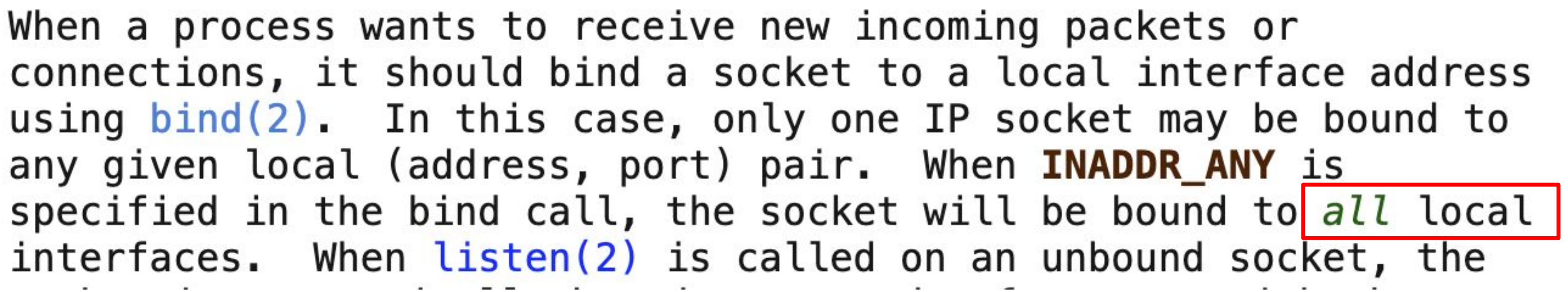
# (non-)overlapping bind (any vs tproxy)
# Misc: WireGuard, BIG TCP

Daniel Borkmann

ISOVALENT
now part of CISCO

## bind & INADDR_ANY

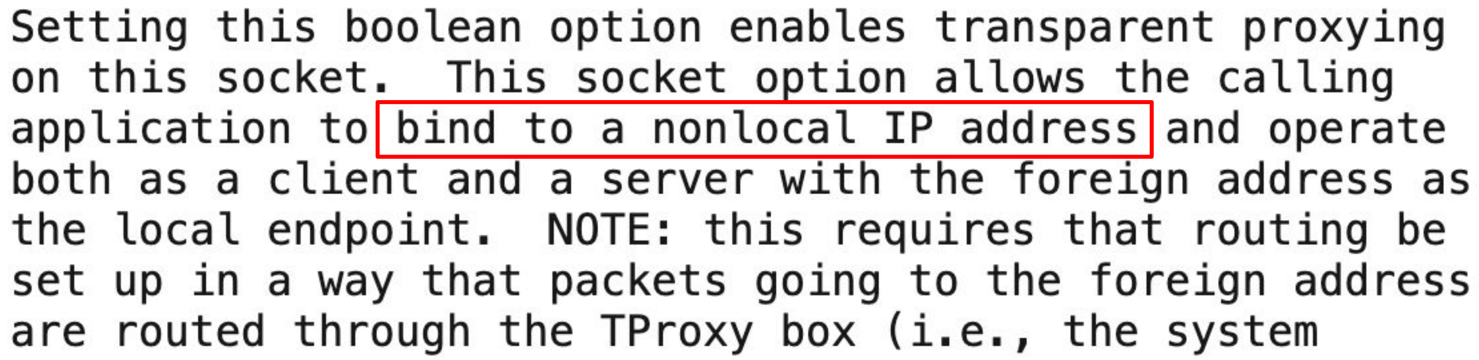
man ip(7) :



When a process wants to receive new incoming packets or connections, it should bind a socket to a local interface address using bind(2). In this case, only one IP socket may be bound to any given local (address, port) pair. When **INADDR_ANY** is specified in the bind call, the socket will be bound to *all* local interfaces. When listen(2) is called on an unbound socket, the

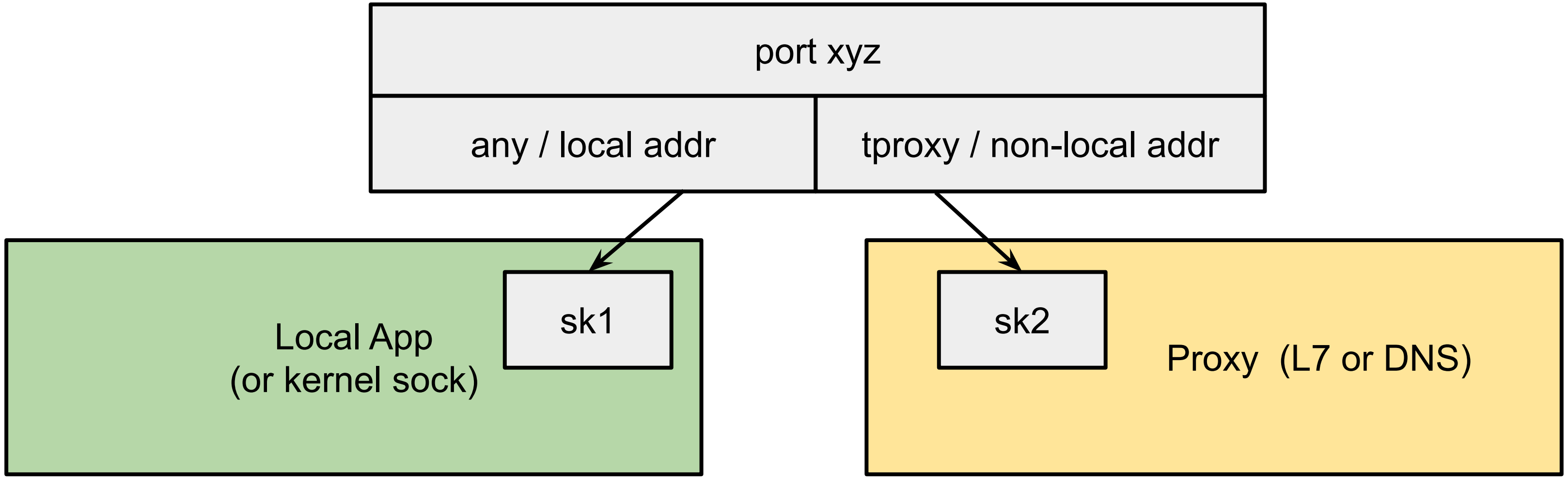
## bind & IP_TRANSPARENT

man ip(7) :



Setting this boolean option enables transparent proxying on this socket. This socket option allows the calling application to bind to a nonlocal IP address and operate both as a client and a server with the foreign address as the local endpoint. NOTE: this requires that routing be set up in a way that packets going to the foreign address are routed through the TProxy box (i.e., the system

## (non-)overlapping binds on same port?



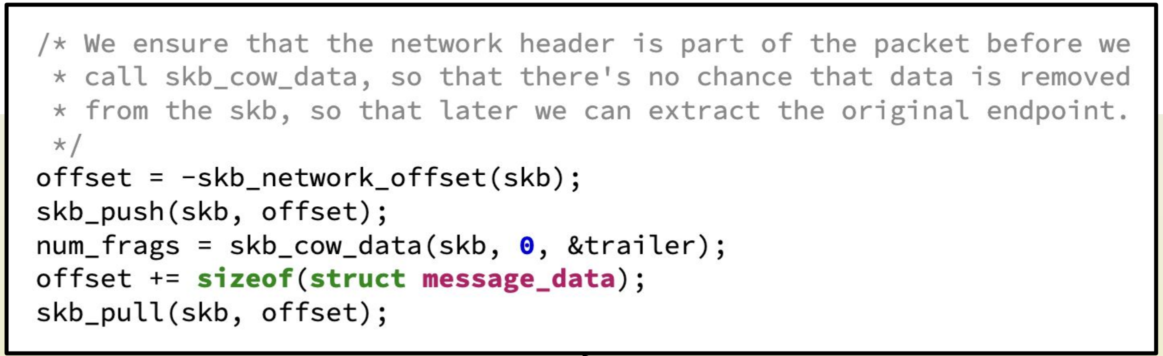
## (non-)overlapping binds on same port?

- inet_bind() already checks if the given address is a local address or not
- Opt-in via INET_FLAGS_BIND_OVERLAP_ANY on a transparent socket
- Allows transparent proxies to use of the same port numbers with a non-local address as is being used for an ANY listener in the same networking namespace
- IPv4 PoC
- IPv6 PoC

## Misc: WireGuard

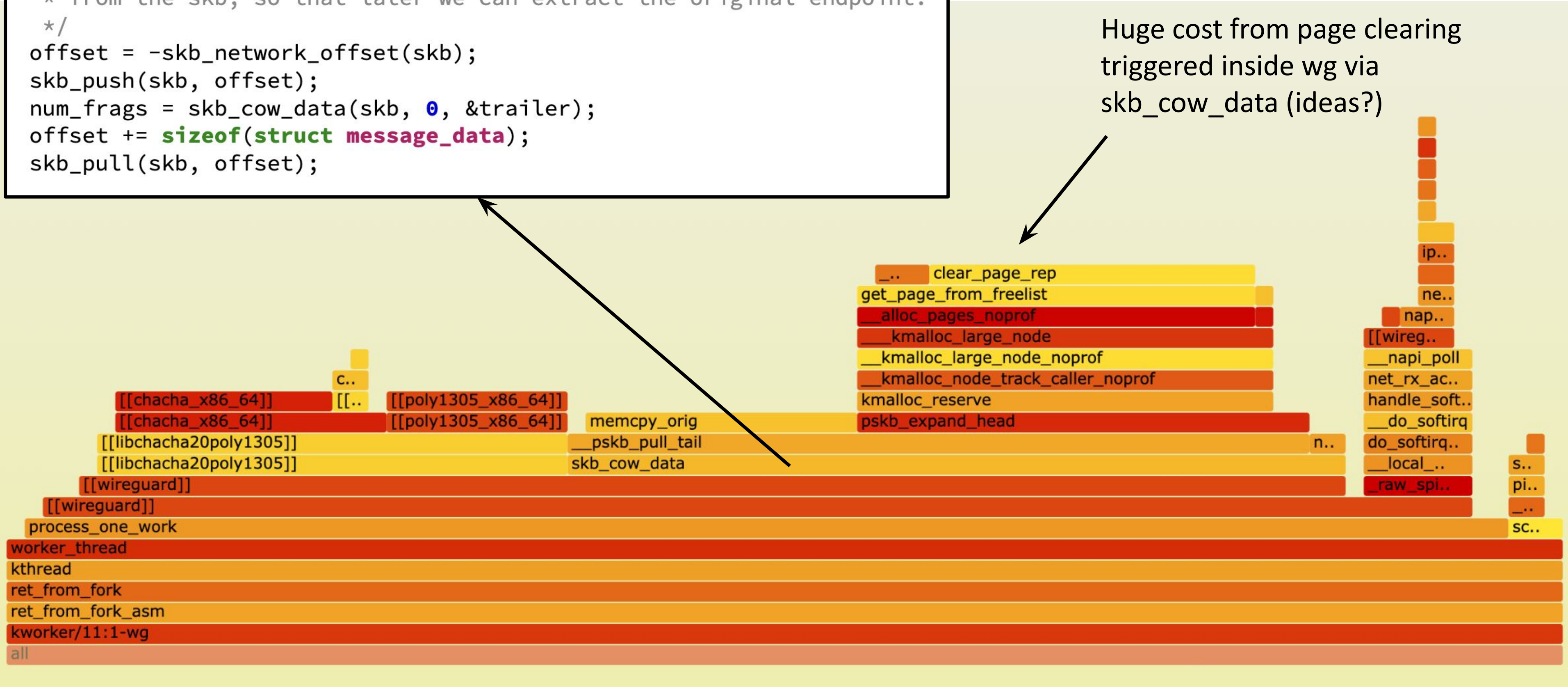
- Finding from flamegraph when running netperf over wireguard ifaces
- clear_page_rep() stood out as being even more expensive than the crypto ops
- Is there a way we could avoid it?

## wg: decrypt_packet() :



```
/* We ensure that the network header is part of the packet before we
 * call skb_cow_data, so that there's no chance that data is removed
 * from the skb, so that later we can extract the original endpoint.
 */
offset = -skb_network_offset(skb);
skb_push(skb, offset);
num_frags = skb_cow_data(skb, 0, &trailer);
offset += sizeof(struct message_data);
skb_pull(skb, offset);
```

Huge cost from page clearing triggered inside wg via skb_cow_data (ideas?)



# Misc: BIG TCP

- Regression when only enabling BIG TCP for IPv4 but not for IPv6
- Reported by users when having Cilium + IPv4 BIG TCP enabled
- Planning to send a fix for gso_features_check(): depending on protocol either use READ_ONCE(dev->gso_max_size) or READ_ONCE(dev->gso_ipv4_max_size)
- Some locations respect the above, but others not

Can be common helper:

```
        gro_max_size = skb->protocol == htons(ETH_P_IPV4) ?
                        READ_ONCE(skb->dev->gro_ipv4_max_size) :
                        READ_ONCE(skb->dev->gro_max_size);
```