Rocky Enterprise Linux 9.2 Manual Pages on command 'io_submit.2'

$ man io_submit.2

IO_SUBMIT(2)   Linux Programmer's Manual   IO_SUBMIT(2)

NAME

  io_submit - submit asynchronous I/O blocks for processing

SYNOPSIS

  #include <linux/aio_abi.h>   /* Defines needed types */

  int io_submit(aio_context_t ctx_id, long nr, struct iocb **iocbpp);

  Note: There is no glibc wrapper for this system call; see NOTES.

DESCRIPTION

  Note: this page describes the raw Linux system call interface. The wrapper function pro?

  vided by libaio uses a different type for the ctx_id argument. See NOTES.

  The io_submit() system call queues nr I/O request blocks for processing in the AIO context

  ctx_id.   The iocbpp argument should be an array of nr AIO control blocks, which will be

  submitted to context ctx_id.

  The iocb (I/O control block) structure defined in linux/aio_abi.h defines the parameters

  that control the I/O operation.

```
  #include <linux/aio_abi.h>

  struct iocb {
    __u64   aio_data;
    __u32   PADDED(aio_key, aio_rw_flags);
    __u16   aio_lio_opcode;
    __s16   aio_reqprio;
    __u32   aio_fildes;
    __u64   aio_buf;
```

```
        __u64   aio_nbytes;

        __s64   aio_offset;

        __u64   aio_reserved2;

        __u32   aio_flags;

        __u32   aio_resfd;

    };
```

The fields of this structure are as follows:

aio_data

> This data is copied into the data field of the io_event structure upon I/O comple?
>
> tion (see io_getevents(2)).

aio_key

> This is an internal field used by the kernel. Do not modify this field after an
>
> io_submit() call.

aio_rw_flags

> This defines the R/W flags passed with structure. The valid values are:
>
> RWF_APPEND (since Linux 4.16)
>
>> Append data to the end of the file. See the description of the flag of the
>>
>> same name in pwritev2(2) as well as the description of O_APPEND in open(2).
>>
>> The aio_offset field is ignored. The file offset is not changed.
>
> RWF_DSYNC (since Linux 4.13)
>
>> Write operation complete according to requirement of synchronized I/O data
>>
>> integrity. See the description of the flag of the same name in pwritev2(2)
>>
>> as well the description of O_DSYNC in open(2).
>
> RWF_HIPRI (since Linux 4.13)
>
>> High priority request, poll if possible
>
> RWF_NOWAIT (since Linux 4.14)
>
>> Don't wait if the I/O will block for operations such as file block alloca?
>>
>> tions, dirty page flush, mutex locks, or a congested block device inside the
>>
>> kernel.   If any of these conditions are met, the control block is returned
>>
>> immediately with a return value of -EAGAIN in the res field of the io_event
>>
>> structure (see io_getevents(2)).
>
> RWF_SYNC (since Linux 4.13)
>
>> Write operation complete according to requirement of synchronized I/O file

integrity. See the description of the flag of the same name in pwritev2(2)

as well the description of O_SYNC in open(2).

aio_lio_opcode

This defines the type of I/O to be performed by the iocb structure. The valid val?

ues are defined by the enum defined in linux/aio_abi.h:

enum {

IOCB_CMD_PREAD = 0,

IOCB_CMD_PWRITE = 1,

IOCB_CMD_FSYNC = 2,

IOCB_CMD_FDSYNC = 3,

IOCB_CMD_POLL = 5,

IOCB_CMD_NOOP = 6,

IOCB_CMD_PREADV = 7,

IOCB_CMD_PWRITEV = 8,

};

aio_reqprio

This defines the requests priority.

aio_fildes

The file descriptor on which the I/O operation is to be performed.

aio_buf

This is the buffer used to transfer data for a read or write operation.

aio_nbytes

This is the size of the buffer pointed to by aio_buf.

aio_offset

This is the file offset at which the I/O operation is to be performed.

aio_flags

This is the set of flags associated with the iocb structure. The valid values are:

IOCB_FLAG_RESFD

Asynchronous I/O control must signal the file descriptor mentioned in

aio_resfd upon completion.

IOCB_FLAG_IOPRIO (since Linux 4.18)

Interpret the aio_reqprio field as an IOPRIO_VALUE as defined by linux/io?

prio.h.

aio_resfd

> The file descriptor to signal in the event of asynchronous I/O completion.

RETURN VALUE

> On success, io_submit() returns the number of iocbs submitted (which may be less than nr,
>
> or 0 if nr is zero). For the failure return, see NOTES.

ERRORS

> EAGAIN Insufficient resources are available to queue any iocbs.
>
> EBADF The file descriptor specified in the first iocb is invalid.
>
> EFAULT One of the data structures points to invalid data.
>
> EINVAL The AIO context specified by ctx_id is invalid. nr is less than 0. The iocb at
>
>> *iocbpp[0] is not properly initialized, the operation specified is invalid for the
>>
>> file descriptor in the iocb, or the value in the aio_reqprio field is invalid.
>
> ENOSYS io_submit() is not implemented on this architecture.
>
> EPERM The aio_reqprio field is set with the class IOPRIO_CLASS_RT, but the submitting
>
>> context does not have the CAP_SYS_ADMIN capability.

VERSIONS

> The asynchronous I/O system calls first appeared in Linux 2.5.

CONFORMING TO

> io_submit() is Linux-specific and should not be used in programs that are intended to be
>
> portable.

NOTES

> Glibc does not provide a wrapper function for this system call. You could invoke it using
>
> syscall(2). But instead, you probably want to use the io_submit() wrapper function pro?
>
> vided by libaio.
>
> Note that the libaio wrapper function uses a different type (io_context_t) for the ctx_id
>
> argument. Note also that the libaio wrapper does not follow the usual C library conven?
>
> tions for indicating errors: on error it returns a negated error number (the negative of
>
> one of the values listed in ERRORS). If the system call is invoked via syscall(2), then
>
> the return value follows the usual conventions for indicating an error: -1, with errno set
>
> to a (positive) value that indicates the error.

SEE ALSO

> io_cancel(2), io_destroy(2), io_getevents(2), io_setup(2), aio(7)

COLOPHON

This page is part of release 5.10 of the Linux man-pages project. A description of the project, information about reporting bugs, and the latest version of this page, can be found at https://www.kernel.org/doc/man-pages/.

Linux   2020-12-21   IO_SUBMIT(2)