author	Breno Leitao <leitao@debian.org>	2023-03-08 11:07:45 -0800
committer	Greg Kroah-Hartman <gregkh@linuxfoundation.org>	2023-03-22 13:37:45 +0100
commit	7613cde8c0c1f02a7ec2e1d536c01b65b135fc1c (patch)	
tree	ad000cdc8d8b0efcdade86da5bf51242f01cd40e	
parent	2c3a0f239bffe376ca02b0313b2f1e0f487795d2 (diff)	
download	linux-7613cde8c0c1f02a7ec2e1d536c01b65b135fc1c.tar.gz	

tcp: tcp_make_synack() can be called from process context

```
[ Upstream commit bced3f7db95ff2e6ca29dc4d1c9751ab5e736a09 ]

tcp_rtx_synack() now could be called in process context as explained in
0a375c822497 ("tcp: tcp_rtx_synack() can be called from process
context").

tcp_rtx_synack() might call tcp_make_synack(), which will touch per-CPU
variables with preemption enabled. This causes the following BUG:

    BUG: using __this_cpu_add() in preemptible [00000000] code: ThriftIO1/5464
    caller is tcp_make_synack+0x841/0xac0
    Call Trace:
     <TASK>
     dump_stack_lvl+0x10d/0x1a0
     check_preemption_disabled+0x104/0x110
     tcp_make_synack+0x841/0xac0
     tcp_v6_send_synack+0x5c/0x450
     tcp_rtx_synack+0xeb/0x1f0
     inet_rtx_syn_ack+0x34/0x60
     tcp_check_req+0x3af/0x9e0
     tcp_rcv_state_process+0x59b/0x2030
     tcp_v6_do_rcv+0x5f5/0x700
     release_sock+0x3a/0xf0
     tcp_sendmsg+0x33/0x40
     ____sys_sendmsg+0x2f2/0x490
     __sys_sendmsg+0x184/0x230
     do_syscall_64+0x3d/0x90

Avoid calling __TCP_INC_STATS() with will touch per-cpu variables. Use
TCP_INC_STATS() which is safe to be called from context switch.

Fixes: 8336886f786f ("tcp: TCP Fast Open Server - support TFO listeners")
Signed-off-by: Breno Leitao <leitao@debian.org>
Reviewed-by: Eric Dumazet <edumazet@google.com>
Link: https://lore.kernel.org/r/20230308190745.780221-1-leitao@debian.org
Signed-off-by: Jakub Kicinski <kuba@kernel.org>
Signed-off-by: Sasha Levin <sashal@kernel.org>
```

Diffstat

-rw-r--r-- net/ipv4/tcp_output.c 2

1 files changed, 1 insertions, 1 deletions

```
diff --git a/net/ipv4/tcp_output.c b/net/ipv4/tcp_output.c
index 71d01cf3c13eb4..ba839e441450f1 100644
--- a/net/ipv4/tcp_output.c
+++ b/net/ipv4/tcp_output.c
@@ -3605,7 +3605,7 @@ struct sk_buff *tcp_make_synack(const struct sock *sk, struct dst_entry *dst,
        th->window = htons(min(req->rsk_rcv_wnd, 65535U));
        tcp_options_write(th, NULL, &opts);
        th->doff = (tcp_header_size >> 2);
-       __TCP_INC_STATS(sock_net(sk), TCP_MIB_OUTSEGS);
+       TCP_INC_STATS(sock_net(sk), TCP_MIB_OUTSEGS);

 #ifdef CONFIG_TCP_MD5SIG
        /* Okay, we have all we need - do the md5 hash if needed */
```