Zero-Copy Socket Splicing

Alexander Bluhm

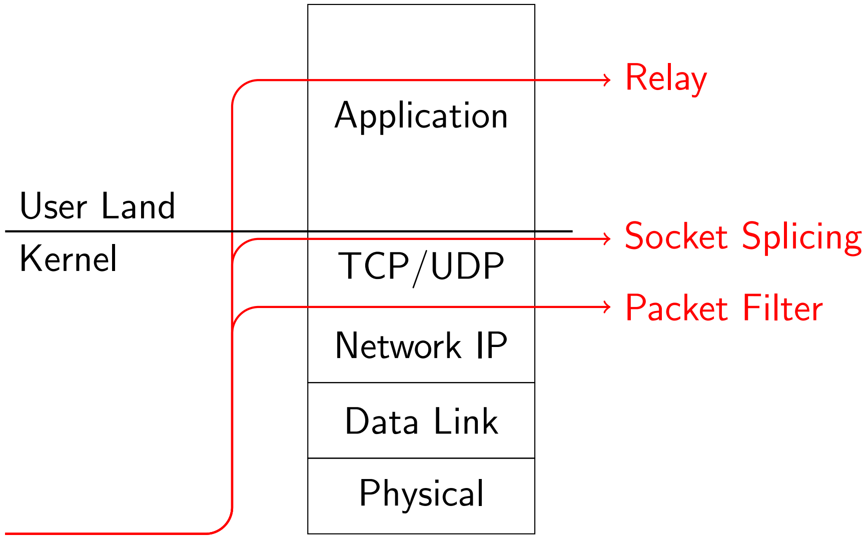
bluhm@openbsd.org

Sunday, 29. September 2013

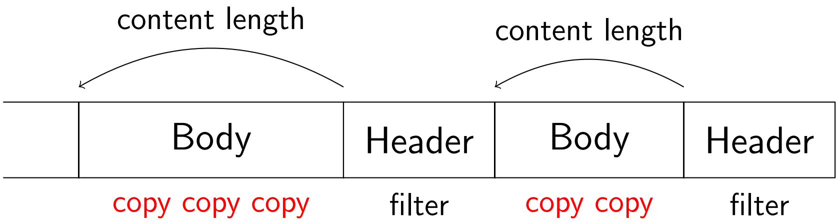
Agenda

Agenda

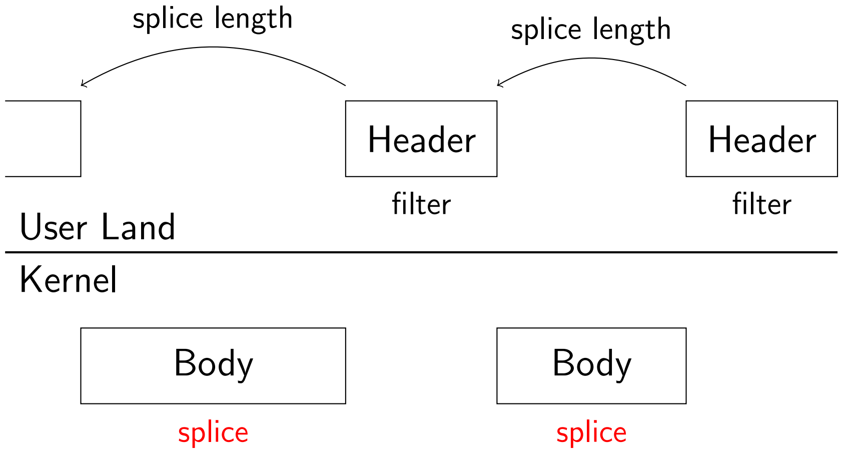
Application Level Gateway



Persistent HTTP Filtering

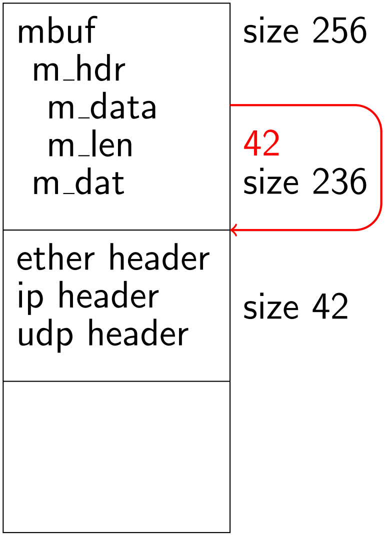


HTTP Socket Splicing

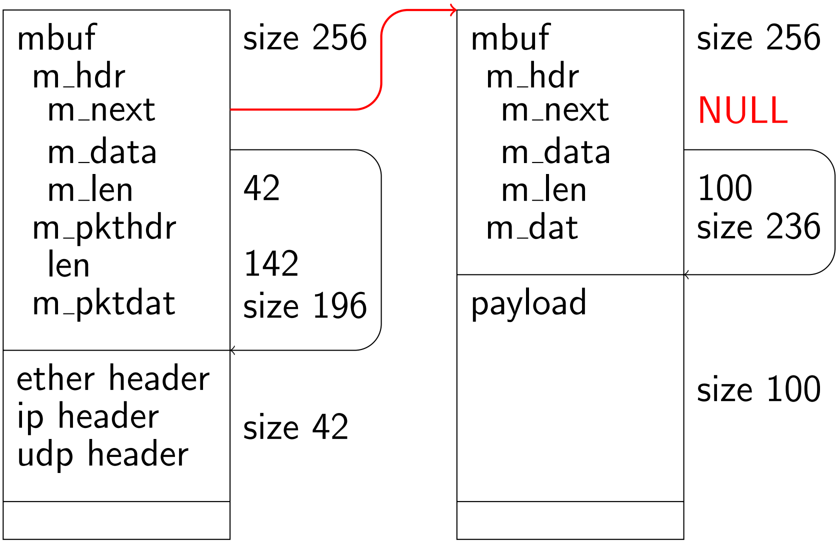


Agenda

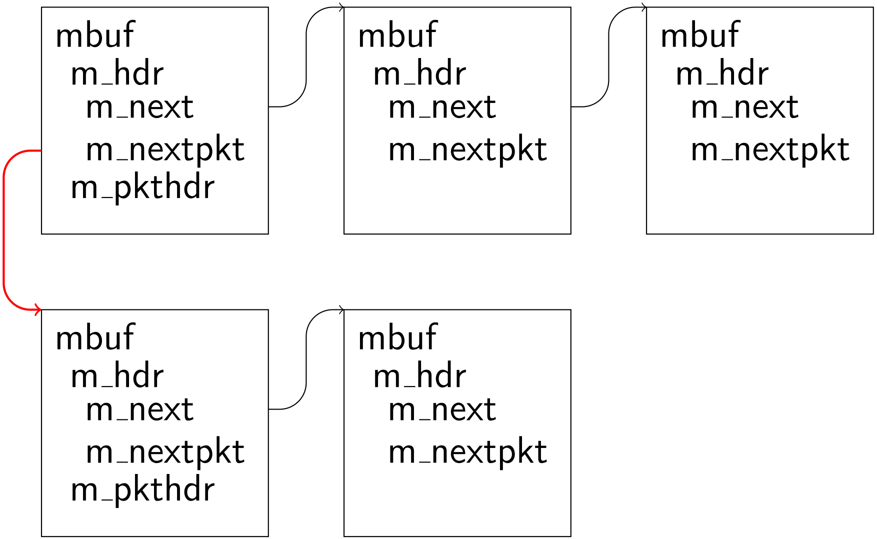
MBuf Data


mbuf
m_hdr
m_data
m_len
m_dat
size 256
42
size 236
ether header
ip header
udp header
size 42

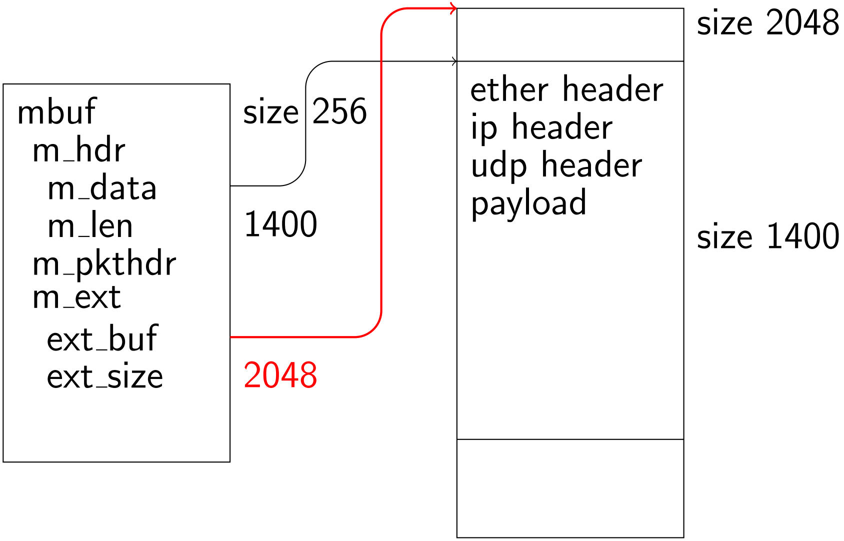
MBuf Data Chaining



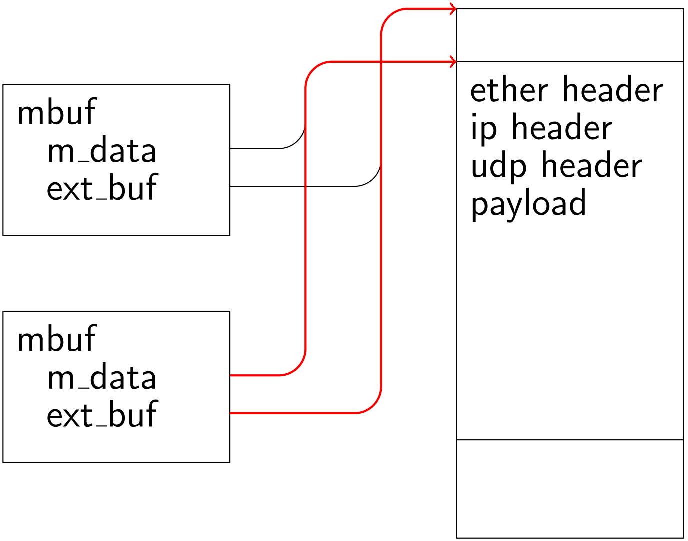
MBuf Packet Chaining



MBuf Cluster

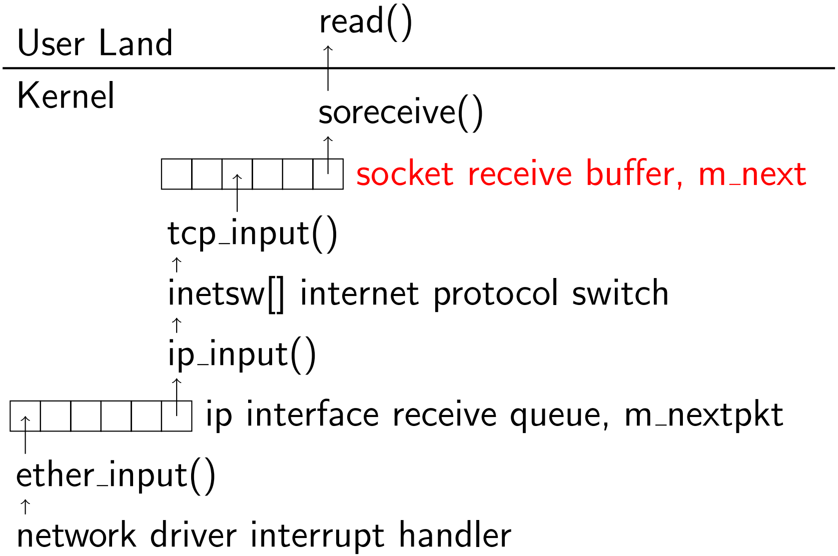


MBuf Cluster Copy

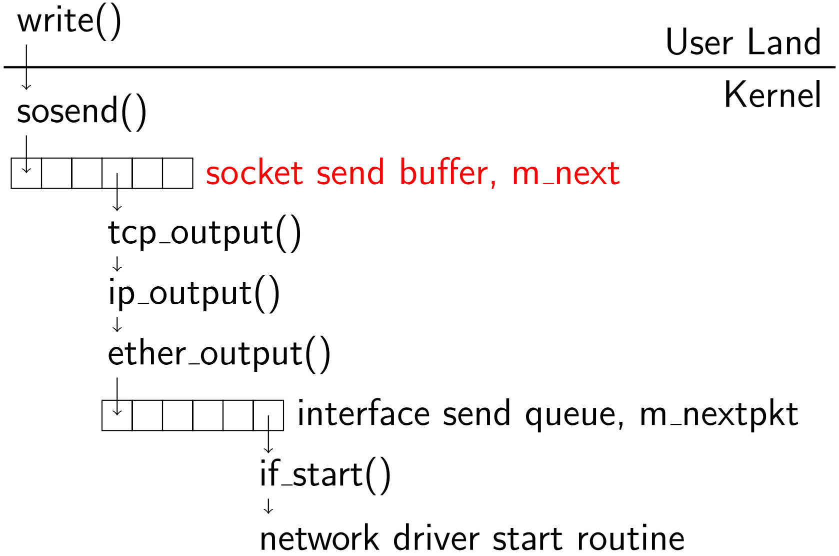


Agenda

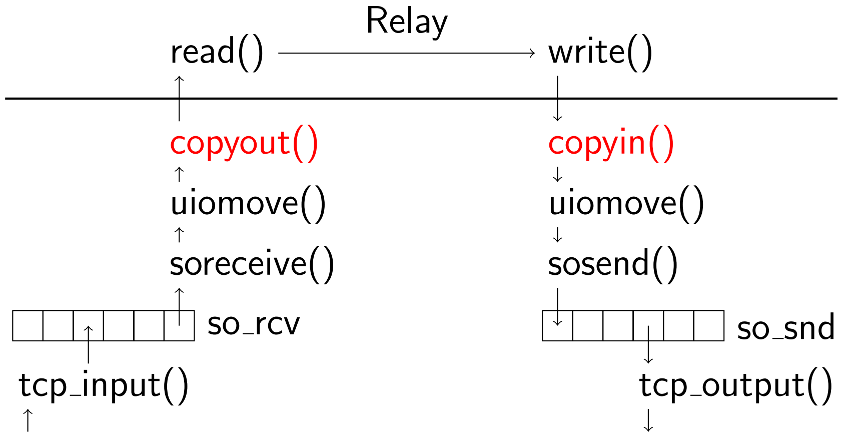
Packet Input


read()
User Land
Kernel
soreceive()
socket receive buffer, m_next
tcp_input()
inetsw[] internet protocol switch
ip_input()
ip interface receive queue, m_nextpkt
ether_input()
network driver interrupt handler

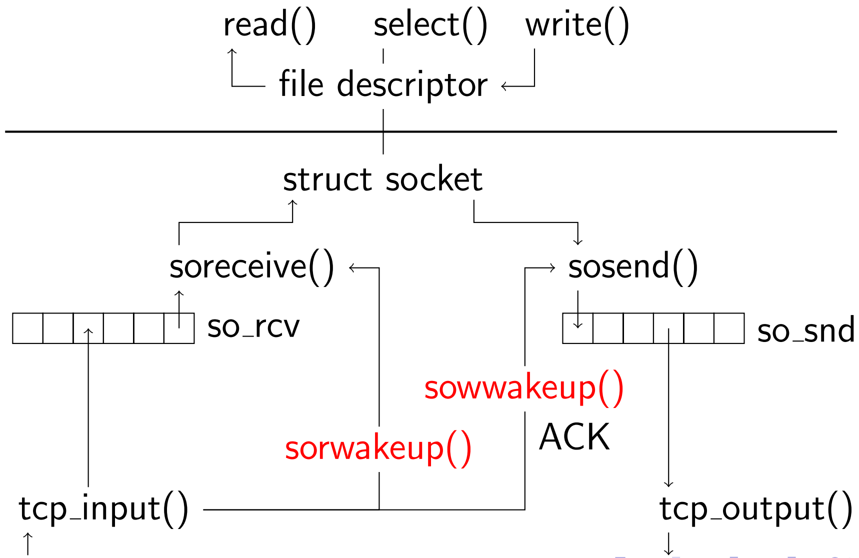
Packet Output


write()
User Land
Kernel
sosend()
socket send buffer, m_next
tcp_output()
ip_output()
ether_output()
interface send queue, m_nextpkt
if_start()
network driver start routine

Data Copy

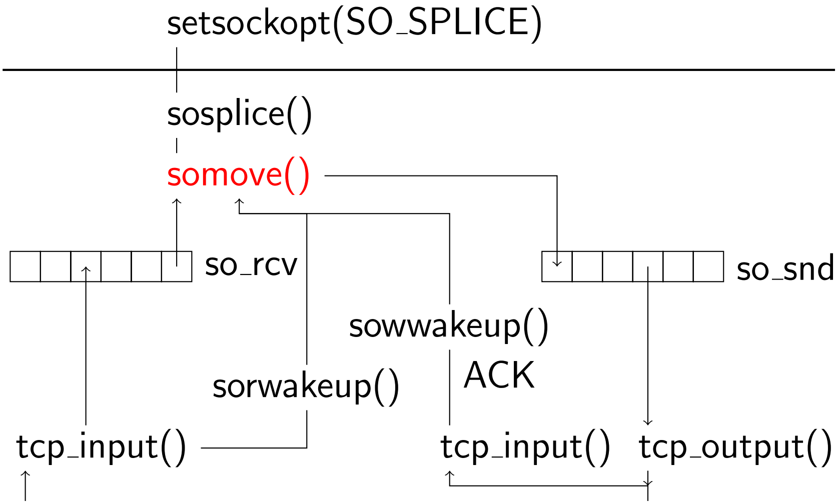

Relay
read()
write()
copyout()
copyin()
uiomove()
uiomove()
soreceive()
sosend()
so_rcv
so_snd
tcp_input()
tcp_output()

Process Wakeup

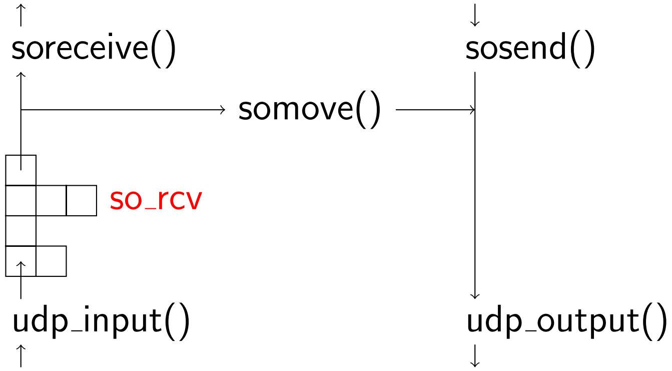


Agenda

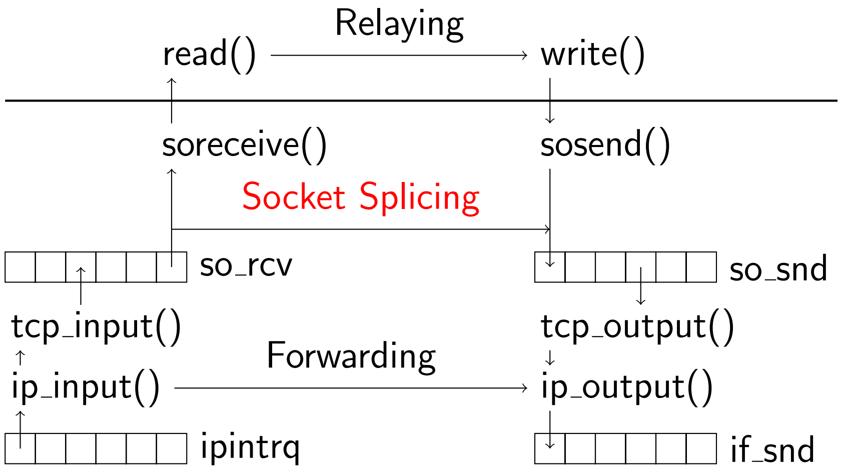
Socket Splicing



UDP Sockets



Layer


Relaying
read()
write()
soreceive()
sosend()
Socket Splicing
so_rcv
so_snd
tcp_input()
tcp_output()
Forwarding
ip_input()
ip_output()
ipintrq
if_snd

Agenda

Simple API

- Begin splicing from source to drain
  setsockopt(source_fd, SO_SPLICE, drain_fd)
- Stop splicing
  setsockopt(source_fd, SO_SPLICE, -1)
- Get spliced data length
  getsockopt(source_fd, SO_SPLICE, &length)

Extended API

```
struct splice {
  int     sp_fd;                 /* drain */
  off_t   sp_max;                /* maximum */
  struct timeval sp_idle;        /* timeout */
};
```

setsockopt(source_fd, SO_SPLICE, &splice)

Properties

- Splicing is unidirectional
- Invoke it twice for bidirectional splicing
- Process can turn it on and off
- Works for TCP and UDP
- Can mix IPv4 and IPv6 sockets

Unsplice

- Dissolve socket splicing manually
- read(2) or select(2) from the source
- EOF source socket shutdown
- EPIPE drain socket error
- EFBIG maximum data length
- ETIMEDOUT idle timeout

Agenda

Struct Socket

```
struct socket {
    ...
    struct  socket *so_splice;
    struct  socket *so_spliceback;
    off_t   so_splicelen;
    off_t   so_splicemax;
    struct  timeval so_idletv;
    struct  timeout so_idleto;
    ...
};
```

sosplice(9)

- Protocol must match
- Sockets must be connected
- Double link sockets
- Move existing data

somove(9)

- Check for errors
- Check for space
- Handle maximum
- Handle out of band data
- Move socket buffer data

sounsplice()

- Manual unsplice
- Cannot receive
- Cannot send
- Maximum
- Timeout
- Socket closed

sorwakeup() sowwakeup()

- Called from tcp_input()
- Source calls sorwakeup()
- Drain calls sowwakeup()
- Both invoke somove(9)

Agenda

Relayd

- Plain TCP connections
- HTTP connections
- Filter persistent HTTP
- HTTP Chunking

Tests

- /usr/src/regress/sys/kern/sosplice/
- 15 API tests
- 18 UDP tests
- 76 TCP tests
- perf/relay.c simple example
- BSD::Socket::Splice Perl API
- 28 relayd tests

Performance

- Factor 1 or 2 for TCP
- Factor 6 or 8 for UDP

Documentation

- Manpage setsockopt(2) SO_SPLICE
- Manpage sosplice(9) somove(9)

Questions

?