CIDR: бесклассовая маршрутизация между доменами (Classless Interdomain Routing)

Это способ, который позволяет свести к минимуму рост таблиц маршрутизации

описывается в RFC 1518 и в RFC 1519

Для того чтобы использовать подобное суммирование, необходимо, чтобы выполнялось три условия.

1. Несколько IP адресов, которые будут сложены вместе для маршрутизации, должны иметь в своих адресах одинаковые биты старшего порядка.
2. Таблицы маршрутизации и алгоритмы маршрутизации должны быть расширены таким образом, чтобы решения о маршрутизации принимались с использованием 32-битных IP адресов и 32-битных масок.
3. Протоколы маршрутизации, которые будут использоваться, должны быть расширены таким образом, чтобы позволять передачу 32-битных масок и 32-битных адресов. OSPF и RIP-2 способны передавать 32-битные маски, так же как и BGP Version 4

Термин "бесклассовый" используется потому, что решения о маршрутизации принимаются на основе масок, накладываемых на полный 32-битный IP адрес. При этом не существует различия между адресами класса A, B или C

Протокол TCP (Transmission Control Protocol, Протокол контроля передачи) - обеспечивает сквозную доставку данных между прикладными процессами, запущенными на узлах, взаимодействующих по сети. Стандартное описание TCP содержится в RFC 793.

TCP обеспечивает надежность передачи благодаря следующему:

- Данные от приложения разбиваются на блоки определенного размера, которые будут отправлены..
- Когда TCP посылает сегмент, он устанавливает таймер, ожидая, что с удаленного конца придет подтверждение на этот сегмент. Если подтверждение не получено по истечении времени, сегмент передается повторно.
- Когда TCP принимает данные от удаленной стороны соединения, он отправляет подтверждение. Это подтверждение не отправляется немедленно, а обычно задерживается на доли секунды.
- TCP осуществляет расчет контрольной суммы для своего заголовка и данных. Это контрольная сумма, рассчитываемая на концах соединения, целью которой является выявить любое изменение данных в процессе передачи. Если сегмент прибывает с неверной контрольной суммой, TCP отбрасывает его и подтверждение не генерируется. (Ожидается, что отправитель отработает тайм-аут и осуществит повторную передачу.)
- Так как TCP сегменты передаются в виде IP датаграмм, а IP датаграммы могут прибывать беспорядочно, также беспорядочно могут прибывать и TCP сегменты. После получения данных TCP может по необходимости изменить их последовательность, в результате приложение получает данные в правильном порядке.
- Так как IP датаграмма может быть продублирована, принимающий TCP должен отбрасывать продублированные данные.
- TCP осуществляет контроль потока данных. Каждая сторона TCP соединения имеет определенное пространство буфера. TCP на принимающей стороне позволяет удаленной стороне посылать данные только в том случае, если получатель может поместить их в буфер. Это предотвращает переполнения буферов медленных хостов быстрыми хостами.

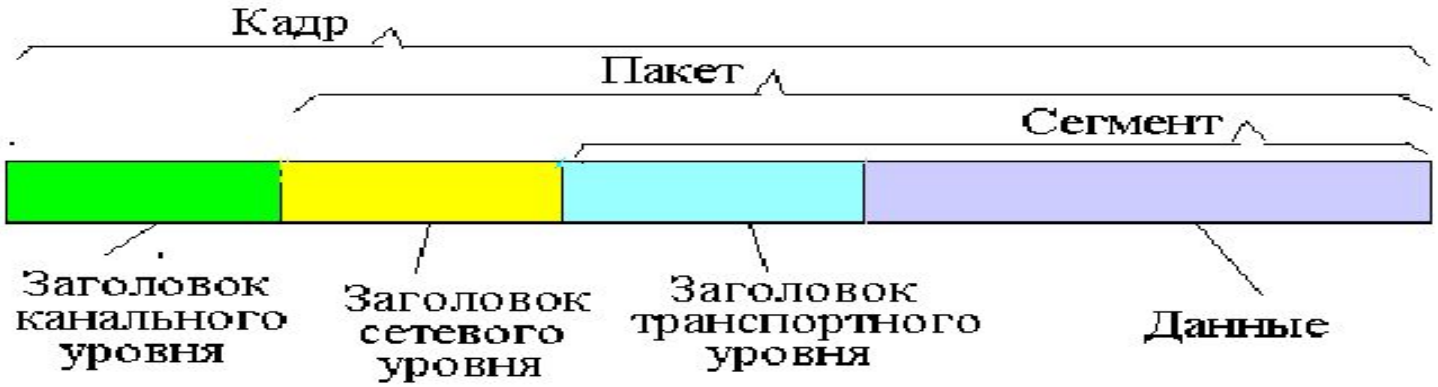
Протокол TCP обеспечивает работу одновременно нескольких соединений. Каждый прикладной процесс идентифицируется *номером порта*

Алгоритм:

1. При получении дейтаграммы, в поле Protocol которой указан код протокола TCP (6), модуль IP передает данные этой дейтаграммы модулю TCP. Эти данные представляют собой TCP-сегмент, содержащий TCP-заголовок и данные пользователя (прикладного процесса).
2. Модуль TCP анализирует служебную информацию заголовка, определяет, какому именно процессу предназначены данные пользователя, проверяет целостность и порядок прихода данных и подтверждает их прием другой стороне. По мере получения правильной последовательности неискаженных данных пользователя они передаются прикладному процессу

Совокупность IP-адреса и номера порта называется *сокетом*

Сети TCP/



Протокол TCP

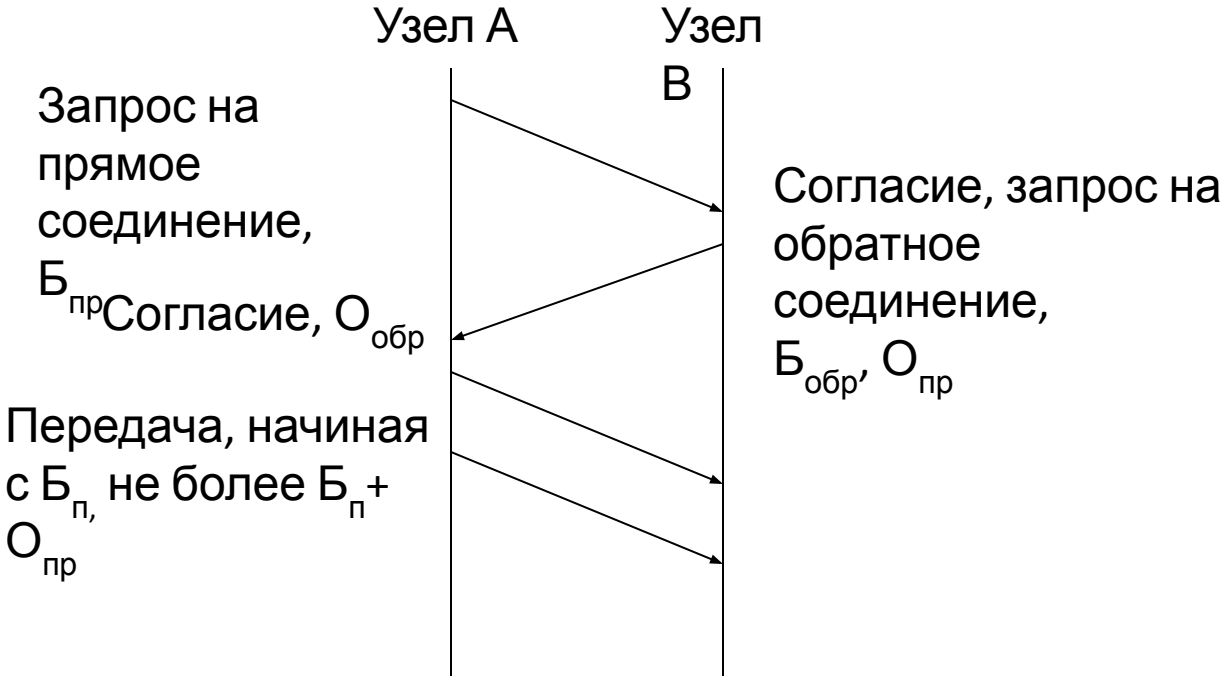
- **порт отправителя - 16;**
- **порт получателя - 16;**
- **код позиции (номер первого байта в поле данных) - 32;**
- **подтверждение (номер первого еще не полученного байта) - 32;**
- **управление - 16;**
- **размер окна -16;**
- **CRC - 16;**
- **признаки - 16;**
- **опции - 24;**
- **заполнитель - 8;**
- **данные.**

Функции TCP

Порт отправителя	Порт получателя
Начало сегмента (адрес 1-го байта) SN	
Подтверждение (ожидаемый адрес) ACK	
Управление	Размер окна
Контр. код	Дополнит. признаки
Опции	
Данные	

Открытие соединения клиентом осуществляется вызовом функции OPEN
Закрытие соединения клиентом производится с помощью функции CLOSE

***Установление соединения - трехшаговая процедура*:**
узел А - указывает номер своего первого байта, узел В подтверждает согласие и указывает номер своего первого байта, узел А подтверждает согласие.

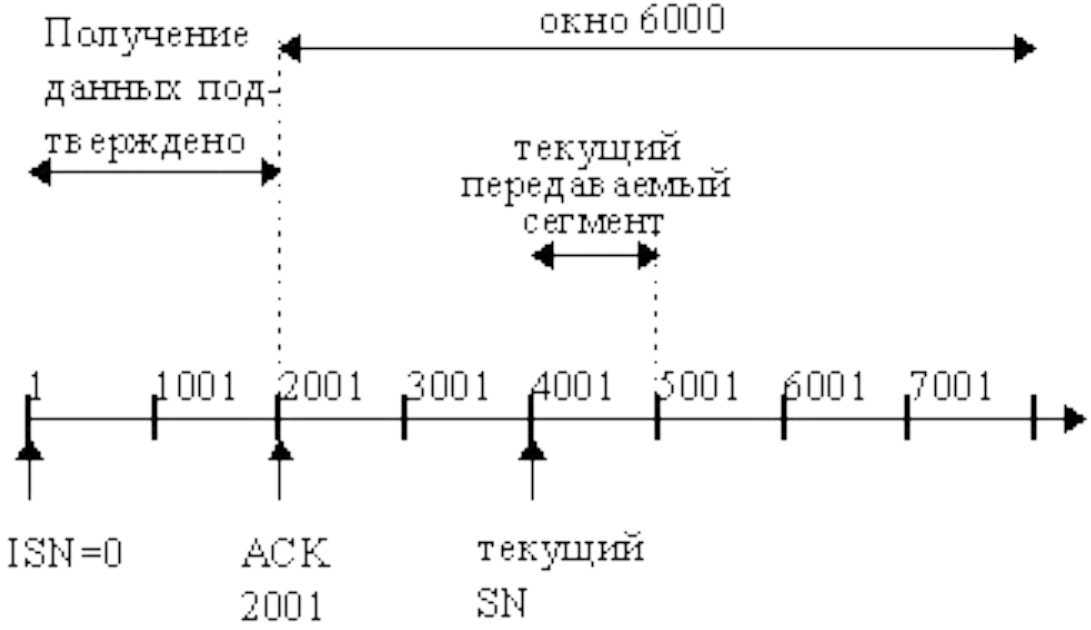


Процедура медленного старта – использует рост окна до возникновения отказа.
Окно показывает, сколько сегментов TCP-модуль, с его собственной точки зрения, может отправить без получения подтверждения.

Управление потоком

Для ускорения и оптимизации процесса передачи больших объемов данных протокол TCP использует **метод скользящего окна**, который позволяет отправителю посылать очередной сегмент, не дожидаясь подтверждения о получении в пункте назначения предшествующего сегмента.

Скользящее же окно показывает, какой объем неподтвержденных данных модулю разрешено отправить с точки зрения удаленного модуля, получателя его данных



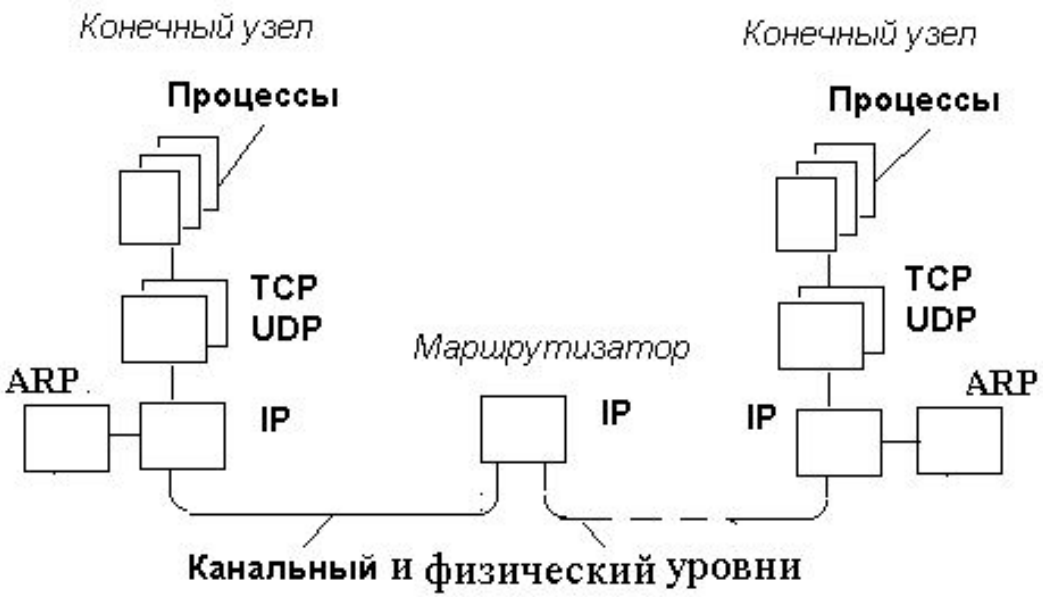
Протокол TCP формирует подтверждения не для каждого успешно полученного пакета, а для всех данных от начала посылки до некоторого порядкового номера ACK SN (Acknowledge Sequence Number).

В качестве подтверждения успешного приема, например, первых 2000 байт, высылается ACK SN = 2001: это означает, что все данные в байтовом потоке под номерами от ISN+1=1 до данного ACK SN -1 (2000) успешно получены

Вместе с посылкой отправителю ACK SN получатель объявляет также “размер окна”

Размер окна выбирается таким образом, чтобы подтверждения успевали приходить вовремя и остановки передачи не происходило

Взаимодействие процессов в сетях TCP/IP



<u>Установление соединения</u>:

свой номер порта и IP-имя получателя --> на транспортный уровень -->

обращение к DNS, получаем IP-адрес --> на сетевой уровень -->

обращение к ARP-таблице: или получаем MAC-адрес

или посылается ARP-запрос, при неудаче передача в сеть через порт маршрутизатора.