author	Wang Yufen <wangyufen@huawei.com>	2022-11-01 09:31:36 +0800
committer	Greg Kroah-Hartman <gregkh@linuxfoundation.org>	2022-11-16 09:58:14 +0100
commit	95adbd2ac8de82e43fd6b347e7e1b47f74dc1abb (patch)	
tree	2863dc1c1b3f1d85d7e944ad0c7672ce8abf60b6	
parent	06615967d4889b08b19ff3dda96e8b131282f73d (diff)	
download	linux-95adbd2ac8de82e43fd6b347e7e1b47f74dc1abb.tar.gz	

bpf, sockmap: Fix the sk->sk_forward_alloc warning of sk_stream_kill_queues

```
[ Upstream commit 8ec95b94716a1e4d126edc3fb2bc426a717e2dba ]

When running `test_sockmap` selftests, the following warning appears:

  WARNING: CPU: 2 PID: 197 at net/core/stream.c:205 sk_stream_kill_queues+0xd3/0xf0
  Call Trace:
  <TASK>
  inet_csk_destroy_sock+0x55/0x110
  tcp_rcv_state_process+0xd28/0x1380
  ? tcp_v4_do_rcv+0x77/0x2c0
  tcp_v4_do_rcv+0x77/0x2c0
  __release_sock+0x106/0x130
  __tcp_close+0x1a7/0x4e0
  tcp_close+0x20/0x70
  inet_release+0x3c/0x80
  __sock_release+0x3a/0xb0
  sock_close+0x14/0x20
  __fput+0xa3/0x260
  task_work_run+0x59/0xb0
  exit_to_user_mode_prepare+0x1b3/0x1c0
  syscall_exit_to_user_mode+0x19/0x50
  do_syscall_64+0x48/0x90
  entry_SYSCALL_64_after_hwframe+0x44/0xae

The root case is in commit 84472b436e76 ("bpf, sockmap: Fix more uncharged
while msg has more_data"), where I used msg->sg.size to replace the tosend,
causing breakage:

  if (msg->apply_bytes && msg->apply_bytes < tosend)
    tosend = psock->apply_bytes;

Fixes: 84472b436e76 ("bpf, sockmap: Fix more uncharged while msg has more_data")
Reported-by: Jakub Sitnicki <jakub@cloudflare.com>
Signed-off-by: Wang Yufen <wangyufen@huawei.com>
Signed-off-by: Daniel Borkmann <daniel@iogearbox.net>
Acked-by: John Fastabend <john.fastabend@gmail.com>
Acked-by: Jakub Sitnicki <jakub@cloudflare.com>
Link: https://lore.kernel.org/bpf/1667266296-8794-1-git-send-email-wangyufen@huawei.com
Signed-off-by: Sasha Levin <sashal@kernel.org>
```

Diffstat

-rw-r--r-- net/ipv4/tcp_bpf.c 8

1 files changed, 5 insertions, 3 deletions

```
diff --git a/net/ipv4/tcp_bpf.c b/net/ipv4/tcp_bpf.c
index 2c597a4e429aba..72892ebe96074a 100644
--- a/net/ipv4/tcp_bpf.c
+++ b/net/ipv4/tcp_bpf.c
@@ -279,7 +279,7 @@ static int tcp_bpf_send_verdict(struct sock *sk, struct sk_psock *psock,
 {
        bool cork = false, enospc = sk_msg_full(msg);
        struct sock *sk_redir;
-       u32 tosend, delta = 0;
+       u32 tosend, origsize, sent, delta = 0;
        u32 eval = __SK_NONE;
        int ret;

@@ -334,10 +334,12 @@ more_data:
                        cork = true;
                        psock->cork = NULL;
                }
-               sk_msg_return(sk, msg, msg->sg.size);
+               sk_msg_return(sk, msg, tosend);
                release_sock(sk);

+               origsize = msg->sg.size;
                ret = tcp_bpf_sendmsg_redir(sk_redir, msg, tosend, flags);
+               sent = origsize - msg->sg.size;

                if (eval == __SK_REDIRECT)
                        sock_put(sk_redir);
@@ -376,7 +378,7 @@ more_data:
                    msg->sg.data[msg->sg.start].page_link &&
                    msg->sg.data[msg->sg.start].length) {
                        if (eval == __SK_REDIRECT)
-                               sk_mem_charge(sk, msg->sg.size);
+                               sk_mem_charge(sk, tosend - sent);
                        goto more_data;
                }
        }
```