

# io\_uring

Status Update within Samba

Stefan Metzmacher <metze@samba.org>

Samba Team / SerNet

2023-09-20

https://samba.org/~metze/presentations/2023/SDC/

#### **Topics**



- ▶ What is io-uring?
- ▶ io-uring for Samba
- ▶ Performance research, prototyping and ideas
- ► The road to upstream
- ► Future Improvements
- ► Questions? Feedback!

## Last Status Updates (SDC 2020/2021 - SambaX

- ▶ I gave a similar talk at the storage developer conference 2020:
  - See https://samba.org/~metze/presentations/2020/SDC/
  - ▶ It explains the milestones and design up to Samba 4.13 (in detail)
- ▶ I gave a similar talk at the storage developer conference 2021:
  - See https://samba.org/~metze/presentations/2021/SDC/
  - It explains the milestones and updates up to Samba 4.15 (in detail)
- ▶ I gave a similar talk at the SambaXP conference 2023:
  - See https://samba.org/~metze/presentations/2023/SambaXP/
  - It explains the milestones and updates up to Samba 4.19 (in detail)



- Linux 5.1 introduced a new scalable AIO infrastructure
  - It's designed to avoid syscalls as much as possible
  - kernel and userspace share mmap'ed rings:
    - submission queue (SQ) ring buffer
    - completion queue (CQ) ring buffer
  - ► See "Ringing in a new asynchronous I/O API" on LWN.NET
- ► This can be nicely integrated with our async tevent model
  - It may delegate work to kernel threads
  - It seems to perform better compared to our userspace threadpool
  - It can also inline non-blocking operations





### io-uring for Samba (Part 1)



- ► IORING\_OP\_READV, IORING\_OP\_WRITEV and IORING\_OP\_FSYNC
- Supports buffered and direct io
- ► IORING\_OP\_FSETXATTR, IORING\_OP\_FGETXATTR (from 5.19)
- ► IORING\_OP\_GETDENTS, under discussion, but seems to be tricky
- ► IORING\_OP\_FADVISE (from 5.6)
- ▶ Path based syscalls with async impersonation (from 5.6)
  - ► IORING\_OP\_OPENAT2, IORING\_OP\_STATX
  - ▶ Using IORING\_REGISTER\_PERSONALITY for impersonation
  - ► IORING\_OP\_UNLINKAT, IORING\_OP\_RENAMEAT (from 5.10)
  - ► IORING\_OP\_MKDIRAT, IORING\_OP\_SYMLINKAT, IORING\_OP\_LINKAT (from 5.15)
  - ► IORING\_OP\_SETXATTR, IORING\_OP\_GETXATTR (from 5.19)



- ▶ Between userspace and socket (and also filesystem) (from 5.8)
  - ► IORING\_OP\_SENDMSG, IORING\_OP\_RECVMSG
  - ▶ Improved MSG\_WAITALL support (5.12, backported to 5.11, 5.10)
  - ▶ Maybe using IOSQE\_ASYNC in order to avoid inline memcpy
  - ▶ IORING\_OP\_SPLICE, IORING\_OP\_TEE
  - ► IORING\_OP\_SENDMSG\_ZC, zero copy with an extra completion (from 6.1)
  - IORING\_OP\_GET\_BUF, under discussion to replace IORING\_OP\_SPLICE





### vfs\_io\_uring in Samba 4.12 (2020)



- ▶ With Samba 4.12 we added "io\_uring" vfs module
  - ► For now it only implements SMB\_VFS\_PREAD,PWRITE,FSYNC\_SEND/RECV
  - lt has less overhead than our pthreadpool default implementations
  - ▶ I was able to speed up a smbclient 'get largefile /dev/null'
    - Using against smbd on loopback
    - ► The speed changes from 2.2GBytes/s to 2.7GBytes/s
- ▶ The improvement only happens by avoiding context switches
  - But the data copying still happens:
    - ► From/to a userspace buffer to/from the filesystem/page cache
  - ▶ The data path between userspace and socket is completely unchanged
  - For both cases the cpu is mostly busy with memcpy

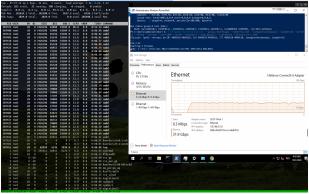


- ▶ In October 2020 I was able to do some performance research
  - With 100GBit/s interfaces and two NUMA nodes per server.
- ▶ At that time I focussed on the SMB2 Read performance only
  - We had limited time on the given hardware
  - We mainly tested with fio.exe on a Windows client
  - Linux kernel 5.8.12 on the server
- More verbose details can be found here:
  - https://lists.samba.org/archive/samba-technical/2020-October/135856.html



## Performance with MultiChannel, sendmsg()





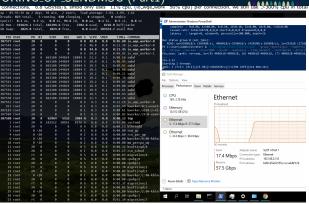
\$SD@ SAMBA<sup>†</sup>

Stefan Metzmacher

io uring (9/21)

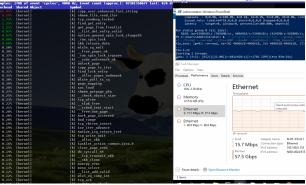
# SerNet !!!!!

### IORING\_OP\_SENDMSG (Part1)



### IORING\_OP\_SENDMSG (Part2)





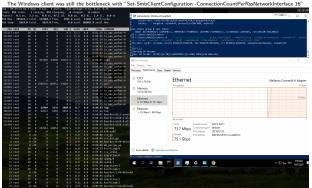
**\$SD**@

Stefan Metzmacher

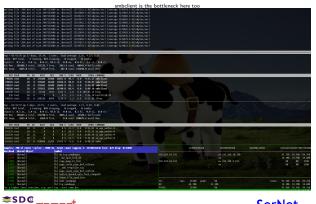
io uring (11/21)

SerNet E (Part1)

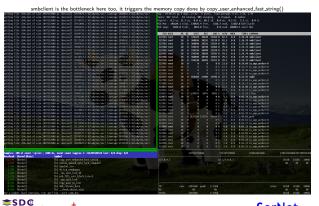
# IORING\_OP\_SENDMSG + IORING\_OP\_SPLICE (1944.1)



# smbclient IORING\_OP\_SENDMSG/SPLICE (network)



SAMBA<sup>†</sup> Stefan Metzmacher io uring (13/21) smbclient IORING\_OP\_SENDMSG/SPLICE (loopback)



### More loopback testing on brand new hardware



- Recently I re-did the loopback read tests IORING\_OP\_SENDMSG/SPLICE (from /dev/shm/)
  - 1 connection, ~10-13 GBytes/s, smbd 7% cpu, with 4 iou-wrk threads at 7%-50% cpu.
  - 4 connections, 24-30 GBytes/s, smbd 18% cpu, with 16 iou-wrk threads at 3%-35% cpu.
- ▶ I also implemented SMB2 writes with IORING\_OP\_RECVMSG/SPLICE (tested to /dev/null)
  - ▶ 1 connection, ~7-8 GBytes/s, smbd 5% cpu, with 3 io-wrk threads at 1%-20% cpu.
  - ▶ 4 connections, ~10 GBytes/s, smbd 15% cpu, with 12 io-wrk threads at 1%-20% cpu.
- ▶ I tested with a Linux Kernel 5.13
  - In both cases the bottleneck is clearly on the smbclient side
  - We could apply similar changes to smbclient and add true multichannel support
  - It seems that the filesystem->pipe->socket path is much better optimized

\$SD@ <u>SAMBA</u>

tefan Metzmache

io uring (15/21)

SerNet

### The road to upstream (TEVENT\_FD\_ERROR)

- ▶ We need support for TEVENT\_FD\_ERROR in order to monitor errors
  - When using IORING\_OP\_SEND,RECVMSG we still want to notice errors
  - ► This is the main merge request:
  - https://gitlab.com/samba-team/samba/-/merge\_requests/2793
  - ► This merge request converts Samba to use TEVENT\_FD\_ERROR:
  - https://gitlab.com/samba-team/samba/-/merge\_requests/2885
  - ► (It also simplifies other places in the code without io\_uring)



SerNet

### The road to upstream (samba\_io\_uring abstraction

#### API glue to tevent:

- samba\_io\_uring abstraction factored out of vfs\_io\_uring:
  - samba\_io\_uring\_ev\_hybrid tevent backend (glued on epoll backend)
  - It means every layer getting the tevent\_context can use io\_uring
  - No #ifdef's just checking if the required features are available



Stefan Metzmache

io uring (17/21)

SerNet Mizy III

### The road to upstream (samba\_io\_uring abstraction

generic submission/completion api:

```
| void samba_io_uring_completion_prepare(struct samba_io_uring_completion *completion, void (*completion_prepare(struct samba_io_uring_completion *completion, void *completion_private, comst struct io_uring_completion_private); | void *completion_private); | void *completion_private); | void samba_io_uring_submission *submission, void (*submission_prepare(struct samba_io_uring_submission *submission, void (*submission_prepare(struct samba_io_uring_submission *submission, void *submission_private), | void *submission_private, | struct samba_io_uring_completion *completion); | struct io_uring_see *samba_io_uring_completion *completion; | submission; | submission; | submission; | submission; | submission; | submission; | submission *submission *submission *submission; | struct samba_io_uring_submission *submission; | struct samba_io_uring_submission *submission; | struct samba_io_uring_submission *submission); | struct samba_io_uring_submission *submission; | struct samba_io_uring_submission *submission; | struct samba_io_uring_submission *submission; |
```

- ▶ Using it ...
  - convert vfs\_io\_uring
  - use it in smb2\_server.c
  - In future use it in other performance critical places too.





#### The road to upstream (smb2\_server.c)

000000000

- Refactoring of smb2\_server.c
  - ▶ add optional IORING\_OP\_SENDMSG, IORING\_OP\_RECVMSG support
- ► There are structural problems with splice from a file
  - ▶ I had a discussion with the Linux developers about it:
    - ► The page content from the page cache may change unexpectetly
    - https://lists.samba.org/archive/samba-technical/2023-February/thread.html#137945
    - ▶ We may not able to use IORING\_OP\_SENDMSG/SPLICE by default
    - ► Maybe IORING\_OP\_RECVMSG/SPLICE is possible
- ▶ At least we can have only 1 one copy instead of two:
  - ▶ IORING\_OP\_SENDMSG\_ZC is able to avoid copying to the socket
    ▶ we get an extra completion once the buffers are not needed anymore
  - ► This gives good results, between with and without IORING\_OP\_SENDMSG/SPLICE
  - But I don't have numbers as it doesn't work on loopback
  - Within VM's improvement can be seen



Stefan Metzmacher

io uring (19/21)

# SerNet

### Future Improvements

- ▶ I have a prototype for a native io\_uring tevent backend:
  - ► The idea is to avoid epoll and only block in io\_uring\_enter()
  - ► But the semantics of IORING\_OP\_POLL\_ADD, REMOVE are not useable
  - https://lists.samba.org/archive/samba-technical/2022-October/thread.html#137734
  - ▶ We may get an IORING\_POLL\_CANCEL\_ON\_CLOSE in future
  - And a usable IORING\_POLL\_LEVEL
- ▶ We can use io\_uring deep inside of the smbclient code
  - The low layers can just use samba\_io\_uring\_ev\_context\_get\_ring()
  - And use if available without changing the whole stack



000000000

- ► Stefan Metzmacher, metze@samba.org
- ► https://www.sernet.com
- https://samba.plus
- → SerNet/SAMBA+ sponsor booth

Slides: https://samba.org/~metze/presentations/2023/SDC/



Stefan Metzmacher

io\_uring (21/21)

SerNet