat.c file in the future.

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

--  
Thanks,  
Dmitry.

```
--- ./include/linux/net.h.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./include/linux/net.h 2006-03-10 12:24:11.000000000 +0300
@@ -149,6 +149,10 @@ struct proto_ops {
     int optname, char __user *optval, int optlen);
     int (*getsockopt)(struct socket *sock, int level,
                     int optname, char __user *optval, int __user *optlen);
+ int (*compat_setsockopt)(struct socket *sock, int level,
+ int optname, char __user *optval, int optlen);
+ int (*compat_getsockopt)(struct socket *sock, int level,
+ int optname, char __user *optval, int __user *optlen);
     int (*sendmsg) (struct kiocb *iocb, struct socket *sock,
                    struct msghdr *m, size_t total_len);
     int (*recvmsg) (struct kiocb *iocb, struct socket *sock,
--- ./include/linux/netfilter.h.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./include/linux/netfilter.h 2006-03-10 12:24:11.000000000 +0300
@@ -80,10 +80,14 @@ struct nf_sockopt_ops
     int set_optmin;
     int set_optmax;
     int (*set)(struct sock *sk, int optval, void __user *user, unsigned int len);
+ int (*compat_set)(struct sock *sk, int optval,
+ void __user *user, unsigned int len);

     int get_optmin;
     int get_optmax;
     int (*get)(struct sock *sk, int optval, void __user *user, int *len);
+ int (*compat_get)(struct sock *sk, int optval,
+ void __user *user, int *len);

     /* Number of users inside set() or get(). */
     unsigned int use;
@@ -246,6 +250,11 @@ int nf_setsockopt(struct sock *sk, int p
int nf_getsockopt(struct sock *sk, int pf, int optval, char __user *opt,
int *len);
```

```

+int compat_nf_setsockopt(struct sock *sk, int pf, int optval,
+ char __user *opt, int len);
+int compat_nf_getsockopt(struct sock *sk, int pf, int optval,
+ char __user *opt, int *len);
+
/* Packet queuing */
struct nf_queue_handler {
    int (*outfn)(struct sk_buff *skb, struct nf_info *info,
--- ./include/net/inet_connection_sock.h.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./include/net/inet_connection_sock.h 2006-03-10 12:24:11.000000000 +0300
@@ -50,6 +50,12 @@ struct inet_connection_sock_af_ops {
    char __user *optval, int optlen);
    int (*getsockopt)(struct sock *sk, int level, int optname,
        char __user *optval, int __user *optlen);
+ int (*compat_setsockopt)(struct sock *sk,
+ int level, int optname,
+ char __user *optval, int optlen);
+ int (*compat_getsockopt)(struct sock *sk,
+ int level, int optname,
+ char __user *optval, int __user *optlen);
    void (*addr2sockaddr)(struct sock *sk, struct sockaddr *);
    int sockaddr_len;
};
--- ./include/net/ip.h.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./include/net/ip.h 2006-03-10 12:24:11.000000000 +0300
@@ -356,6 +356,10 @@ extern void ip_cmsg_recv(struct msghdr *
extern int ip_cmsg_send(struct msghdr *msg, struct ipcm_cookie *ipc);
extern int ip_setsockopt(struct sock *sk, int level, int optname, char __user *optval, int optlen);
extern int ip_getsockopt(struct sock *sk, int level, int optname, char __user *optval, int __user
*optlen);
+extern int compat_ip_setsockopt(struct sock *sk, int level,
+ int optname, char __user *optval, int optlen);
+extern int compat_ip_getsockopt(struct sock *sk, int level,
+ int optname, char __user *optval, int __user *optlen);
extern int ip_ra_control(struct sock *sk, unsigned char on, void (*destructor)(struct sock *));

extern int ip_recv_error(struct sock *sk, struct msghdr *msg, int len);
--- ./include/net/ipv6.h.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./include/net/ipv6.h 2006-03-10 13:16:18.000000000 +0300
@@ -520,6 +520,16 @@ extern int ipv6_getsockopt(struct sock
    int optname,
    char __user *optval,
    int __user *optlen);
+extern int compat_ipv6_setsockopt(struct sock *sk,
+ int level,
+ int optname,
+ char __user *optval,
+ int optlen);

```

```

+extern int  compat_ipv6_getsockopt(struct sock *sk,
+  int level,
+  int optname,
+  char __user *optval,
+  int __user *optlen);

extern void  ipv6_packet_init(void);

--- ./include/net/sctp/structs.h.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./include/net/sctp/structs.h 2006-03-10 12:24:11.000000000 +0300
@@ -514,6 +514,16 @@ struct sctp_af {
    int optname,
    char __user *optval,
    int __user *optlen);
+ int  (*compat_setsockopt) (struct sock *sk,
+  int level,
+  int optname,
+  char __user *optval,
+  int optlen);
+ int  (*compat_getsockopt) (struct sock *sk,
+  int level,
+  int optname,
+  char __user *optval,
+  int __user *optlen);
    struct dst_entry *(*get_dst) (struct sctp_association *asoc,
    union sctp_addr *daddr,
    union sctp_addr *saddr);
--- ./include/net/sock.h.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./include/net/sock.h 2006-03-10 12:24:11.000000000 +0300
@@ -520,6 +520,14 @@ struct proto {
    int  (*getsockopt)(struct sock *sk, int level,
    int optname, char __user *optval,
    int __user *option);
+ int  (*compat_setsockopt)(struct sock *sk,
+  int level,
+  int optname, char __user *optval,
+  int optlen);
+ int  (*compat_getsockopt)(struct sock *sk,
+  int level,
+  int optname, char __user *optval,
+  int __user *option);
    int  (*sendmsg)(struct kiocb *iocb, struct sock *sk,
    struct msghdr *msg, size_t len);
    int  (*recvmsg)(struct kiocb *iocb, struct sock *sk,
@@ -816,6 +824,10 @@ extern int sock_common_recvmsg(struct ki
    struct msghdr *msg, size_t size, int flags);
extern int sock_common_setsockopt(struct socket *sock, int level, int optname,
    char __user *optval, int optlen);

```

```

+extern int compat_sock_common_getsockopt(struct socket *sock, int level,
+ int optname, char __user *optval, int __user *optlen);
+extern int compat_sock_common_setsockopt(struct socket *sock, int level,
+ int optname, char __user *optval, int optlen);

extern void sk_common_release(struct sock *sk);

--- ./include/net/tcp.h.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./include/net/tcp.h 2006-03-10 12:24:11.000000000 +0300
@@ -352,6 +352,12 @@ extern int tcp_getsockopt(struct sock
extern int tcp_setsockopt(struct sock *sk, int level,
    int optname, char __user *optval,
    int optlen);
+extern int compat_tcp_getsockopt(struct sock *sk,
+ int level, int optname,
+ char __user *optval, int __user *optlen);
+extern int compat_tcp_setsockopt(struct sock *sk,
+ int level, int optname,
+ char __user *optval, int optlen);
extern void tcp_set_keepalive(struct sock *sk, int val);
extern int tcp_recvmmsg(struct kiocb *iocb, struct sock *sk,
    struct msghdr *msg,
--- ./net/compat.c.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./net/compat.c 2006-03-10 12:24:11.000000000 +0300
@@ -416,7 +416,7 @@ struct compat_sock_fprog {
    compat_uptr_t filter; /* struct sock_filter */
};

-static int do_set_attach_filter(int fd, int level, int optname,
+static int do_set_attach_filter(struct socket *sock, int level, int optname,
    char __user *optval, int optlen)
{
    struct compat_sock_fprog __user *fprog32 = (struct compat_sock_fprog __user *)optval;
@@ -432,11 +432,12 @@ static int do_set_attach_filter(int fd,
    __put_user(compat_ptr(ptr), &kfprog->filter))
    return -EFAULT;

- return sys_setsockopt(fd, level, optname, (char __user *)kfprog,
+ return sock_setsockopt(sock, level, optname, (char __user *)kfprog,
    sizeof(struct sock_fprog));
}

-static int do_set_sock_timeout(int fd, int level, int optname, char __user *optval, int optlen)
+static int do_set_sock_timeout(struct socket *sock, int level,
+ int optname, char __user *optval, int optlen)
{
    struct compat_timeval __user *up = (struct compat_timeval __user *) optval;
    struct timeval ktime;

```

```

@@ -451,30 +452,61 @@ static int do_set_sock_timeout(int fd, i
    return -EFAULT;
    old_fs = get_fs();
    set_fs(KERNEL_DS);
- err = sys_setsockopt(fd, level, optname, (char *) &ktime, sizeof(ktime));
+ err = sock_setsockopt(sock, level, optname, (char *) &ktime, sizeof(ktime));
    set_fs(old_fs);

    return err;
}

+static int compat_sock_setsockopt(struct socket *sock, int level, int optname,
+ char __user *optval, int optlen)
+{
+ if (optname == SO_ATTACH_FILTER)
+ return do_set_attach_filter(sock, level, optname,
+     optval, optlen);
+ if (optname == SO_RCVTIMEO || optname == SO_SNDTIMEO)
+ return do_set_sock_timeout(sock, level, optname, optval, optlen);
+
+ return sock_setsockopt(sock, level, optname, optval, optlen);
+}
+
+asmlinkage long compat_sys_setsockopt(int fd, int level, int optname,
+ char __user *optval, int optlen)
+ {
+ int err;
+ struct socket *sock;
+
+ /* SO_SET_REPLACE seems to be the same in all levels */
+ if (optname == IPT_SO_SET_REPLACE)
+ return do_netfilter_replace(fd, level, optname,
+     optval, optlen);
- if (level == SOL_SOCKET && optname == SO_ATTACH_FILTER)
- return do_set_attach_filter(fd, level, optname,
-     optval, optlen);
- if (level == SOL_SOCKET &&
-     (optname == SO_RCVTIMEO || optname == SO_SNDTIMEO))
- return do_set_sock_timeout(fd, level, optname, optval, optlen);

- return sys_setsockopt(fd, level, optname, optval, optlen);
+ if (optlen < 0)
+ return -EINVAL;
+
+ if ((sock = sockfd_lookup(fd, &err))!=NULL)
+ {
+ err = security_socket_setsockopt(sock,level,optname);
+ if (err) {

```

```

+ sockfd_put(sock);
+ return err;
+ }
+
+ if (level == SOL_SOCKET)
+   err = compat_sock_setsockopt(sock, level,
+   optname, optval, optlen);
+ else if (sock->ops->compat_setsockopt)
+   err = sock->ops->compat_setsockopt(sock, level,
+   optname, optval, optlen);
+ else
+   err = sock->ops->setsockopt(sock, level,
+   optname, optval, optlen);
+ sockfd_put(sock);
+ }
+ return err;
}

-static int do_get_sock_timeout(int fd, int level, int optname,
+static int do_get_sock_timeout(struct socket *sock, int level, int optname,
    char __user *optval, int __user *optlen)
{
    struct compat_timeval __user *up;
@@ -490,7 +522,7 @@ static int do_get_sock_timeout(int fd, i
    len = sizeof(ktime);
    old_fs = get_fs();
    set_fs(KERNEL_DS);
- err = sys_getsockopt(fd, level, optname, (char *) &ktime, &len);
+ err = sock_getsockopt(sock, level, optname, (char *) &ktime, &len);
    set_fs(old_fs);

    if (!err) {
@@ -503,15 +535,42 @@ static int do_get_sock_timeout(int fd, i
        return err;
    }
}

-asmlinkage long compat_sys_getsockopt(int fd, int level, int optname,
+static int compat_sock_getsockopt(struct socket *sock, int level, int optname,
    char __user *optval, int __user *optlen)
{
- if (level == SOL_SOCKET &&
-     (optname == SO_RCVTIMEO || optname == SO_SNDTIMEO))
-   return do_get_sock_timeout(fd, level, optname, optval, optlen);
- return sys_getsockopt(fd, level, optname, optval, optlen);
+ if (optname == SO_RCVTIMEO || optname == SO_SNDTIMEO)
+   return do_get_sock_timeout(sock, level, optname, optval, optlen);
+ return sock_getsockopt(sock, level, optname, optval, optlen);
}

```

```

+asmlinkage long compat_sys_getsockopt(int fd, int level, int optname,
+ char __user *optval, int __user *optlen)
+{
+ int err;
+ struct socket *sock;
+
+ if ((sock = sockfd_lookup(fd, &err))!=NULL)
+ {
+ err = security_socket_getsockopt(sock, level,
+ optname);
+ if (err) {
+ sockfd_put(sock);
+ return err;
+ }
+
+ if (level == SOL_SOCKET)
+ err = compat_sock_getsockopt(sock, level,
+ optname, optval, optlen);
+ else if (sock->ops->compat_getsockopt)
+ err = sock->ops->compat_getsockopt(sock, level,
+ optname, optval, optlen);
+ else
+ err = sock->ops->getsockopt(sock, level,
+ optname, optval, optlen);
+ sockfd_put(sock);
+ }
+ return err;
+}
/* Argument list sizes for compat_sys_socketcall */
#define AL(x) ((x) * sizeof(u32))
static unsigned char nas[18]={AL(0),AL(3),AL(3),AL(3),AL(2),AL(3),
--- ./net/core/sock.c.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./net/core/sock.c 2006-03-10 12:24:11.000000000 +0300
@@ -1385,6 +1385,20 @@ int sock_common_getsockopt(struct socket

EXPORT_SYMBOL(sock_common_getsockopt);

#ifdef CONFIG_COMPAT
+int compat_sock_common_getsockopt(struct socket *sock, int level,
+ int optname, char __user *optval, int __user *optlen)
+{
+ struct sock *sk = sock->sk;
+
+ if (sk->sk_prot->compat_setsockopt)
+ return sk->sk_prot->compat_getsockopt(sk, level,
+ optname, optval, optlen);
+ return sk->sk_prot->getsockopt(sk, level, optname, optval, optlen);

```

```

+}
+EXPORT_SYMBOL(compat_sock_common_getsockopt);
+#endif
+
int sock_common_recvmsg(struct kiocb *iocb, struct socket *sock,
    struct msghdr *msg, size_t size, int flags)
{
@@ -1414,6 +1428,20 @@ int sock_common_setsockopt(struct socket

EXPORT_SYMBOL(sock_common_setsockopt);

#ifdef CONFIG_COMPAT
+int compat_sock_common_setsockopt(struct socket *sock,
+ int level, int optname, char __user *optval, int optlen)
+{
+ struct sock *sk = sock->sk;
+
+ if (sk->sk_prot->compat_setsockopt)
+ return sk->sk_prot->compat_setsockopt(sk, level,
+ optname, optval, optlen);
+ return sk->sk_prot->setsockopt(sk, level, optname, optval, optlen);
+}
+EXPORT_SYMBOL(compat_sock_common_setsockopt);
+#endif
+
void sk_common_release(struct sock *sk)
{
    if (sk->sk_prot->destroy)
--- ./net/dccp/dccp.h.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./net/dccp/dccp.h 2006-03-10 12:24:11.000000000 +0300
@@ -192,6 +192,14 @@ extern int dccp_getsockopt(struct soc
    char __user *optval, int __user *optlen);
extern int dccp_setsockopt(struct sock *sk, int level, int optname,
    char __user *optval, int optlen);
#ifdef CONFIG_COMPAT
+extern int compat_dccp_getsockopt(struct sock *sk,
+ int level, int optname,
+ char __user *optval, int __user *optlen);
+extern int compat_dccp_setsockopt(struct sock *sk,
+ int level, int optname,
+ char __user *optval, int optlen);
+#endif
extern int dccp_ioctl(struct sock *sk, int cmd, unsigned long arg);
extern int dccp_sendmsg(struct kiocb *iocb, struct sock *sk,
    struct msghdr *msg, size_t size);
--- ./net/dccp/ipv4.c.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./net/dccp/ipv4.c 2006-03-10 12:30:33.000000000 +0300
@@ -994,6 +994,10 @@ static struct inet_connection_sock_af_op

```

```

.net_header_len = sizeof(struct iphdr),
.setsockopt = ip_setsockopt,
.getsockopt = ip_getsockopt,
#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_ip_setsockopt,
+ .compat_getsockopt = compat_ip_getsockopt,
#endif
.addr2sockaddr = inet_csk_addr2sockaddr,
.sockaddr_len = sizeof(struct sockaddr_in),
};
@@ -1040,6 +1044,10 @@ static struct proto dccp_v4_prot = {
.init = dccp_v4_init_sock,
.setsockopt = dccp_setsockopt,
.getsockopt = dccp_getsockopt,
#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_dccp_setsockopt,
+ .compat_getsockopt = compat_dccp_getsockopt,
#endif
.sendmsg = dccp_sendmsg,
.recvmsg = dccp_recvmsg,
.backlog_rcv = dccp_v4_do_rcv,
@@ -1079,6 +1087,10 @@ static const struct proto_ops inet_dccp_
.shutdown = inet_shutdown,
.setsockopt = sock_common_setsockopt,
.getsockopt = sock_common_getsockopt,
#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_sock_common_setsockopt,
+ .compat_getsockopt = compat_sock_common_getsockopt,
#endif
.sendmsg = inet_sendmsg,
.recvmsg = sock_common_recvmsg,
.mmap = sock_no_mmap,
--- ./net/dccp/ipv6.c.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./net/dccp/ipv6.c 2006-03-10 13:18:04.000000000 +0300
@@ -1114,6 +1114,10 @@ static struct inet_connection_sock_af_op
.net_header_len = sizeof(struct ipv6hdr),
.setsockopt = ipv6_setsockopt,
.getsockopt = ipv6_getsockopt,
#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_ipv6_setsockopt,
+ .compat_getsockopt = compat_ipv6_getsockopt,
#endif
.addr2sockaddr = inet6_csk_addr2sockaddr,
.sockaddr_len = sizeof(struct sockaddr_in6)
};
@@ -1130,6 +1134,10 @@ static struct inet_connection_sock_af_op
.net_header_len = sizeof(struct iphdr),
.setsockopt = ipv6_setsockopt,

```

```

.getsockopt = ipv6_getsockopt,
#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_ipv6_setsockopt,
+ .compat_getsockopt = compat_ipv6_getsockopt,
#endif
.addr2sockaddr = inet6_csk_addr2sockaddr,
.sockaddr_len = sizeof(struct sockaddr_in6)
};
@@ -1167,6 +1175,10 @@ static struct proto dccp_v6_prot = {
.init = dccp_v6_init_sock,
.setsockopt = dccp_setsockopt,
.getsockopt = dccp_getsockopt,
#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_dccp_setsockopt,
+ .compat_getsockopt = compat_dccp_getsockopt,
#endif
.sendmsg = dccp_sendmsg,
.recvmsg = dccp_recvmsg,
.backlog_rcv = dccp_v6_do_rcv,
@@ -1204,6 +1216,10 @@ static struct proto_ops inet6_dccp_ops =
.shutdown = inet_shutdown,
.setsockopt = sock_common_setsockopt,
.getsockopt = sock_common_getsockopt,
#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_sock_common_setsockopt,
+ .compat_getsockopt = compat_sock_common_getsockopt,
#endif
.sendmsg = inet_sendmsg,
.recvmsg = sock_common_recvmsg,
.mmap = sock_no_mmap,
--- ./net/dccp/proto.c.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./net/dccp/proto.c 2006-03-10 12:24:11.000000000 +0300
@@ -455,18 +455,13 @@ out_free_val:
goto out;
}

```

```

-int dccp_setsockopt(struct sock *sk, int level, int optname,
- char __user *optval, int optlen)
+static int do_dccp_setsockopt(struct sock *sk, int level, int optname,
+ char __user *optval, int optlen)
{
struct dccp_sock *dp;
int err;
int val;

- if (level != SOL_DCCP)
- return inet_csk(sk)->icsk_af_ops->setsockopt(sk, level,
- optname, optval,

```

```

-     optlen);
-
+ if (optlen < sizeof(int))
+   return -EINVAL;

@@ -512,8 +507,34 @@ int dccp_setsockopt(struct sock *sk, int
+   return err;
+ }

+int dccp_setsockopt(struct sock *sk, int level, int optname,
+ char __user *optval, int optlen)
+{
+ if (level != SOL_DCCP)
+ return inet_csk(sk)->icsk_af_ops->setsockopt(sk, level,
+   optname, optval,
+   optlen);
+ return do_dccp_setsockopt(sk, level, optname, optval, optlen);
+}
EXPORT_SYMBOL_GPL(dccp_setsockopt);

+#ifdef CONFIG_COMPAT
+int compat_dccp_setsockopt(struct sock *sk, int level, int optname,
+ char __user *optval, int optlen)
+{
+ if (level != SOL_DCCP) {
+ if (inet_csk(sk)->icsk_af_ops->compat_setsockopt)
+ return inet_csk(sk)->icsk_af_ops->compat_setsockopt(sk,
+   level, optname, optval, optlen);
+ else
+ return inet_csk(sk)->icsk_af_ops->setsockopt(sk,
+   level, optname, optval, optlen);
+ }
+ return do_dccp_setsockopt(sk, level, optname, optval, optlen);
+}
+EXPORT_SYMBOL_GPL(compat_dccp_setsockopt);
+#endif
+
+static int dccp_getsockopt_service(struct sock *sk, int len,
+   __be32 __user *optval,
+   int __user *optlen)
@@ -545,16 +566,12 @@ out:
+   return err;
+ }

-int dccp_getsockopt(struct sock *sk, int level, int optname,
+static int do_dccp_getsockopt(struct sock *sk, int level, int optname,
+   char __user *optval, int __user *optlen)
+ {

```

```

struct dccp_sock *dp;
int val, len;

- if (level != SOL_DCCP)
- return inet_csk(sk)->icsk_af_ops->getsockopt(sk, level,
-       optname, optval,
-       optlen);
  if (get_user(len, optlen))
    return -EFAULT;

@@ -587,8 +604,34 @@ int dccp_getsockopt(struct sock *sk, int
  return 0;
}

+int dccp_getsockopt(struct sock *sk, int level, int optname,
+  char __user *optval, int __user *optlen)
+{
+ if (level != SOL_DCCP)
+ return inet_csk(sk)->icsk_af_ops->getsockopt(sk, level,
+       optname, optval,
+       optlen);
+ return do_dccp_getsockopt(sk, level, optname, optval, optlen);
+}
EXPORT_SYMBOL_GPL(dccp_getsockopt);

+#ifdef CONFIG_COMPAT
+int compat_dccp_getsockopt(struct sock *sk, int level, int optname,
+  char __user *optval, int __user *optlen)
+{
+ if (level != SOL_DCCP) {
+ if (inet_csk(sk)->icsk_af_ops->compat_setsockopt)
+ return inet_csk(sk)->icsk_af_ops->compat_getsockopt(sk,
+ level, optname, optval, optlen);
+ else
+ return inet_csk(sk)->icsk_af_ops->getsockopt(sk,
+ level, optname, optval, optlen);
+ }
+ return do_dccp_getsockopt(sk, level, optname, optval, optlen);
+}
+EXPORT_SYMBOL_GPL(compat_dccp_getsockopt);
+#endif
+
+int dccp_sendmsg(struct kiocb *iocb, struct sock *sk, struct msghdr *msg,
+  size_t len)
+{
--- ./net/ipv4/af_inet.c.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./net/ipv4/af_inet.c 2006-03-10 12:24:11.000000000 +0300
@@ -802,6 +802,10 @@ const struct proto_ops inet_stream_ops =

```

```

.shutdown = inet_shutdown,
.setsockopt = sock_common_setsockopt,
.getsockopt = sock_common_getsockopt,
#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_sock_common_setsockopt,
+ .compat_getsockopt = compat_sock_common_getsockopt,
#endif
.sendmsg = inet_sendmsg,
.recvmsg = sock_common_recvmsg,
.mmap = sock_no_mmap,
@@ -823,6 +827,10 @@ const struct proto_ops inet_dgram_ops =
.shutdown = inet_shutdown,
.setsockopt = sock_common_setsockopt,
.getsockopt = sock_common_getsockopt,
#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_sock_common_setsockopt,
+ .compat_getsockopt = compat_sock_common_getsockopt,
#endif
.sendmsg = inet_sendmsg,
.recvmsg = sock_common_recvmsg,
.mmap = sock_no_mmap,
@@ -848,6 +856,10 @@ static const struct proto_ops inet_sockr
.shutdown = inet_shutdown,
.setsockopt = sock_common_setsockopt,
.getsockopt = sock_common_getsockopt,
#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_sock_common_setsockopt,
+ .compat_getsockopt = compat_sock_common_getsockopt,
#endif
.sendmsg = inet_sendmsg,
.recvmsg = sock_common_recvmsg,
.mmap = sock_no_mmap,
--- ./net/ipv4/ip_sockglue.c.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./net/ipv4/ip_sockglue.c 2006-03-10 12:24:11.000000000 +0300
@@ -380,14 +380,12 @@ out:
 * an IP socket.
 */

```

```

-int ip_setsockopt(struct sock *sk, int level, int optname, char __user *optval, int optlen)
+static int do_ip_setsockopt(struct sock *sk, int level,
+ int optname, char __user *optval, int optlen)
{
struct inet_sock *inet = inet_sk(sk);
int val=0,err;

- if (level != SOL_IP)
- return -ENOPROTOOPT;
-

```

```

if (((1<<optname) & ((1<<IP_PKTINFO) | (1<<IP_RECVTTL) |
    (1<<IP_RECVOPTS) | (1<<IP_RECVTOS) |
    (1<<IP_RETOPTS) | (1<<IP_TOS) |
@@ -849,12 +847,7 @@ mc_msf_out:
    break;

    default:
-#ifdef CONFIG_NETFILTER
-    err = nf_setsockopt(sk, PF_INET, optname, optval,
-        optlen);
-#else
    err = -ENOPROTOOPT;
-#endif
    break;
    }
    release_sock(sk);
@@ -865,12 +858,66 @@ e_inval:
    return -EINVAL;
}

+int ip_setsockopt(struct sock *sk, int level,
+ int optname, char __user *optval, int optlen)
+{
+ int err;
+
+ if (level != SOL_IP)
+ return -ENOPROTOOPT;
+
+ err = do_ip_setsockopt(sk, level, optname, optval, optlen);
+#ifdef CONFIG_NETFILTER
+ /* we need to exclude all possible ENOPROTOOPTs except default case */
+ if (err == -ENOPROTOOPT && optname != IP_HDRINCL &&
+ optname != IP_IPSEC_POLICY && optname != IP_XFRM_POLICY
+#ifdef CONFIG_IP_MROUTE
+ && (optname < MRT_BASE || optname > (MRT_BASE + 10))
+#endif
+ ) {
+ lock_sock(sk);
+ err = nf_setsockopt(sk, PF_INET, optname, optval, optlen);
+ release_sock(sk);
+ }
+#endif
+ return err;
+}
+
+#ifdef CONFIG_COMPAT
+int compat_ip_setsockopt(struct sock *sk, int level,
+ int optname, char __user *optval, int optlen)

```

```

+{
+ int err;
+
+ if (level != SOL_IP)
+ return -ENOPROTOOPT;
+
+ err = do_ip_setsockopt(sk, level, optname, optval, optlen);
+#ifdef CONFIG_NETFILTER
+ /* we need to exclude all possible ENOPROTOOPTs except default case */
+ if (err == -ENOPROTOOPT && optname != IP_HDRINCL &&
+ optname != IP_IPSEC_POLICY && optname != IP_XFRM_POLICY
+#ifdef CONFIG_IP_MROUTE
+ && (optname < MRT_BASE || optname > (MRT_BASE + 10))
+#endif
+ ) {
+ lock_sock(sk);
+ err = compat_nf_setsockopt(sk, PF_INET,
+ optname, optval, optlen);
+ release_sock(sk);
+ }
+#endif
+ return err;
+}
+#endif
+
+/*
+ * Get the options. Note for future reference. The GET of IP options gets the
+ * _received_ ones. The set sets the _sent_ ones.
+ */

-int ip_getsockopt(struct sock *sk, int level, int optname, char __user *optval, int __user *optlen)
+static int do_ip_getsockopt(struct sock *sk, int level, int optname,
+ char __user *optval, int __user *optlen)
{
    struct inet_sock *inet = inet_sk(sk);
    int val;
@@ -1051,17 +1098,8 @@ int ip_getsockopt(struct sock *sk, int l
    val = inet->freebind;
    break;
    default:
-#ifdef CONFIG_NETFILTER
-    val = nf_getsockopt(sk, PF_INET, optname, optval,
-        &len);
-    release_sock(sk);
-    if (val >= 0)
-        val = put_user(len, optlen);
-    return val;
-#else

```

```

    release_sock(sk);
    return -ENOPROTOOPT;
-#endif
}
    release_sock(sk);

@@ -1082,7 +1120,73 @@ int ip_getsockopt(struct sock *sk, int l
    return 0;
}

+int ip_getsockopt(struct sock *sk, int level,
+ int optname, char __user *optval, int __user *optlen)
+{
+ int err;
+
+ err = do_ip_getsockopt(sk, level, optname, optval, optlen);
+#ifdef CONFIG_NETFILTER
+ /* we need to exclude all possible ENOPROTOOPTs except default case */
+ if (err == -ENOPROTOOPT && optname != IP_PKTOPTIONS
+#ifdef CONFIG_IP_MROUTE
+ && (optname < MRT_BASE || optname > MRT_BASE+10)
+#endif
+ ) {
+     int len;
+
+     if(get_user(len,optlen))
+         return -EFAULT;
+
+     lock_sock(sk);
+     err = nf_getsockopt(sk, PF_INET, optname, optval,
+         &len);
+     release_sock(sk);
+     if (err >= 0)
+         err = put_user(len, optlen);
+     return err;
+ }
+#endif
+ return err;
+}
+
+#ifdef CONFIG_COMPAT
+int compat_ip_getsockopt(struct sock *sk, int level,
+ int optname, char __user *optval, int __user *optlen)
+{
+ int err;
+
+ err = do_ip_getsockopt(sk, level, optname, optval, optlen);
+#ifdef CONFIG_NETFILTER

```

```

+ /* we need to exclude all possible ENOPROTOOPTs except default case */
+ if (err == -ENOPROTOOPT && optname != IP_PKTOPTIONS
+#ifdef CONFIG_IP_MROUTE
+ && (optname < MRT_BASE || optname > MRT_BASE+10)
+#endif
+ ) {
+   int len;
+
+   if(get_user(len,optlen))
+     return -EFAULT;
+
+   lock_sock(sk);
+   err = compat_nf_getsockopt(sk, PF_INET,
+   optname, optval, &len);
+   release_sock(sk);
+   if (err >= 0)
+     err = put_user(len, optlen);
+   return err;
+ }
+#endif
+ return err;
+}
+#endif
+
EXPORT_SYMBOL(ip_cmsg_rcv);

EXPORT_SYMBOL(ip_getsockopt);
EXPORT_SYMBOL(ip_setsockopt);
+#ifdef CONFIG_COMPAT
+EXPORT_SYMBOL(compat_ip_getsockopt);
+EXPORT_SYMBOL(compat_ip_setsockopt);
+#endif
--- ./net/ipv4/raw.c.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./net/ipv4/raw.c 2006-03-10 12:24:11.000000000 +0300
@@ -660,12 +660,9 @@ static int raw_geticmpfilter(struct sock
out: return ret;
}

-static int raw_setsockopt(struct sock *sk, int level, int optname,
+static int do_raw_setsockopt(struct sock *sk, int level, int optname,
+   char __user *optval, int optlen)
{
- if (level != SOL_RAW)
- return ip_setsockopt(sk, level, optname, optval, optlen);
-
if (optname == ICMP_FILTER) {
if (inet_sk(sk)->num != IPPROTO_ICMP)
return -EOPNOTSUPP;

```

```

@@ -675,12 +672,28 @@ static int raw_setsockopt(struct sock *s
    return -ENOPROTOOPT;
}

-static int raw_getsockopt(struct sock *sk, int level, int optname,
-    char __user *optval, int __user *optlen)
+static int raw_setsockopt(struct sock *sk, int level, int optname,
+    char __user *optval, int optlen)
{
    if (level != SOL_RAW)
-    return ip_getsockopt(sk, level, optname, optval, optlen);
+    return ip_setsockopt(sk, level, optname, optval, optlen);
+    return do_raw_setsockopt(sk, level, optname, optval, optlen);
+}

+#ifdef CONFIG_COMPAT
+static int compat_raw_setsockopt(struct sock *sk, int level, int optname,
+    char __user *optval, int optlen)
+{
+    if (level != SOL_RAW)
+    return compat_ip_setsockopt(sk, level,
+    optname, optval, optlen);
+    return do_raw_setsockopt(sk, level, optname, optval, optlen);
+}
+#endif
+
+static int do_raw_getsockopt(struct sock *sk, int level, int optname,
+    char __user *optval, int __user *optlen)
+{
+    if (optname == ICMP_FILTER) {
+        if (inet_sk(sk)->num != IPPROTO_ICMP)
+            return -EOPNOTSUPP;
@@ -690,6 +703,25 @@ static int raw_getsockopt(struct sock *s
    return -ENOPROTOOPT;
}

+static int raw_getsockopt(struct sock *sk, int level, int optname,
+    char __user *optval, int __user *optlen)
+{
+    if (level != SOL_RAW)
+    return ip_getsockopt(sk, level, optname, optval, optlen);
+    return do_raw_getsockopt(sk, level, optname, optval, optlen);
+}
+
+#ifdef CONFIG_COMPAT
+static int compat_raw_getsockopt(struct sock *sk, int level, int optname,
+    char __user *optval, int __user *optlen)
+{

```

```

+ if (level != SOL_RAW)
+ return compat_ip_getsockopt(sk, level,
+ optname, optval, optlen);
+ return do_raw_getsockopt(sk, level, optname, optval, optlen);
+}
+#endif
+
static int raw_ioctl(struct sock *sk, int cmd, unsigned long arg)
{
switch (cmd) {
@@ -728,6 +760,10 @@ struct proto raw_prot = {
.init = raw_init,
.setsockopt = raw_setsockopt,
.getsockopt = raw_getsockopt,
#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_raw_setsockopt,
+ .compat_getsockopt = compat_raw_getsockopt,
#endif
.sendmsg = raw_sendmsg,
.recvmsg = raw_recvmsg,
.bind = raw_bind,
--- ./net/ipv4/tcp.c.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./net/ipv4/tcp.c 2006-03-10 12:24:11.000000000 +0300
@@ -1687,18 +1687,14 @@ int tcp_disconnect(struct sock *sk, int
/*
* Socket option code for TCP.
*/
-int tcp_setsockopt(struct sock *sk, int level, int optname, char __user *optval,
- int optlen)
+static int do_tcp_setsockopt(struct sock *sk, int level,
+ int optname, char __user *optval, int optlen)
{
struct tcp_sock *tp = tcp_sk(sk);
struct inet_connection_sock *icsk = inet_csk(sk);
int val;
int err = 0;

- if (level != SOL_TCP)
- return icsk->icsk_af_ops->setsockopt(sk, level, optname,
- optval, optlen);
-
/* This is a string value all the others are int's */
if (optname == TCP_CONGESTION) {
char name[TCP_CA_NAME_MAX];
@@ -1871,6 +1867,35 @@ int tcp_setsockopt(struct sock *sk, int
return err;
}

```

```

+int tcp_setsockopt(struct sock *sk, int level, int optname, char __user *optval,
+ int optlen)
+{
+ struct inet_connection_sock *icsk = inet_csk(sk);
+
+ if (level != SOL_TCP)
+ return icsk->icsk_af_ops->setsockopt(sk, level, optname,
+ optval, optlen);
+ return do_tcp_setsockopt(sk, level, optname, optval, optlen);
+}
+
+#ifdef CONFIG_COMPAT
+int compat_tcp_setsockopt(struct sock *sk, int level,
+ int optname, char __user *optval, int optlen)
+{
+ struct inet_connection_sock *icsk = inet_csk(sk);
+
+ if (level != SOL_TCP) {
+ if (icsk->icsk_af_ops->compat_setsockopt)
+ return icsk->icsk_af_ops->compat_setsockopt(sk,
+ level, optname, optval, optlen);
+ else
+ return icsk->icsk_af_ops->setsockopt(sk,
+ level, optname, optval, optlen);
+ }
+ return do_tcp_setsockopt(sk, level, optname, optval, optlen);
+}
+#endif
+
+/* Return information about state of tcp endpoint in API format. */
+void tcp_get_info(struct sock *sk, struct tcp_info *info)
+{
@@ -1931,17 +1956,13 @@ void tcp_get_info(struct sock *sk, struc

EXPORT_SYMBOL_GPL(tcp_get_info);

-int tcp_getsockopt(struct sock *sk, int level, int optname, char __user *optval,
- int __user *optlen)
+static int do_tcp_getsockopt(struct sock *sk, int level,
+ int optname, char __user *optval, int __user *optlen)
+{
+ struct inet_connection_sock *icsk = inet_csk(sk);
+ struct tcp_sock *tp = tcp_sk(sk);
+ int val, len;

- if (level != SOL_TCP)
- return icsk->icsk_af_ops->getsockopt(sk, level, optname,
- optval, optlen);

```

```

-
  if (get_user(len, optlen))
    return -EFAULT;

@@ -2025,6 +2046,34 @@ int tcp_getsockopt(struct sock *sk, int
  return 0;
}

+int tcp_getsockopt(struct sock *sk, int level, int optname, char __user *optval,
+ int __user *optlen)
+{
+ struct inet_connection_sock *icsk = inet_csk(sk);
+
+ if (level != SOL_TCP)
+ return icsk->icsk_af_ops->getsockopt(sk, level, optname,
+ optval, optlen);
+ return do_tcp_getsockopt(sk, level, optname, optval, optlen);
+}
+
+ #ifdef CONFIG_COMPAT
+int compat_tcp_getsockopt(struct sock *sk, int level,
+ int optname, char __user *optval, int __user *optlen)
+{
+ struct inet_connection_sock *icsk = inet_csk(sk);
+
+ if (level != SOL_TCP) {
+ if (icsk->icsk_af_ops->compat_getsockopt)
+ return icsk->icsk_af_ops->compat_getsockopt(sk,
+ level, optname, optval, optlen);
+ else
+ return icsk->icsk_af_ops->getsockopt(sk,
+ level, optname, optval, optlen);
+ }
+ return do_tcp_getsockopt(sk, level, optname, optval, optlen);
+}
+ #endif

extern void __skb_cb_too_small_for_tcp(int, int);
extern struct tcp_congestion_ops tcp_reno;
@@ -2142,3 +2191,7 @@ EXPORT_SYMBOL(tcp_sendpage);
EXPORT_SYMBOL(tcp_setsockopt);
EXPORT_SYMBOL(tcp_shutdown);
EXPORT_SYMBOL(tcp_statistics);
+ #ifdef CONFIG_COMPAT
+EXPORT_SYMBOL(compat_tcp_setsockopt);
+EXPORT_SYMBOL(compat_tcp_getsockopt);
+ #endif
--- ./net/ipv4/tcp_ipv4.c.compat 2006-03-10 11:58:11.000000000 +0300

```

```

+++ ./net/ipv4/tcp_ipv4.c 2006-03-10 12:24:11.000000000 +0300
@@ -1226,6 +1226,10 @@ struct inet_connection_sock_af_ops ipv4_
    .net_header_len = sizeof(struct iphdr),
    .setsockopt = ip_setsockopt,
    .getsockopt = ip_getsockopt,
+#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_ip_setsockopt,
+ .compat_getsockopt = compat_ip_getsockopt,
+#endif
    .addr2sockaddr = inet_csk_addr2sockaddr,
    .sockaddr_len = sizeof(struct sockaddr_in),
};
@@ -1808,6 +1812,10 @@ struct proto tcp_prot = {
    .shutdown = tcp_shutdown,
    .setsockopt = tcp_setsockopt,
    .getsockopt = tcp_getsockopt,
+#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_tcp_setsockopt,
+ .compat_getsockopt = compat_tcp_getsockopt,
+#endif
    .sendmsg = tcp_sendmsg,
    .recvmsg = tcp_recvmsg,
    .backlog_rcv = tcp_v4_do_rcv,
--- ./net/ipv4/udp.c.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./net/ipv4/udp.c 2006-03-10 12:24:11.000000000 +0300
@@ -1207,16 +1207,13 @@ static int udp_destroy_sock(struct sock
/*
 * Socket option code for UDP
 */
-static int udp_setsockopt(struct sock *sk, int level, int optname,
+static int do_udp_setsockopt(struct sock *sk, int level, int optname,
    char __user *optval, int optlen)
{
    struct udp_sock *up = udp_sk(sk);
    int val;
    int err = 0;

- if (level != SOL_UDP)
- return ip_setsockopt(sk, level, optname, optval, optlen);
-
    if(optlen<sizeof(int))
        return -EINVAL;

@@ -1256,15 +1253,31 @@ static int udp_setsockopt(struct sock *s
    return err;
}

-static int udp_getsockopt(struct sock *sk, int level, int optname,

```

```

+static int udp_setsockopt(struct sock *sk, int level, int optname,
+ char __user *optval, int optlen)
+{
+ if (level != SOL_UDP)
+ return ip_setsockopt(sk, level, optname, optval, optlen);
+ return do_udp_setsockopt(sk, level, optname, optval, optlen);
+}
+
+#ifdef CONFIG_COMPAT
+static int compat_udp_setsockopt(struct sock *sk, int level, int optname,
+ char __user *optval, int optlen)
+{
+ if (level != SOL_UDP)
+ return compat_ip_setsockopt(sk, level,
+ optname, optval, optlen);
+ return do_udp_setsockopt(sk, level, optname, optval, optlen);
+}
+#endif
+
+static int do_udp_getsockopt(struct sock *sk, int level, int optname,
+ char __user *optval, int __user *optlen)
+{
+ struct udp_sock *up = udp_sk(sk);
+ int val, len;

- if (level != SOL_UDP)
- return ip_getsockopt(sk, level, optname, optval, optlen);
-
+ if(get_user(len,optlen))
+ return -EFAULT;

@@ -1293,6 +1306,24 @@ static int udp_getsockopt(struct sock *s
+ return 0;
+}

+static int udp_getsockopt(struct sock *sk, int level, int optname,
+ char __user *optval, int __user *optlen)
+{
+ if (level != SOL_UDP)
+ return ip_getsockopt(sk, level, optname, optval, optlen);
+ return do_udp_getsockopt(sk, level, optname, optval, optlen);
+}
+
+#ifdef CONFIG_COMPAT
+static int compat_udp_getsockopt(struct sock *sk, int level, int optname,
+ char __user *optval, int __user *optlen)
+{
+ if (level != SOL_UDP)

```

```

+ return compat_ip_getsockopt(sk, level,
+ optname, optval, optlen);
+ return do_udp_getsockopt(sk, level, optname, optval, optlen);
+}
+#endif
/**
 * udp_poll - wait for a UDP event.
 * @file - file struct
@@ -1350,6 +1381,10 @@ struct proto udp_prot = {
 .destroy = udp_destroy_sock,
 .setsockopt = udp_setsockopt,
 .getsockopt = udp_getsockopt,
+#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_udp_setsockopt,
+ .compat_getsockopt = compat_udp_getsockopt,
+#endif
 .sendmsg = udp_sendmsg,
 .recvmsg = udp_recvmsg,
 .sendpage = udp_sendpage,
--- ./net/ipv6/af_inet6.c.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./net/ipv6/af_inet6.c 2006-03-10 12:24:11.000000000 +0300
@@ -470,6 +470,10 @@ const struct proto_ops inet6_stream_ops
 .shutdown = inet_shutdown, /* ok */
 .setsockopt = sock_common_setsockopt, /* ok */
 .getsockopt = sock_common_getsockopt, /* ok */
+#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_sock_common_setsockopt,
+ .compat_getsockopt = compat_sock_common_getsockopt,
+#endif
 .sendmsg = inet_sendmsg, /* ok */
 .recvmsg = sock_common_recvmsg, /* ok */
 .mmap = sock_no_mmap,
@@ -491,6 +495,10 @@ const struct proto_ops inet6_dgram_ops =
 .shutdown = inet_shutdown, /* ok */
 .setsockopt = sock_common_setsockopt, /* ok */
 .getsockopt = sock_common_getsockopt, /* ok */
+#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_sock_common_setsockopt,
+ .compat_getsockopt = compat_sock_common_getsockopt,
+#endif
 .sendmsg = inet_sendmsg, /* ok */
 .recvmsg = sock_common_recvmsg, /* ok */
 .mmap = sock_no_mmap,
@@ -519,6 +527,10 @@ static const struct proto_ops inet6_sock
 .shutdown = inet_shutdown, /* ok */
 .setsockopt = sock_common_setsockopt, /* ok */
 .getsockopt = sock_common_getsockopt, /* ok */
+#ifdef CONFIG_COMPAT

```

```

+ .compat_setsockopt = compat_sock_common_setsockopt,
+ .compat_getsockopt = compat_sock_common_getsockopt,
+#endif
  .sendmsg = inet_sendmsg, /* ok */
  .recvmsg = sock_common_recvmsg, /* ok */
  .mmap = sock_no_mmap,
--- ./net/ipv6/ipv6_sockglue.c.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./net/ipv6/ipv6_sockglue.c 2006-03-10 14:02:07.000000000 +0300
@@ -109,19 +109,13 @@ int ip6_ra_control(struct sock *sk, int
    return 0;
}

-int ipv6_setsockopt(struct sock *sk, int level, int optname,
+static int do_ipv6_setsockopt(struct sock *sk, int level, int optname,
    char __user *optval, int optlen)
{
    struct ipv6_pinfo *np = inet6_sk(sk);
    int val, valbool;
    int retv = -ENOPROTOOPT;

- if (level == SOL_IP && sk->sk_type != SOCK_RAW)
- return udp_prot.setsockopt(sk, level, optname, optval, optlen);
-
- if(level!=SOL_IPV6)
- goto out;
-
    if (optval == NULL)
        val=0;
    else if (get_user(val, (int __user *) optval))
@@ -613,17 +607,9 @@ done:
    retv = xfrm_user_policy(sk, optname, optval, optlen);
    break;

-#ifdef CONFIG_NETFILTER
- default:
- retv = nf_setsockopt(sk, PF_INET6, optname, optval,
-    optlen);
- break;
-#endif
-
}
release_sock(sk);

-out:
    return retv;

e_inval:
@@ -631,6 +617,65 @@ e_inval:

```

```

    return -EINVAL;
}

+int ipv6_setsockopt(struct sock *sk, int level, int optname,
+    char __user *optval, int optlen)
+{
+ int err;
+
+ if (level == SOL_IP && sk->sk_type != SOCK_RAW)
+ return udp_prot.setsockopt(sk, level, optname, optval, optlen);
+
+ if (level != SOL_IPV6)
+ return -ENOPROTOOPT;
+
+ err = do_ipv6_setsockopt(sk, level, optname, optval, optlen);
+#ifdef CONFIG_NETFILTER
+ /* we need to exclude all possible ENOPROTOOPTs except default case */
+ if (err == -ENOPROTOOPT && optname != IPV6_IPSEC_POLICY &&
+     optname != IPV6_XFRM_POLICY) {
+ lock_sock(sk);
+ err = nf_setsockopt(sk, PF_INET6, optname, optval,
+     optlen);
+ release_sock(sk);
+ }
+#endif
+ return err;
+}
+
+#ifdef CONFIG_COMPAT
+int compat_ipv6_setsockopt(struct sock *sk, int level, int optname,
+    char __user *optval, int optlen)
+{
+ int err;
+
+ if (level == SOL_IP && sk->sk_type != SOCK_RAW) {
+ if (udp_prot.compat_setsockopt)
+ return udp_prot.compat_setsockopt(sk, level,
+     optname, optval, optlen);
+ else
+ return udp_prot.setsockopt(sk, level,
+     optname, optval, optlen);
+ }
+
+ if (level != SOL_IPV6)
+ return -ENOPROTOOPT;
+
+ err = do_ipv6_setsockopt(sk, level, optname, optval, optlen);

```

```

+#ifdef CONFIG_NETFILTER
+ /* we need to exclude all possible ENOPROTOOPTs except default case */
+ if (err == -ENOPROTOOPT && optname != IPV6_IPSEC_POLICY &&
+ optname != IPV6_XFRM_POLICY) {
+ lock_sock(sk);
+ err = compat_nf_setsockopt(sk, PF_INET6, optname, optval,
+ optlen);
+ release_sock(sk);
+ }
+#endif
+ return err;
+}
+#endif
+
static int ipv6_getsockopt_sticky(struct sock *sk, struct ipv6_opt_hdr *hdr,
char __user *optval, int len)
{
@@ -642,17 +687,13 @@ static int ipv6_getsockopt_sticky(struct
return len;
}

-int ipv6_getsockopt(struct sock *sk, int level, int optname,
+static int do_ipv6_getsockopt(struct sock *sk, int level, int optname,
char __user *optval, int __user *optlen)
{
struct ipv6_pinfo *np = inet6_sk(sk);
int len;
int val;

- if (level == SOL_IP && sk->sk_type != SOCK_RAW)
- return udp_prot.getsockopt(sk, level, optname, optval, optlen);
- if(level!=SOL_IPV6)
- return -ENOPROTOOPT;
if (get_user(len, optlen))
return -EFAULT;
switch (optname) {
@@ -842,17 +883,7 @@ int ipv6_getsockopt(struct sock *sk, int
break;

default:
-#ifdef CONFIG_NETFILTER
- lock_sock(sk);
- val = nf_getsockopt(sk, PF_INET6, optname, optval,
- &len);
- release_sock(sk);
- if (val >= 0)
- val = put_user(len, optlen);
- return val;

```

```

-#else
    return -EINVAL;
-#endif
}
len = min_t(unsigned int, sizeof(int), len);
if(put_user(len, optlen))
@@ -862,6 +893,78 @@ int ipv6_getsockopt(struct sock *sk, int
    return 0;
}

+int ipv6_getsockopt(struct sock *sk, int level, int optname,
+    char __user *optval, int __user *optlen)
+{
+ int err;
+
+
+ if (level == SOL_IP && sk->sk_type != SOCK_RAW)
+ return udp_prot.getsockopt(sk, level, optname, optval, optlen);
+
+
+ if(level != SOL_IPV6)
+ return -ENOPROTOOPT;
+
+
+ err = do_ipv6_getsockopt(sk, level, optname, optval, optlen);
+#ifdef CONFIG_NETFILTER
+ /* we need to exclude all possible EINVALs except default case */
+ if (err == -ENOPROTOOPT && optname != IPV6_ADDRFORM &&
+ optname != MCAST_MSFILTER) {
+ int len;
+
+
+ if (get_user(len, optlen))
+ return -EFAULT;
+
+
+ lock_sock(sk);
+ err = nf_getsockopt(sk, PF_INET6, optname, optval,
+ &len);
+ release_sock(sk);
+ if (err >= 0)
+ err = put_user(len, optlen);
+ }
+#endif
+ return err;
+}
+
+
+#ifdef CONFIG_COMPAT
+int compat_ipv6_getsockopt(struct sock *sk, int level, int optname,
+    char __user *optval, int __user *optlen)
+{
+ int err;
+
+

```

```

+ if (level == SOL_IP && sk->sk_type != SOCK_RAW) {
+   if (udp_prot.compat_getsockopt)
+     return udp_prot.compat_getsockopt(sk, level,
+     optname, optval, optlen);
+   else
+     return udp_prot.getsockopt(sk, level,
+     optname, optval, optlen);
+ }
+
+ if(level != SOL_IPV6)
+   return -ENOPROTOOPT;
+
+ err = do_ipv6_getsockopt(sk, level, optname, optval, optlen);
+#ifdef CONFIG_NETFILTER
+ /* we need to exclude all possible EINVALs except default case */
+ if (err == -ENOPROTOOPT && optname != IPV6_ADDRFORM &&
+   optname != MCAST_MSFILTER) {
+   int len;
+
+   if (get_user(len, optlen))
+     return -EFAULT;
+
+   lock_sock(sk);
+   err = compat_nf_getsockopt(sk, PF_INET6, optname, optval,
+   &len);
+   release_sock(sk);
+   if (err >= 0)
+     err = put_user(len, optlen);
+ }
+#endif
+ return err;
+}
+#endif
+
+ void __init ipv6_packet_init(void)
+ {
+   dev_add_pack(&ipv6_packet_type);
+--- ./net/ipv6/ipv6_syms.c.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./net/ipv6/ipv6_syms.c 2006-03-10 13:16:56.000000000 +0300
@@ -18,6 +18,10 @@ EXPORT_SYMBOL(ip6_route_output);
EXPORT_SYMBOL(addrconf_lock);
EXPORT_SYMBOL(ipv6_setsockopt);
EXPORT_SYMBOL(ipv6_getsockopt);
+#ifdef CONFIG_COMPAT
+EXPORT_SYMBOL(compat_ipv6_setsockopt);
+EXPORT_SYMBOL(compat_ipv6_getsockopt);
+#endif
EXPORT_SYMBOL(inet6_register_protosw);

```

```

EXPORT_SYMBOL(inet6_unregister_protosw);
EXPORT_SYMBOL(inet6_add_protocol);
--- ./net/ipv6/raw.c.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./net/ipv6/raw.c 2006-03-10 13:34:28.000000000 +0300
@@ -859,29 +859,12 @@ static int rawv6_geticmpfilter(struct sock
}

```

```

-static int rawv6_setsockopt(struct sock *sk, int level, int optname,
+static int do_rawv6_setsockopt(struct sock *sk, int level, int optname,
    char __user *optval, int optlen)

```

```

{
    struct raw6_sock *rp = raw6_sk(sk);
    int val;

- switch(level) {
- case SOL_RAW:
- break;
-
- case SOL_ICMPV6:
- if (inet_sk(sk)->num != IPPROTO_ICMPV6)
- return -EOPNOTSUPP;
- return rawv6_seticmpfilter(sk, level, optname, optval,
-    optlen);
- case SOL_IPV6:
- if (optname == IPV6_CHECKSUM)
- break;
- default:
- return ipv6_setsockopt(sk, level, optname, optval,
-    optlen);
- };
-
    if (get_user(val, (int __user *)optval))
        return -EFAULT;

```

```

@@ -906,12 +889,9 @@ static int rawv6_setsockopt(struct sock
}
}

```

```

-static int rawv6_getsockopt(struct sock *sk, int level, int optname,
-    char __user *optval, int __user *optlen)
+static int rawv6_setsockopt(struct sock *sk, int level, int optname,
+    char __user *optval, int optlen)
{
- struct raw6_sock *rp = raw6_sk(sk);
- int val, len;
-
    switch(level) {

```



```

+ int val, len;

    if (get_user(len,optlen))
        return -EFAULT;
@@ -953,6 +965,52 @@ static int rawv6_getsockopt(struct sock
    return 0;
}

+static int rawv6_getsockopt(struct sock *sk, int level, int optname,
+ char __user *optval, int __user *optlen)
+{
+ switch(level) {
+ case SOL_RAW:
+ break;
+
+ case SOL_ICMPV6:
+ if (inet_sk(sk)->num != IPPROTO_ICMPV6)
+ return -EOPNOTSUPP;
+ return rawv6_geticmphfilter(sk, level, optname, optval,
+ optlen);
+ case SOL_IPV6:
+ if (optname == IPV6_CHECKSUM)
+ break;
+ default:
+ return ipv6_getsockopt(sk, level, optname, optval,
+ optlen);
+ };
+ return do_rawv6_getsockopt(sk, level, optname, optval, optlen);
+}
+
+#ifdef CONFIG_COMPAT
+static int compat_rawv6_getsockopt(struct sock *sk, int level, int optname,
+ char __user *optval, int __user *optlen)
+{
+ switch(level) {
+ case SOL_RAW:
+ break;
+
+ case SOL_ICMPV6:
+ if (inet_sk(sk)->num != IPPROTO_ICMPV6)
+ return -EOPNOTSUPP;
+ return rawv6_geticmphfilter(sk, level, optname, optval,
+ optlen);
+ case SOL_IPV6:
+ if (optname == IPV6_CHECKSUM)
+ break;
+ default:
+ return compat_ipv6_getsockopt(sk, level,

```

```

+  optname, optval, optlen);
+ };
+ return do_rawv6_getsockopt(sk, level, optname, optval, optlen);
+}
+#endif
+
static int rawv6_ioctl(struct sock *sk, int cmd, unsigned long arg)
{
    switch(cmd) {
@@ -1008,6 +1066,10 @@ struct proto rawv6_prot = {
    .destroy = inet6_destroy_sock,
    .setsockopt = rawv6_setsockopt,
    .getsockopt = rawv6_getsockopt,
+#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_rawv6_setsockopt,
+ .compat_getsockopt = compat_rawv6_getsockopt,
+#endif
    .sendmsg = rawv6_sendmsg,
    .recvmsg = rawv6_recvmsg,
    .bind = rawv6_bind,
--- ./net/ipv6/tcp_ipv6.c.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./net/ipv6/tcp_ipv6.c 2006-03-10 13:19:49.000000000 +0300
@@ -1308,6 +1308,10 @@ static struct inet_connection_sock_af_op

    .setsockopt = ipv6_setsockopt,
    .getsockopt = ipv6_getsockopt,
+#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_ipv6_setsockopt,
+ .compat_getsockopt = compat_ipv6_getsockopt,
+#endif
    .addr2sockaddr = inet6_csk_addr2sockaddr,
    .sockaddr_len = sizeof(struct sockaddr_in6)
};
@@ -1327,6 +1331,10 @@ static struct inet_connection_sock_af_op

    .setsockopt = ipv6_setsockopt,
    .getsockopt = ipv6_getsockopt,
+#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_ipv6_setsockopt,
+ .compat_getsockopt = compat_ipv6_getsockopt,
+#endif
    .addr2sockaddr = inet6_csk_addr2sockaddr,
    .sockaddr_len = sizeof(struct sockaddr_in6)
};
@@ -1566,6 +1574,10 @@ struct proto tcpv6_prot = {
    .shutdown = tcp_shutdown,
    .setsockopt = tcp_setsockopt,
    .getsockopt = tcp_getsockopt,

```

```

+#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_tcp_setsockopt,
+ .compat_getsockopt = compat_tcp_getsockopt,
+#endif
. sendmsg = tcp_sendmsg,
. recvmsg = tcp_recvmsg,
. backlog_rcv = tcp_v6_do_rcv,
--- ./net/ipv6/udp.c.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./net/ipv6/udp.c 2006-03-10 13:26:47.000000000 +0300
@@ -880,16 +880,13 @@ static int udpv6_destroy_sock(struct sock
/*
 * Socket option code for UDP
 */
-static int udpv6_setsockopt(struct sock *sk, int level, int optname,
+static int do_udp6_setsockopt(struct sock *sk, int level, int optname,
    char __user *optval, int optlen)
{
    struct udp_sock *up = udp_sk(sk);
    int val;
    int err = 0;

- if (level != SOL_UDP)
- return ipv6_setsockopt(sk, level, optname, optval, optlen);
-
    if(optlen < sizeof(int))
        return -EINVAL;

@@ -927,15 +924,31 @@ static int udpv6_setsockopt(struct sock
    return err;
}

-static int udpv6_getsockopt(struct sock *sk, int level, int optname,
+static int udpv6_setsockopt(struct sock *sk, int level, int optname,
+    char __user *optval, int optlen)
+{
+ if (level != SOL_UDP)
+ return ipv6_setsockopt(sk, level, optname, optval, optlen);
+ return do_udp6_setsockopt(sk, level, optname, optval, optlen);
+}
+
+#ifdef CONFIG_COMPAT
+static int compat_udp6_setsockopt(struct sock *sk, int level, int optname,
+    char __user *optval, int optlen)
+{
+ if (level != SOL_UDP)
+ return compat_ipv6_setsockopt(sk, level,
+    optname, optval, optlen);
+ return do_udp6_setsockopt(sk, level, optname, optval, optlen);

```



```

+#endif
    .sendmsg = udpv6_sendmsg,
    .recvmsg = udpv6_recvmsg,
    .backlog_rcv = udpv6_queue_rcv_skb,
--- ./net/netfilter/nf_sockopt.c.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./net/netfilter/nf_sockopt.c 2006-03-10 13:46:12.000000000 +0300
@@ -131,3 +131,72 @@ int nf_getsockopt(struct sock *sk, int p
}
EXPORT_SYMBOL(nf_getsockopt);

+#ifdef CONFIG_COMPAT
+static int compat_nf_sockopt(struct sock *sk, int pf, int val,
+    char __user *opt, int *len, int get)
+{
+ struct list_head *i;
+ struct nf_sockopt_ops *ops;
+ int ret;
+
+ if (mutex_lock_interruptible(&nf_sockopt_mutex) != 0)
+ return -EINTR;
+
+ list_for_each(i, &nf_sockopts) {
+ ops = (struct nf_sockopt_ops *)i;
+ if (ops->pf == pf) {
+ if (get) {
+ if (val >= ops->get_optmin
+     && val < ops->get_optmax) {
+ ops->use++;
+ mutex_unlock(&nf_sockopt_mutex);
+ if (ops->compat_get)
+ ret = ops->compat_get(sk,
+     val, opt, len);
+ else
+ ret = ops->get(sk,
+     val, opt, len);
+ goto out;
+ }
+ } else {
+ if (val >= ops->set_optmin
+     && val < ops->set_optmax) {
+ ops->use++;
+ mutex_unlock(&nf_sockopt_mutex);
+ if (ops->compat_set)
+ ret = ops->compat_set(sk,
+     val, opt, *len);
+ else
+ ret = ops->set(sk,
+     val, opt, *len);

```

```

+ goto out;
+ }
+ }
+ }
+ }
+ mutex_unlock(&nf_sockopt_mutex);
+ return -ENOPROTOOPT;
+
+ out:
+ mutex_lock(&nf_sockopt_mutex);
+ ops->use--;
+ if (ops->cleanup_task)
+ wake_up_process(ops->cleanup_task);
+ mutex_unlock(&nf_sockopt_mutex);
+ return ret;
+}
+
+int compat_nf_setsockopt(struct sock *sk, int pf,
+ int val, char __user *opt, int len)
+{
+ return compat_nf_sockopt(sk, pf, val, opt, &len, 0);
+}
+EXPORT_SYMBOL(compat_nf_setsockopt);
+
+int compat_nf_getsockopt(struct sock *sk, int pf,
+ int val, char __user *opt, int *len)
+{
+ return compat_nf_sockopt(sk, pf, val, opt, len, 1);
+}
+EXPORT_SYMBOL(compat_nf_getsockopt);
+endif
--- ./net/sctp/ipv6.c.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./net/sctp/ipv6.c 2006-03-10 13:21:06.000000000 +0300
@@ -875,6 +875,10 @@ static const struct proto_ops inet6_seqp
 .shutdown = inet_shutdown,
 .setsockopt = sock_common_setsockopt,
 .getsockopt = sock_common_getsockopt,
+#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_sock_common_setsockopt,
+ .compat_getsockopt = compat_sock_common_getsockopt,
+#endif
 .sendmsg = inet_sendmsg,
 .recvmsg = sock_common_recvmsg,
 .mmap = sock_no_mmap,
@@ -914,6 +918,10 @@ static struct sctp_af sctp_ipv6_specific
 .sctp_xmit = sctp_v6_xmit,
 .setsockopt = ipv6_setsockopt,
 .getsockopt = ipv6_getsockopt,

```

```

+#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_ipv6_setsockopt,
+ .compat_getsockopt = compat_ipv6_getsockopt,
+#endif
. get_dst = sctp_v6_get_dst,
. get_saddr = sctp_v6_get_saddr,
. copy_addrlist = sctp_v6_copy_addrlist,
--- ./net/sctp/protocol.c.compat 2006-03-10 11:58:11.000000000 +0300
+++ ./net/sctp/protocol.c 2006-03-10 12:24:11.000000000 +0300
@@ -845,6 +845,10 @@ static const struct proto_ops inet_seqpa
. shutdown = inet_shutdown, /* Looks harmless. */
. setsockopt = sock_common_setsockopt, /* IP_SOL IP_OPTION is a problem. */
. getsockopt = sock_common_getsockopt,
+#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_sock_common_setsockopt,
+ .compat_getsockopt = compat_sock_common_getsockopt,
+#endif
. sendmsg = inet_sendmsg,
. recvmsg = sock_common_recvmsg,
. mmap = sock_no_mmap,
@@ -883,6 +887,10 @@ static struct sctp_af sctp_ipv4_specific
. sctp_xmit = sctp_v4_xmit,
. setsockopt = ip_setsockopt,
. getsockopt = ip_getsockopt,
+#ifdef CONFIG_COMPAT
+ .compat_setsockopt = compat_ip_setsockopt,
+ .compat_getsockopt = compat_ip_getsockopt,
+#endif
. get_dst = sctp_v4_get_dst,
. get_saddr = sctp_v4_get_saddr,
. copy_addrlist = sctp_v4_copy_addrlist,

```

## File Attachments

1) [diff-ms-sockopts-compat-20060310](#), downloaded 674 times

---