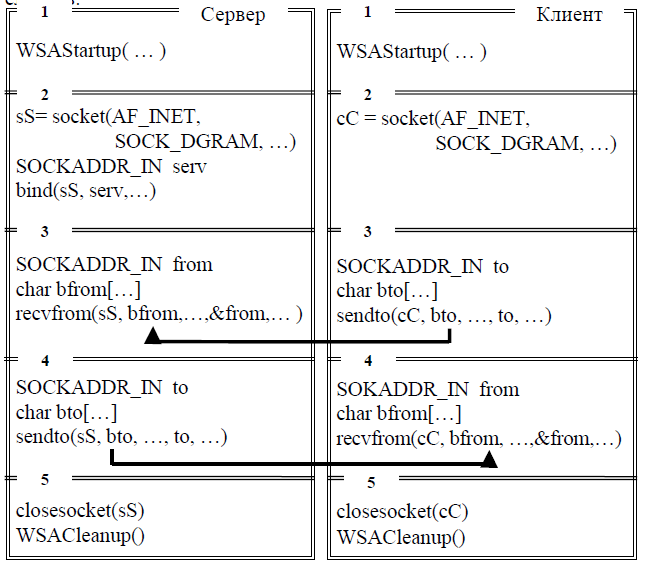
1. **Структура UDP-сервера.**
2. **Структура UDP-клиента.**



Первые блоки обеих программ одинаковые и предназначены для инициализации библиотеки WS2\_32.dll. Второй блок программы-сервера создает сокет (функция socket) и устанавливает его параметры. Следует обратить внимание на параметр SOCK\_DGRAM функции socket, указывающий на тип сокета (в данном случае – сокет, ориентированный на сообщения). Для установки параметров сокета используется функция bind. При этом говорят, что сокеты связывают параметрами. Для хранения параметров сокета в Winsock2 предусмотрена специальная структура SOCKADDR\_IN. перед выполнением функции bind, которая использует эту структуру в качестве параметра, необходимо ее заполнить данными. В SOCKADDR\_IN хранится IP-адрес и номер порта сервера.

В третьем блоке программы сервера выполняется функция recvfrom, которая переводит программу сервера в состояние ожидания до поступления сообщения от программы клиента (функция sendto). Функция recvfrom тоже использует структуру SOCKADDR\_IN. В нее автоматически помещаются параметры сокета клиента после приема от него сообщения. Данные поступают в буфер, который обеспечивает принимающая сторона (на рисунке символьный массив bfrom). Следует отметить, что в качестве параметра функции recvfrom используется связанный сокет и именно через него осуществляется передача данных.

Четвертый блок программы сервера предназначен для пересылки данных клиенту. Процесс осуществляется с помощью функции sendto. В качестве параметров sendto использует структуру SOCKADDR\_IN с параметрами сокета принимающей стороны (в данном случае клиента) и заполненный буфер с данными.

Пятые блоки программ сервера и клиента одинаковые и предназначены для закрытия сокета и завершения работы с библиотекой WS2\_32.dll.

Всем блокам программы клиента, кроме второго, есть аналог в программе сервера. Второй блок по сравнению с сервером не использует команду bind. Здесь проявляется основное отличие между сервером и клиентом. Если сервер должен использовать однозначно определенные параметры (IP-адрес и номер порта), то для клиента это не обязательно – ему Windows выделяет эфемерный порт. Поскольку инициатором связи является клиент, то он должен точно «знать» параметры сокета сервера, а свои параметры клиент получит от Windows и сообщит их вместе с переданным пакетом серверу.

1. **Отличие UDP-сервера от TCP-сервера.**

UDP (User Datagram Protocol) и TCP (Transmission Control Protocol) - это два протокола передачи данных, которые используются в сетевых приложениях для установления соединения и передачи данных между узлами сети.

Основное отличие между UDP-сервером и TCP-сервером заключается в том, как они управляют передачей данных:

1. Надежность передачи: TCP-сервер обеспечивает надежную передачу данных, т.е. гарантирует, что все отправленные данные будут доставлены получателю в правильном порядке и без потерь. В то время как UDP-сервер не гарантирует надежную доставку данных, и потери пакетов могут произойти в случае перегрузки сети или других факторов.
2. Установление соединения: TCP-сервер устанавливает соединение с клиентом перед началом передачи данных, т.е. сначала происходит процесс установления соединения, а затем передача данных. В отличие от TCP, UDP-сервер не устанавливает соединение и начинает передачу данных немедленно.
3. Производительность: UDP-сервер более быстрый, чем TCP-сервер, так как не требует установления соединения и проверки надежности передачи данных. Однако это может привести к потере данных или их дублированию, если сеть перегружена или ненадежна.
4. Размер пакета: TCP-пакеты обычно более объемные, чем UDP-пакеты, так как TCP включает в себя много служебной информации для обеспечения надежности передачи данных.
5. Применение: UDP-серверы часто используются для передачи потоковых данных, таких как аудио или видео, где некоторая потеря данных не столь критична. В то время как TCP-серверы наиболее подходят для передачи данных, где требуется высокая степень надежности и точности
6. **Отличие UDP-клиента от TCP-клиента.**

UDP-клиент и TCP-клиент — это различные типы клиентских приложений, использующих протоколы UDP и TCP соответственно для обмена данными с сервером.

Отличие между UDP-клиентом и TCP-клиентом заключается в протоколе, который используется для обмена данными между клиентом и сервером, а также в некоторых характеристиках этих протоколов:

1. Надежность: TCP-клиент обеспечивает надежную передачу данных путем установления соединения и проверки доставки каждого пакета. В то время как UDP-клиент не гарантирует доставку данных, и пакеты могут быть потеряны, дублированы или прийти в неправильном порядке.
2. Ориентация на соединение: TCP-клиент устанавливает соединение между клиентом и сервером перед передачей данных, в то время как UDP-клиент не устанавливает соединение, и каждый пакет рассматривается независимо от других.
3. Скорость передачи: UDP-клиент обычно быстрее, чем TCP-клиент, потому что UDP не имеет механизмов установления соединения и проверки доставки каждого пакета, что снижает накладные расходы.
4. Объем передаваемых данных: UDP-клиент используется в основном для передачи небольших объемов данных, таких как аудио и видео потоков, тогда как TCP-клиент часто используется для передачи больших объемов данных, таких как файлов.
5. ***Мультикаст***: UDP-клиент поддерживает мультикаст, то есть может отправлять данные на несколько устройств одновременно, в то время как TCP-клиент не поддерживает мультикаст.

В целом, выбор между UDP и TCP зависит от конкретных требований приложения и условий сети. Если важна надежность и целостность передаваемых данных, то следует использовать TCP, а если важна скорость и мультикаст, то UDP может быть более подходящим выбором.

1. **Что такое канал связи с точки зрения TCP?**

Канал связи (или соединение) создается между двумя сокетами, ориентированными на поток. На стороне сервера это должен быть связанный (функция bind) и переключенный в режим прослушивания (функция listen) сокет. На стороне клиента должен быть создан дескриптор ориентированного на поток сокета (функция socket).

Канал связи создается в результате взаимодействия функция accept (на стороне сервера) и connect (на стороне клиента). Алгоритм взаимодействия этих функция зависит от установленного режима ввода-вывода для участвующих в создании каналов сокетов.

1. **Что значит: обмен без соединения(ориентированный на сообщения)?**

Это такой обмен данными (сообщениями), особенностью которого является то, что протоколом, который обеспечивает такой обмен, не гарантируется доставка и правильная последовательность приема отправленных сообщений. Весь контроль надежности доставки сообщений возлагается на разработчика приложения. В связи с этим, обмен данными с помощью сообщений используется в основном для широковещательных сообщений или для пересылки коротких сообщений, последовательность получения которых не имеет значения.

1. **Что значит UDP ненадежный протокол?**

Ненадежные протоколы - это протоколы, которые не гарантируют доставку сообщений. Они могут быть использованы в условиях, когда потеря сообщения не сильно повлияет на работу приложения. Ненадежные протоколы обычно используются для передачи данных в реальном времени, например, видео- и аудиопотоков. Обладает следующими свойствами:

***1) отсутствие механизмов обеспечения надежности;***

***2) отсутствие гарантий доставки;***

***3) отсутствие возможности подсчёта контрольной суммы;***

***4) отсутствие буферизации;***

***5) отсутствие фрагментации;***

***6) отсутствие соединения.***

1. **В каких случаях следует выбирать UDP-протокол для обмена данными?**

Обмен данными с помощью сообщений используется в основном для широковещательных сообщений или для пересылки коротких сообщений, последовательность получения которых не имеет значения.

UDP-протокол следует выбирать в следующих случаях:

1. Необходимо отправлять данные на быструю передачу, где потерянные данные не критичны для работы приложения. Например, трансляции мультимедиа-контента, игры в реальном времени.

2. Передача данных в режиме мультикаст, где один сервер отправляет данные нескольким получателям одновременно.

3. Отправка небольших пакетов данных с минимальной задержкой.

4. Приложения, где требуется минимальная нагрузка на сеть.

5. Протоколы, которые не требуют установления соединения, например, DNS или DHCP.

1. **Что значит: UDP не буферизирует соединения?**

Обрабатывает только один пакет, который пришел. Пока не обработает этот пакет, другие пакеты не обрабатываются, т.к. нет буфера памяти.

1. **Можно ли одновременно задействовать один номер порта для двух серверов: TCP и TCP, UDP и UDP, TCP и UDP? Если возможно, то продемонстрируйте.**

Нельзя задействовать один и тот же номер порта одновременно для двух серверов одного типа (TCP или UDP), так как номер порта является уникальным идентификатором приложения на хосте, и два приложения не могут использовать один и тот же порт одновременно. Однако можно, если хоть что-то из этого будет отличаться:

1. IP
2. Порт
3. Протолок

Однако, можно задействовать один номер порта для двух серверов разных типов (TCP и UDP). В этом случае, один сервер будет слушать этот порт по протоколу TCP, а другой - по протоколу UDP.

1. **Определите порт UDP-сервера с помощью утилиты NETSTAT.**

netstat -ano -p udp

1. **Продемонстрируйте потерю сообщений UDP-сервером. Используйте функцию Sleep.**

У нас отправляется большое количество пакетов серверу, после “выхода из sleep” сервера, у нас сервер не справляется с таким количеством пакетов.