Operating systems laba 2

Varfolomeeva Anastasia

11 октября 2023 г.

1 accept

int accept(int sockfd, struct sockaddr *_Nullable restrict addr, socklen_t *_Nullable restrict addrlen);

1.1 Назначение

Системный вызов accept() используется с сокетами, ориентированными на установление соединения. Она извлекает первый запрос на соединение из очереди ожидающих соединений прослушивающего сокета, sockfd, создаёт новый подключенный сокет и и возвращает новый файловый дескриптор, указывающий на сокет. Новый сокет более не находится в слушающем состоянии. Исходный сокет sockfd не изменяется при этом вызове.

1.2 Аргументы

sockfd — это новый сокет, который привязан к локальному адресу и прослушивает соединения.

addr — это указатель на структуру sockaddr. В эту структуру помещается адрес ответной стороны в том виде, в каком он известен на коммуникационном уровне. Точный формат адреса, возвращаемого в параметре addr, определяется семейством адресов сокета. Если addr равен NULL, то ничего не помещается; в этом случае addrlen не используется и также должен быть NULL.

Через аргумент addrlen осуществляется возврат результата: вызывающая сторона должна указать в нём размер (в байтах) структуры, на которую указывает addr; при возврате он будет содержать реальный размер адреса ответной стороны.

1.3 Возвращаемые значения

Этот системный вызов возвращает -1 в случае ошибки. При успешном завершении возвращается неотрицательное целое, являющееся дескриптором сокета.

2 connect

int connect(int sockfd, const struct sockaddr *addr, socklen_t addrlen);

2.1 Назначение

Системный вызов connect() устанавливает соединение с сокетом, заданным файловый дескриптором sockfd, ссылающимся на адрес addr. Аргумент addrlen определяет размер addr. Формат адреса в addr определяется адресным пространством сокета sockfd.

2.2 Аргументы

sockfd - файловый дескриптор, который ссылается на сокет.

Если сокет имеет тип SOCK_DGRAM, значит, адрес addr является адресом по умолчанию, куда посылаются датаграммы, и единственным адресом, откуда они принимаются. Если сокет имеет тип SOCK_STREAM или SOCK_SEQPACKET, то данный системный вызов попытается

установить соединение с другим сокетом. Другой сокет задан параметром addr, являющийся адресом длиной addrlen в пространстве коммуникации сокета. Каждое пространство коммуникации интерпретирует параметр addr по-своему.

2.3 Возвращаемые значения

Если соединение или привязка прошла успешно, возвращается нуль. При ошибке возвращается -1, а errno устанавливается должным образом.

3 send, sendto

ssize_t send(int sockfd, const void buf[.len], size_t len, int flags);

ssize_t sendto(int sockfd, const void buf[.len], size_t len, int flags, const struct sockaddr *dest_addr, socklen_t addrlen);

3.1 Назначение

send и sendto используются для пересылки сообщений в другой сокет. send можно использовать, только если сокет находится в состоянии соединения, тогда как sendto можно использовать в любое время.

вызов
send(sockfd, buf, len, flags);
эквивалентен
sendto(sockfd, buf, len, flags, NULL, 0);

3.2 Аргументы

sockfd - файловый дескриптор сокета отправления.

Если sendto() используется с сокетом в режиме с установлением соединения (SOCK_STREAM, SOCK_SEQPACKET), то аргументы dest_addr и addrlen игнорируются (и может быть возвращена ошибка EISCONN, если их значения не равны NULL и 0) и возвращается ошибка ENOTCONN, если соединение через сокет не установлено. Иначе в dest_addr задаётся адрес назначения и его размер в addrlen.

У send() и sendto() сообщение находится в buf, а его длина в len.

Когда сообщение не помещается в буфер отправки сокета, выполнение блокируется в send(), если сокет не находится в неблокирующем режиме. Если сокет находится в неблокирующем режиме, то возвращается ошибка EAGAIN или EWOULDBLOCK.

Параметр flags является битовой маской и может содержать такие флаги:

MSG_OOB - Посылает внепотоковые данные, если сокет это поддерживает.

MSG_DONTROUTE Не использовать маршрутизацию при отправке пакета, а посылать его только на хосты в локальной сети. Определен только для маршрутизируемых семейств протоколов.

MSG_DONTWAIT Включает режим non-blocking; если операция должна была заблокировать программу, возвращается EAGAIN

MSG_NOSIGNAL Требует не посылать сигнал SIGPIPE, если при работе с ориентированным на поток сокетом другая сторона обрывает соединение. Код ошибки EPIPE возвращается в любом случае.

MSG_CONFIRM Сообщает, что процесс пересылки произошел: вы получаете успешный ответ с другой стороны. Если уровень связи не получает его, он регулярно перепроверяет сеть.

3.3 Возвращаемые значения

Эти системные вызовы возвращают количество отправленных символов или -1, если произошла ошибка.

4 recv, recvfrom

ssize_t recv(int sockfd, void buf[.len], size_t len, int flags);

ssize_t recvfrom(int sockfd, void buf[restrict .len], size_t len, int flags, struct sockaddr *_Nullable restrict src_addr, socklen_t *_Nullable restrict addrlen);

4.1 Назначение

Системные вызовы recv() и recvfrom() используются для получения сообщений из сокета. Они могут использоваться для получения данных, независимо от того, является ли сокет ориентированным на соединения.

вызов
recv(sockfd, buf, len, flags)
эквивалентен
recvfrom(sockfd, buf, len, flags, NULL, NULL)

4.2 Аргументы

Вызов recvfrom() помещает принятое сообщение в буфер buf. Вызывающий должен указать размер буфера в len.

Если значение src_addr не равно NULL, и в нижележащем протоколе используется адрес источника сообщения, то адрес источника помещается в буфер, указанный в src_addr. В этом случае addrlen является аргументом-результатом. Перед вызовом ему должно быть присвоено значение длины буфера, связанного с src_addr. При возврате addrlen обновляется и содержит действительный размер адреса источника. Возвращаемый адрес обрезается, если предоставленный буфер слишком мал; в этом случае addrlen будет содержать значение большее, чем указывалось в вызове.

Вызов recv(), обычно, используется только на соединённом сокете. Он идентичен вызову:

recvfrom(fd, buf, len, flags, NULL, 0);

Аргумент flags системного вызова recv формируется с помощью объединения логической операцией ИЛИ одного или более нижеследующих значений:

MSG_OOB Этот флаг запрашивает прием внепотоковых данных, которые в противном случае не были бы получены в обычном потоке данных.

MSG_PEEK Этот флаг заставляет выбрать данные из начала очереди, но не удалять их оттуда.

MSG_WAITALL Этот флаг просит подождать, пока не придет полное запрошенное количество данных.

MSG_TRUNC Возвращает реальную длину пакета, даже если она была больше, чем предоставленный буфер.

MSG_ERRQUEUE Получить пакет из очереди ошибок.

MSG_NOSIGNAL Этот флаг отключает возникновение сигнала SIGPIPE на потоковых сокетах, если другая сторона вдруг исчезает.

MSG_ERRQUEUE Указание этого флага позволяет получить из очереди ошибок сокета накопившиеся ошибки.

4.3 Возвращаемые значения

Эти системные вызовы возвращают количество принятых байт или -1, если произошла ошибка.

5 shutdown

int shutdown(int sockfd, int how);

5.1 Назначение

Вызов shutdown() приводит к закрытию всего полнодуплексного соединения или его части в сокете, связанном с sockfd.

5.2 Аргументы

sockfd - файловый дескриптор сокета.

Если значение how равно SHUT_RD, то дальнейший приём данных будет запрещён. Если значение how равно SHUT_WR, то дальнейшая передача данных будет запрещена. Если значение how равно SHUT_RDWR, то дальнейший приём и передача данных будут запрещены.

5.3 Возвращаемые значения

В случае успеха возвращается ноль. В случае ошибки возвращается -1

6 mmap

void *mmap(void addr[.length], size_t length, int prot, int flags, int fd, off_t offset);

6.1 Назначение

Вызов mmap() создаёт новое отображение в виртуальном адресном пространстве вызывающего процесса.

6.2 Аргументы

Адрес начала нового отображения указывается в addr. В аргументе length задаётся длина отображения (должна быть больше 0).

Если значение addr равно NULL, то ядро само выбирает адрес, по которому создаётся отображение

Содержимое файлового отображения инициализируется данными из файла (или объекта), на который указывает файловый дескриптор fd, длиной length байт, начиная со смещения offset. Значение offset должно быть кратно размеру страницы.

В аргументе prot указывается желаемая защита памяти отображения. Значением может быть PROT_NONE или побитово сложенные следующие флаги:

PROT_EXEC - Страницы доступны для исполнения.

PROT_READ - Страницы доступны для чтения.

PROT_WRITE - Страницы доступны для записи.

PROT_NONE - Страницы недоступны.

Параметр flags задает тип отражаемого объекта, опции отражения и указывает, принадлежат ли отраженные данные только этому процессу или их могут читать другие. Он состоит из комбинации следующих битов:

MAP_FIXED Не использовать другой адрес, если адрес задан в параметрах функции.

MAP_SHARED Разделить использование этого отражения с другими процессами, отражающими тот же объект.

MAP_PRIVATE Создать неразделяемое отражение с механизмом copy-on-write. Запись в эту область памяти не влияет на файл.

Вы должны задать либо MAP_SHARED, либо MAP_PRIVATE.

6.3 Возвращаемые значения

При удачном выполнении mmap возвращает указатель на область с отраженными данными. При ошибке возвращается значение MAP_FAILED (-1), а переменная errno приобретает соответствующее значение.

7 munmap

int munmap(void addr[.length], size_t length);

7.1 Назначение

Системный вызов munmap() удаляет отображение для указанного адресного диапазона и это приводит к тому, что дальнейшее обращение по адресам внутри диапазона приводит к генерации неправильных ссылок на память. Также для диапазона отображение автоматически удаляется при завершении работы процесса. С другой стороны, закрытие файлового дескриптора не приводит к удалению отображения диапазона.

7.2 Аргументы

Адрес addr должен быть кратен размеру страницы (но значения length это не касается). Все страницы, содержащие часть указанного диапазона, удаляются из отображения и последующие ссылки на эти страницы приводят к генерации сигнала SIGSEGV. Это не ошибка, если указанный диапазон не содержит каких-либо отображённых страниц.

7.3 Возвращаемые значения

При удачном выполнении munmap возвращаемое значение равно нулю. При ошибке возвращается -1, а переменная errno приобретает соответствующее значение.

8 msync

int msync(void addr[.length], size_t length, int flags);

8.1 Назначение

Вызов msync() сбрасывает изменения файла, который отображён в память с помощью mmap(), обратно в файловую систему. Без использования этого вызова нет никакой гарантии, что изменения будут записаны в файл до вызова munmap(). Если быть точнее, то на диск записывается часть файла, начинающаяся в памяти с адреса addr длиной length.

8.2 Аргументы

addr - адрес файла в памяти

length - длина

В аргументе flags должен быть один из флагов MS_ASYNC и MS_SYNC, а также дополнительно можно указать MS_INVALIDATE. Данные биты имеют следующее значение:

MS_ASYNC Запланировать обновление, но вызов завершается сразу.

MS_SYNC Запланировать обновление и ждать его завершения.

MS_INVALIDATE Считать недействительными другие отображения того же файла (для того, чтобы они могли обновиться до достоверных значений, которые запишутся).

8.3 Возвращаемые значения

При удачном завершении вызова возвращаемое значение равно нулю. При ошибке оно равно -1, а переменной errno присваивается номер ошибки.