TCP: Transmission Control Protocol

Управление соединениями

TCP: Transmission Control Protocol

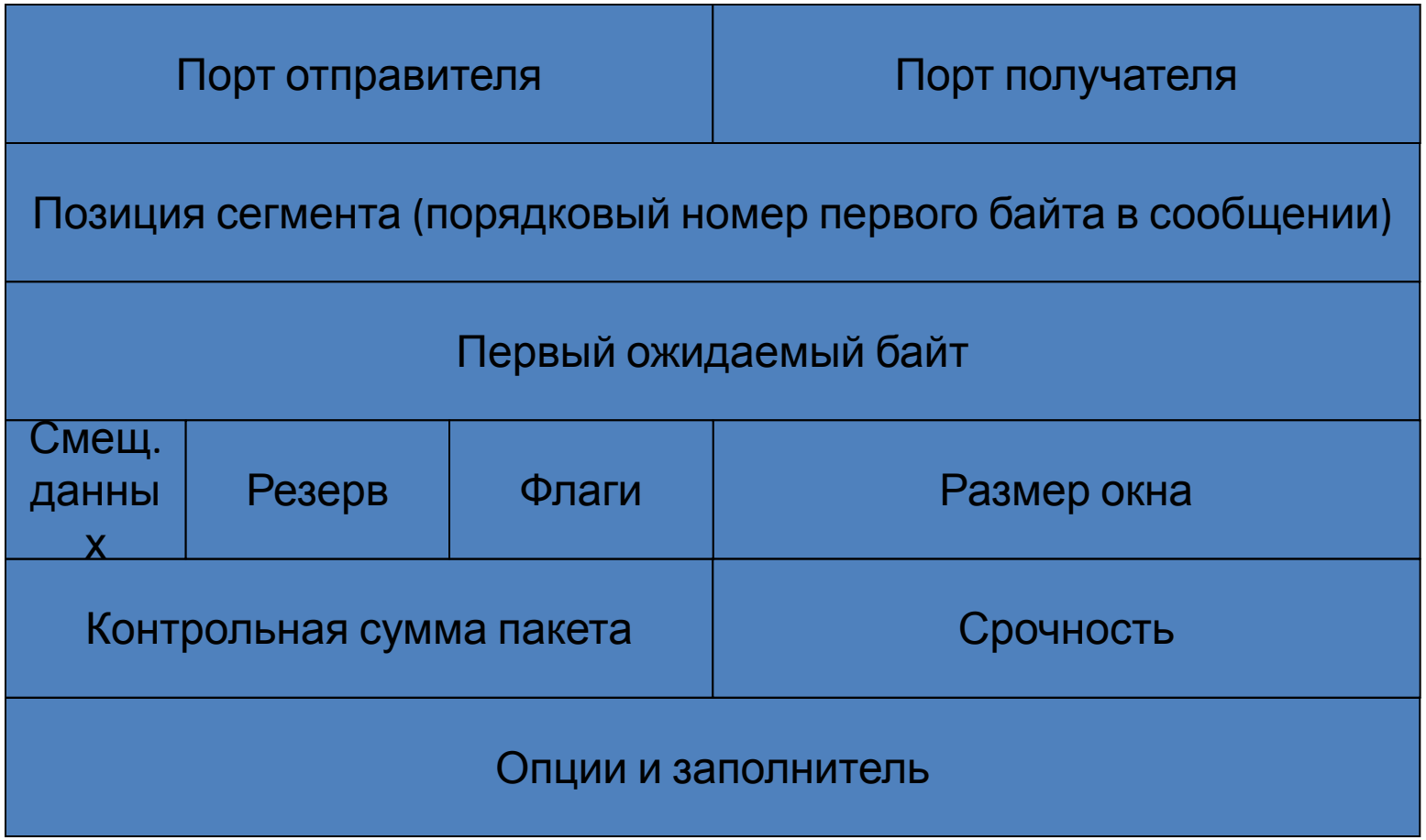
- Протокол TCP (Transmission Control Protocol, Протокол контроля передачи) обеспечивает сквозную доставку данных между прикладными процессами, запущенными на узлах, взаимодействующих по сети. Стандартное описание TCP содержится в RFC-793.
- TCP находится на транспортном уровне стека TCP/IP, между протоколом IP и собственно приложением. Протокол IP занимается пересылкой дейтаграмм по сети, никак не гарантируя доставку, целостность, порядок прибытия информации и готовность получателя к приему данных; все эти задачи возложены на протокол TCP.

TCP: Transmission Control Protocol

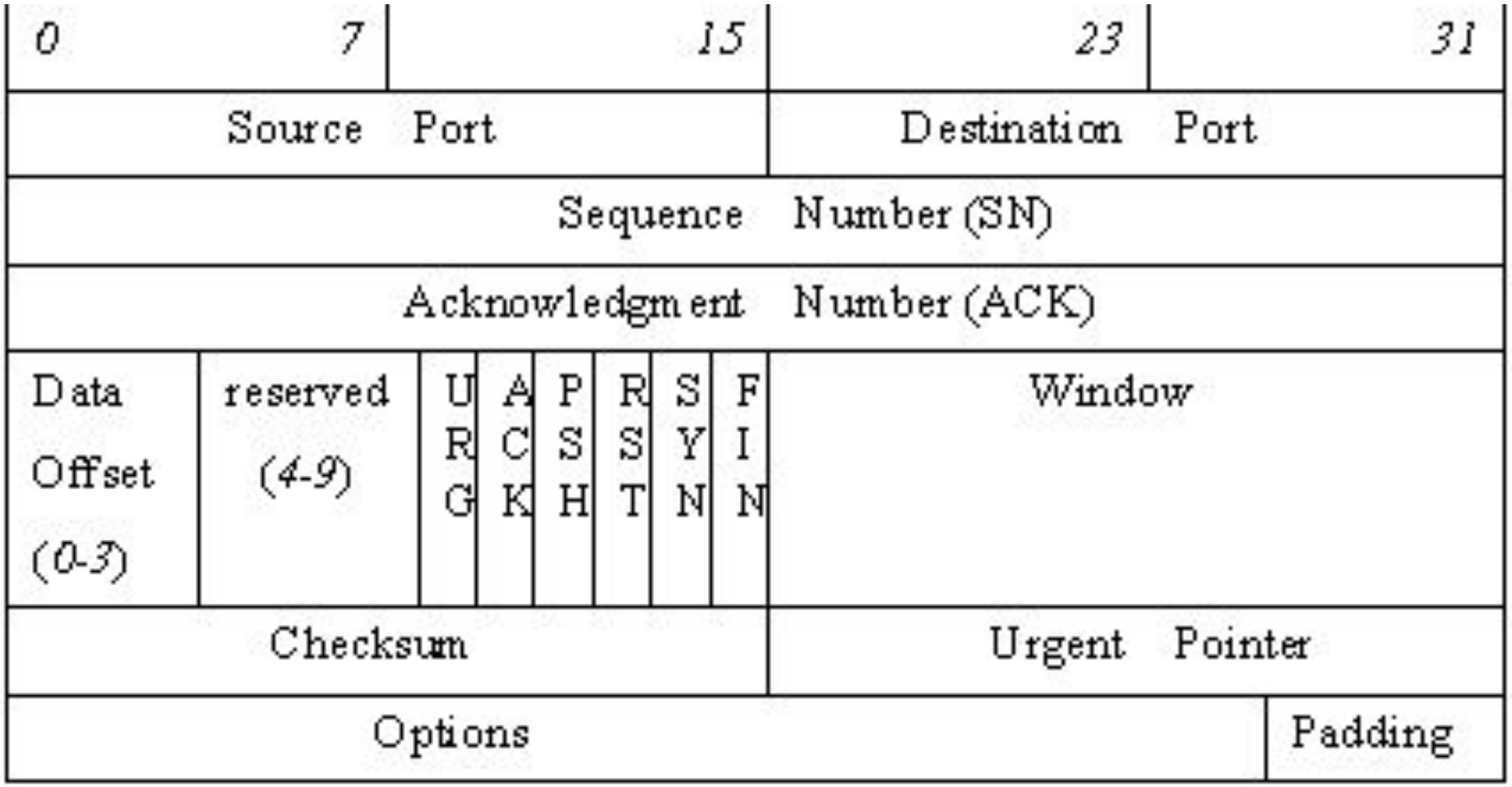
При получении дейтаграммы, в поле Protocol которой указан код протокола TCP , модуль IP передает данные этой дейтаграммы модулю TCP. Эти данные представляют собой TCP-сегмент, содержащий TCP-заголовок и данные пользователя (прикладного процесса). **Модуль TCP анализирует служебную информацию заголовка, 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.

Структура заголовка TCP

4 8 16 32 бита



Заголовок TCP-сегмента



Заголовок TCP-сегмента

- **Source Port** (16 бит), **Destination Port** (16 бит) - номера портов процесса-отправителя и процесса-получателя
- **Data Offset** (4 бита) - длина TCP-заголовка в 32-битных словах.
- **Reserved** (6 бит) - зарезервировано; заполняется нулями
- **Window** (16 бит) - размер окна в октетах

Checksum (16 бит) - контрольная сумма, представляет собой 16 бит, дополняющие биты в сумме всех 16-битовых слов сегмента (само поле контрольной суммы перед вычислением обнуляется).

Контрольная сумма, кроме заголовка сегмента и поля данных, учитывает 96 бит псевдозаголовка, который для внутреннего употребления ставится перед TCP-заголовком. Этот псевдозаголовок содержит IP-адрес отправителя (4 октета), IP-адрес получателя (4 октета), нулевой октет, 8-битное поле "Протокол", аналогичное полю в IP-заголовке, и 16 бит длины TCP сегмента, измеренной в октетах.

Такой подход обеспечивает защиту протокола TCP от ошибшихся в маршруте сегментов. Информация для псевдозаголовка передается через интерфейс "Протокол TCP/межсетевой уровень" в качестве аргументов или результатов запросов от протокола TCP к протоколу IP.

Urgent Pointer (16 бит) - используется для указания длины срочных данных, которые размещаются в начале поля данных сегмента. Указывает смещение октета, следующего за срочными данными, относительно первого октета в сегменте.

Например, в сегменте передаются октеты с 2001-го по 3000-й, при этом первые 100 октетов являются срочными данными, тогда Urgent Pointer = 100.

Протокол TCP не определяет, как именно должны обрабатываться срочные денные, но предполагает, что прикладной процесс будет предпринимать усилия для их быстрой обработки. Поле Urgent Pointer задействовано, если установлен флаг **URG**.

- **Options -** поле переменной длины; может отсутствовать или содержать одну опцию или список опций, реализующих дополнительные услуги протокола TCP. Опция состоит из октета "Тип опции", за которым могут следовать октет "Длина опции в октетах" и октеты с данными для опции.
- Стандарт протокола TCP определяет **три** опции **(типы 0,1,2).**
- Опции типов 0 и 1 ("Конец списка опций" и "Нет операции" соответственно) состоят из одного октета, содержащего значение типа опции. При обнаружении в списке опции "Конец списка опций" разбор опций прекращается, даже если длина заголовка сегмента (Data Offset) еще не исчерпана. Опция "Нет операции" может использоваться для выравнивания между опциями по границе 32 бит.
- Опция типа 2 "Максимальный размер сегмента" состоит из 4 октетов: одного октета типа опции (значение равно 2), одного октета длины (значение равно 4) и двух октетов, содержащих максимальный размер сегмента, который способен получать TCP-модуль, отправивший сегмент с данной опцией.

Опцию следует использовать только в SYN-сегментах на этапе установки соединения.

- **Padding -** выравнивание заголовка по границе 32-битного слова, если список опций занимает нецелое число 32-битных слов. Поле Padding заполняется нулями.

Флаги TCP

URG:	поле срочного указателя задействовано
ACK:	поле подтверждения задействовано
PSH:	функция проталкивания
RST:	перезагрузка данного соединения
SYN:	синхронизация номеров очереди
FIN:	нет больше данных для передачи

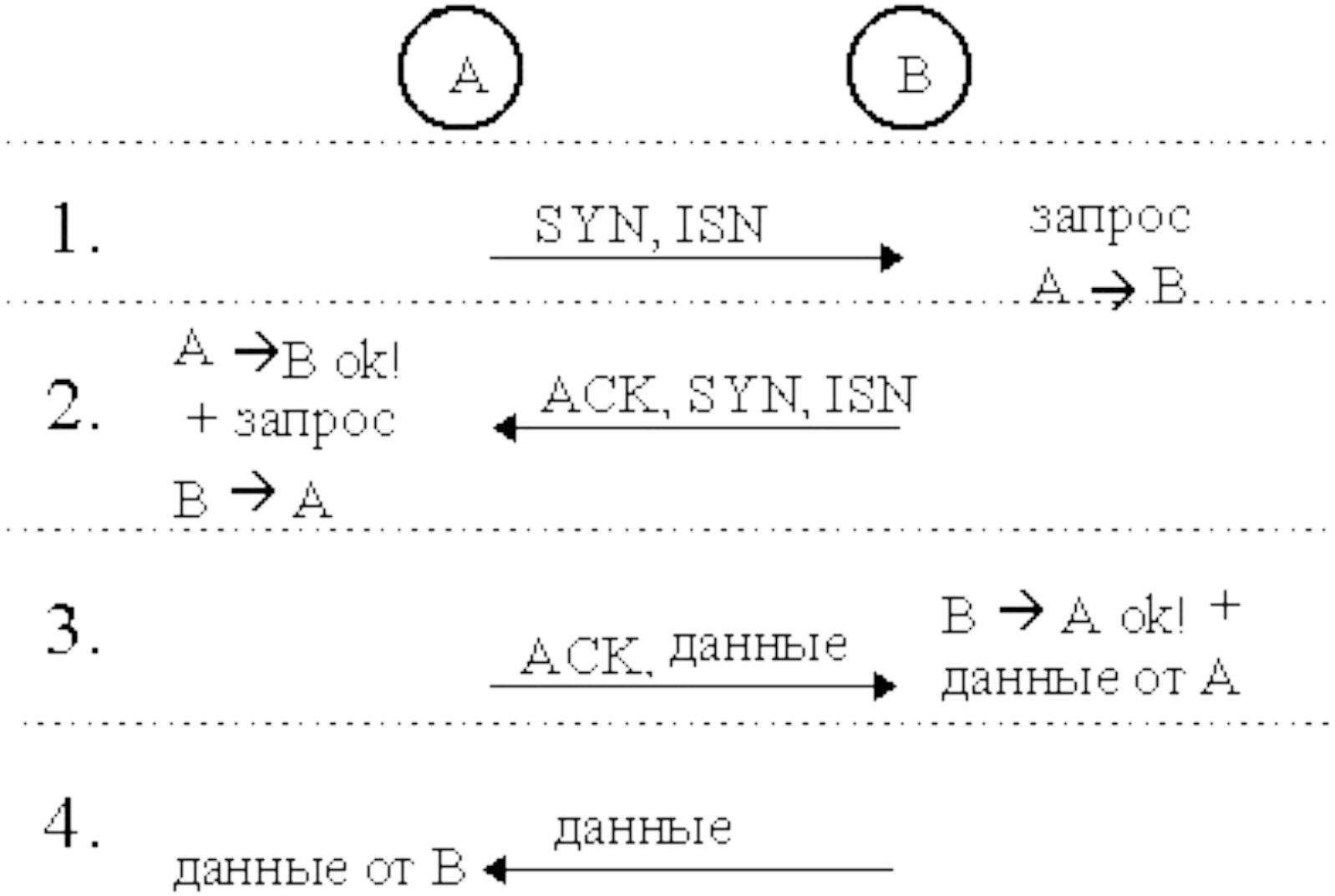
Управление соединениями

- Соединение - это совокупность информации о состоянии потока данных, включающая сокеты, номера посланных, принятых и подтвержденных октетов, размеры окон.
- Каждое соединение уникально идентифицируется в Интернет парой сокетов.
- Соединение характеризуется для клиента именем, которое является указателем на структуру TCB (Transmission Control Block), содержащую информацию о соединении.
- Открытие соединения клиентом осуществляется вызовом функции OPEN, которой передается сокет, с которым требуется установить соединение. Функция возвращает имя соединения. Различают два типа открытия соединения: активное и пассивное.

Управление соединениями

- При активном открытии TCP-модуль начинает процедуру установления соединения с указанным сокетом, при пассивном - ожидает, что удаленный TCP-модуль начнет процедуру установления соединения с указанного сокета. Указание 0.0.0.0:0 в качестве сокета при пассивном открытии означает, что ожидается соединение с любого сокета. Такой способ применяется в демонах - серверах Интернет, которые ждут установления соединения от клиента. Клиент же применяет процедуру активного открытия; сокет при этом формируется из IP-адреса сервера и стандартного номера порта для данного сервиса.
- Закрытие соединения клиентом производится с помощью функции CLOSE, которой передается имя соединения.

Управление соединениями



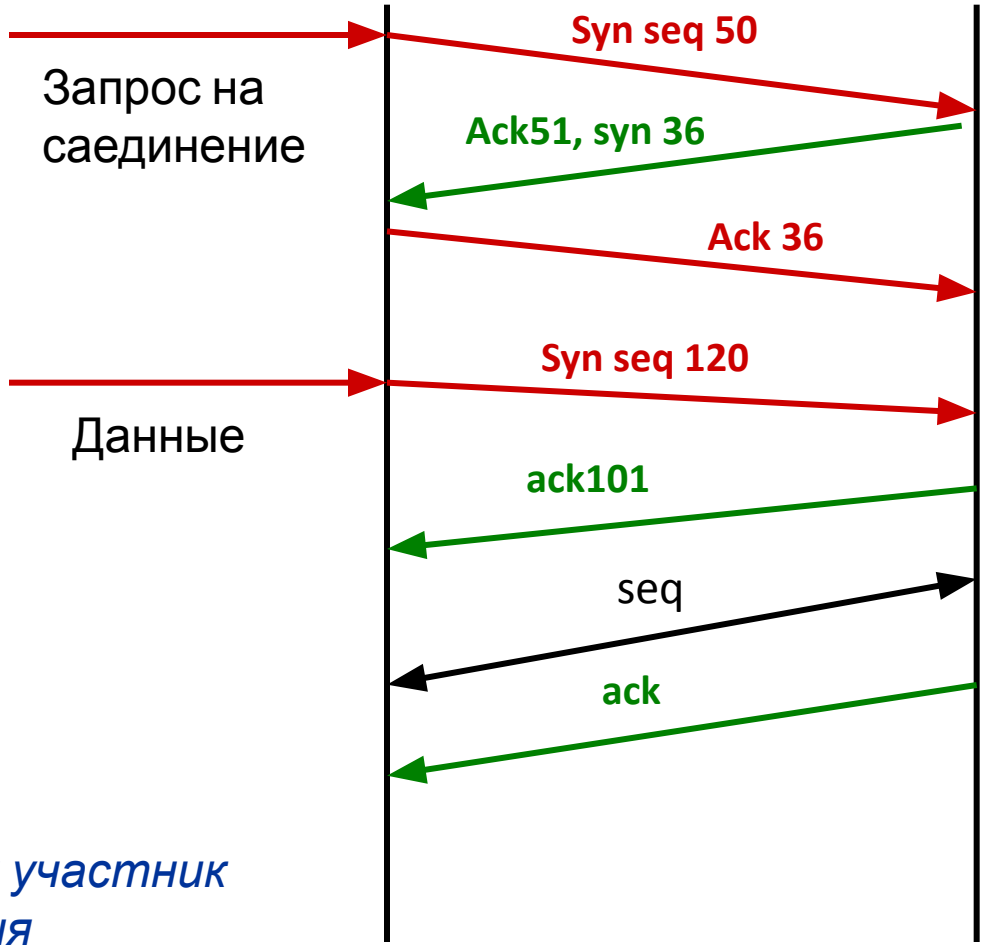
Предположим, узел А желает установить соединение с узлом В. Первый отправляемый из А в В TCP-сегмент не содержит полезных данных, а служит для установления соединения. В его заголовке (в поле Flags) установлен бит SYN, означающий запрос связи, и содержится ISN (Initial Sequence Number - начальный номер последовательности) - число, начиная с которого узел А будет нумеровать отправляемые октеты (например, 0).

В ответ на получение такого сегмента узел В откликается посылкой TCP-сегмента, в заголовке которого установлен бит ACK, подтверждающий установление соединения для получения данных от узла А. Так как протокол TCP обеспечивает полнодуплексную передачу данных, то узел В в этом же сегменте устанавливает бит SYN, означающий запрос связи для передачи данных от В к А, и передает свой ISN (например, 0). Полезных данных этот сегмент также не содержит.

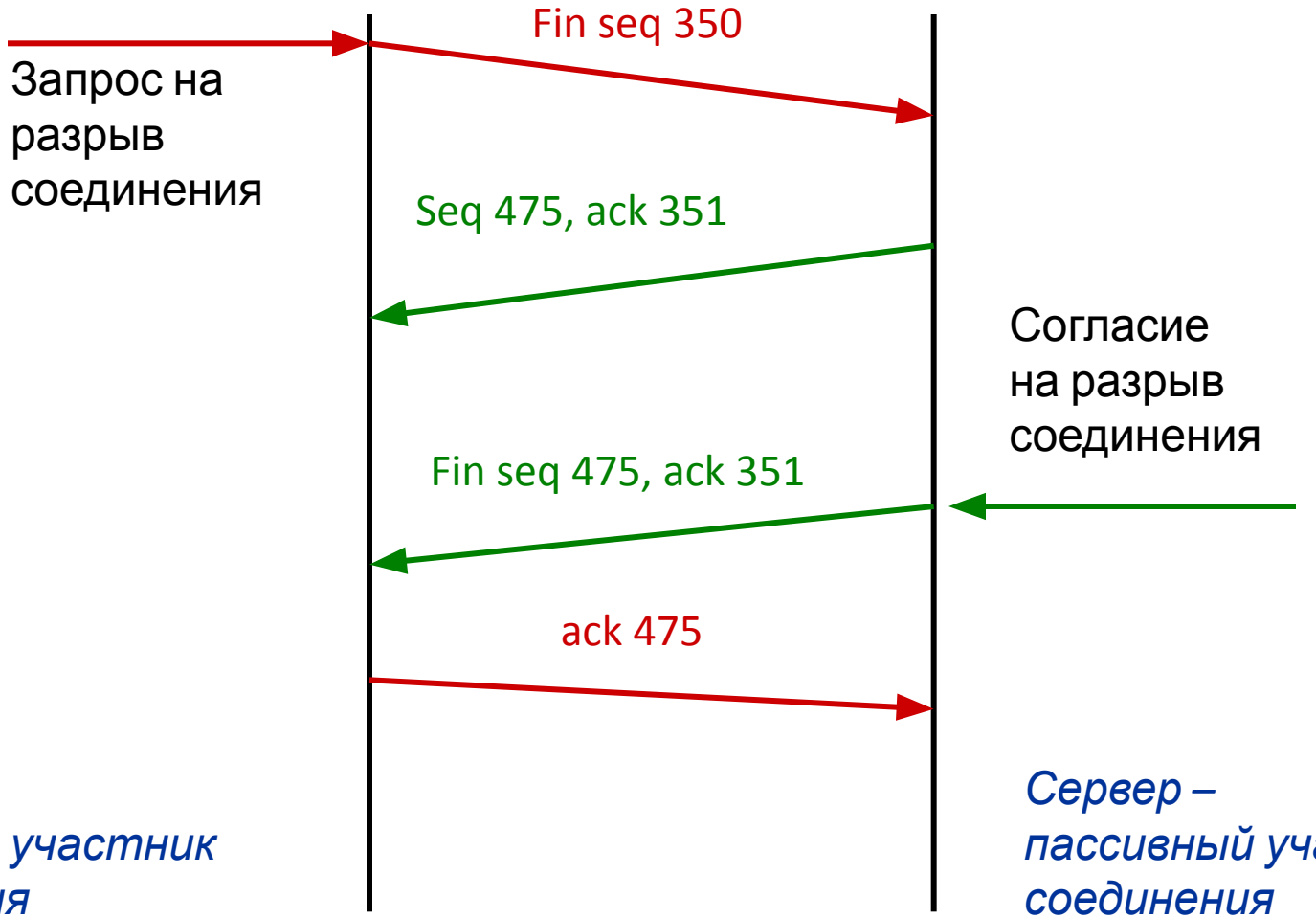
Третий TCP-сегмент в сеансе посылается из А в В в ответ на сегмент, полученный из В. Так как соединение А -> В можно считать установленным (получено подтверждение от В), то узел А включает в свой сегмент полезные данные, нумерация которых начинается с номера ISN(A)+1. Данные нумеруются по количеству отправленных октетов. В заголовке этого же сегмента узел А устанавливает бит ACK, подтверждающий установление связи В -> А, что позволяет хосту В включить в свой следующий сегмент полезные данные для А.

Установление и закрытие соединения

1. Установление соединения



2. Закрытие соединения



Промежуточные состояния соединения

TCP-соединение во время функционирования проходит через ряд промежуточных состояний. Это состояния LISTEN, SYN-SENT, SYN-RECEIVED, ESTABLISHED, FIN-WAIT-1, FIN-WAIT-2, CLOSE-WAIT, CLOSING, LAST-ACK, TIME-WAIT, а также фиктивное состояние CLOSED. (Состояние CLOSED является фиктивным, поскольку оно представляет отсутствие соединения.) Переход из одного состояния в другое происходит в ответ на события, как то: запросы клиента, приход сегментов, истечение контрольного времени.

Определены следующие **запросы процесса-клиента** модулю TCP (с каждым запросом, кроме OPEN, передается имя соединения):

- **ACTIVE-OPEN** - активное открытие соединения;
- **PASSIVE-OPEN** - пассивное открытие соединения
- **SEND** - отправка данных (передается указатель на буфер данных, размер буфера, значения флагов URG и PSH);
- **RECEIVE** - получение данных (передается указатель на буфер данных, размер буфера; возвращается счетчик полученных октетов, значения флагов URG и PSH);
- **STATUS** - запрос состояния соединения;
- **CLOSE** - закрытие соединения (производится досылка всех неотправленных данных и обмен сегментами с битом FIN);
- **ABORT** - ликвидация соединения (уничтожаются блок TCB и все неотправленные данные, посылается сегмент с битом RST).

Состояния соединения

- **CLOSE-WAIT** - процесс, не отправив свой FIN (возможно, не собираясь прекращать соединение), получает чужой FIN; он отправляет ACK на чужой FIN, но при этом, возможно, продолжает отправлять данные.
- **LAST-ACK** - процесс отправил свой FIN, но ранее он уже получил FIN с той стороны и отправил на него ACK; поэтому процесс ожидает чужой ACK на свой FIN для окончательного закрытия соединения.
- **CLOSING** - процесс ранее отправил свой FIN и еще не получил не него подтверждение, но получил чужой FIN (и отправил на него ACK); ждет ACK на свой FIN.
- **TIME-WAIT** - процесс ранее отправил свой FIN и получил на него подтверждение, получил чужой FIN и только что отправил на него ACK; теперь процесс ждет некоторое время (два времени жизни сегмента, обычно 4 минуты) для гарантии того, что та сторона получит его ACK на свой FIN, после чего соединение будет окончательно закрыто.
- **CLOSED** - соединение отсутствует.

Состояния соединения

- **LISTEN** - процесс пассивно ждет запроса со стороны чужих сокетов.
- **SYN-SENT** - процесс отправил свой SYN и ждет чужого SYN.
- **SYN-RECEIVED** - процесс получил чужой SYN, отправил (раньше или только что) свой SYN и ждет ACK на свой SYN.
- **ESTABLISHED** - процесс отправил ACK на чужой SYN, получил ACK на свой SYN; соединение установлено.
- **FIN-WAIT-1** - процесс первый отправил свой FIN и ждет реакцию той стороны; при этом он, возможно, продолжает получать данные.
- **FIN-WAIT-2** - процесс получил ACK на свой ранее отправленный FIN, но не получил чужой FIN; ждет чужой FIN; при этом, возможно, продолжает получать данные.

(FIN-WAIT-1 + получения FIN/ACK-пакета от сервера. Если от сервера FIN-пакет приходит раньше, чем FIN/ACK-пакет (ситуация "одновременного закрытия соединения"), вместо состояния FIN-WAIT-2 устанавливается состояние CLOSING.)

Состояния сеанса TCP	
CLOSED	Начальное состояние узла
LISTEN	Сервер ожидает запросов установления соединения от клиента
SYN-SENT	Клиент отправил запрос серверу на установление соединения и ожидает ответа
SYN-RECEIVED	Сервер получил запрос на соединение, отправил ответный запрос и ожидает подтверждения
ESTABLISHED	Соединение установлено, идёт передача данных
FIN-WAIT-1	Одна из сторон (назовём её узел-1) завершает соединение, отправив сегмент с флагом FIN
CLOSE-WAIT	Другая сторона (узел-2) переходит в это состояние, отправив, в свою очередь сегмент ACK и продолжает одностороннюю передачу
FIN-WAIT-2	Узел-1 получает ACK, продолжает чтение и ждёт получения сегмента с флагом FIN
LAST-ACK	Узел-2 заканчивает передачу и отправляет сегмент с флагом FIN
TIME-WAIT	Узел-1 получил сегмент с флагом FIN, отправил сегмент с флагом ACK и ждёт 2*MSL секунд, перед окончательным закрытием соединения
CLOSING	Обе стороны инициировали закрытие соединения одновременно: после отправки сегмента с флагом FIN узел-1 также получает сегмент FIN, отправляет ACK и находится в ожидании сегмента ACK (подтверждения на свой запрос о разъединении)

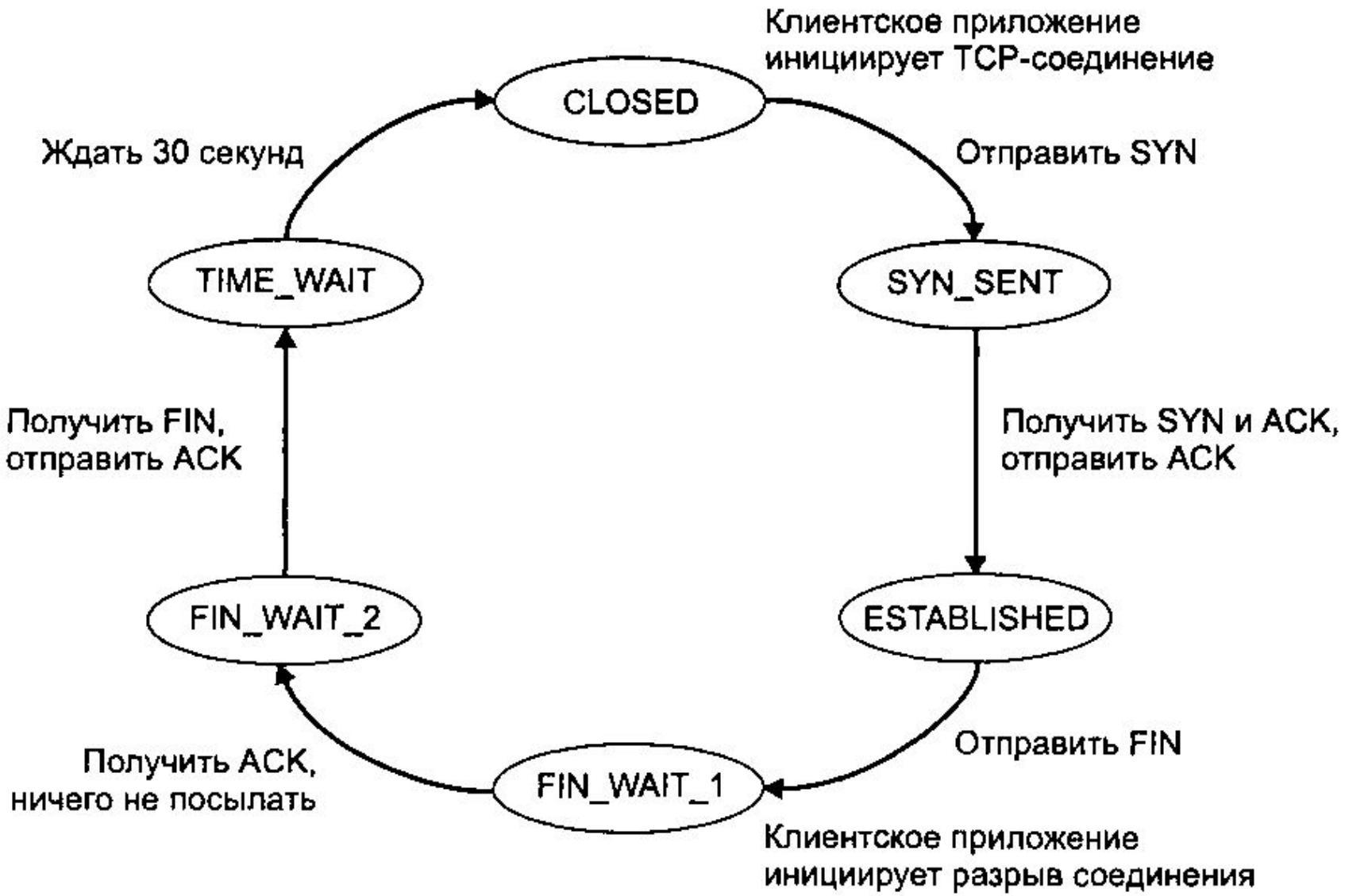


Рис. 3.36. Типичная последовательность состояний клиента TCP

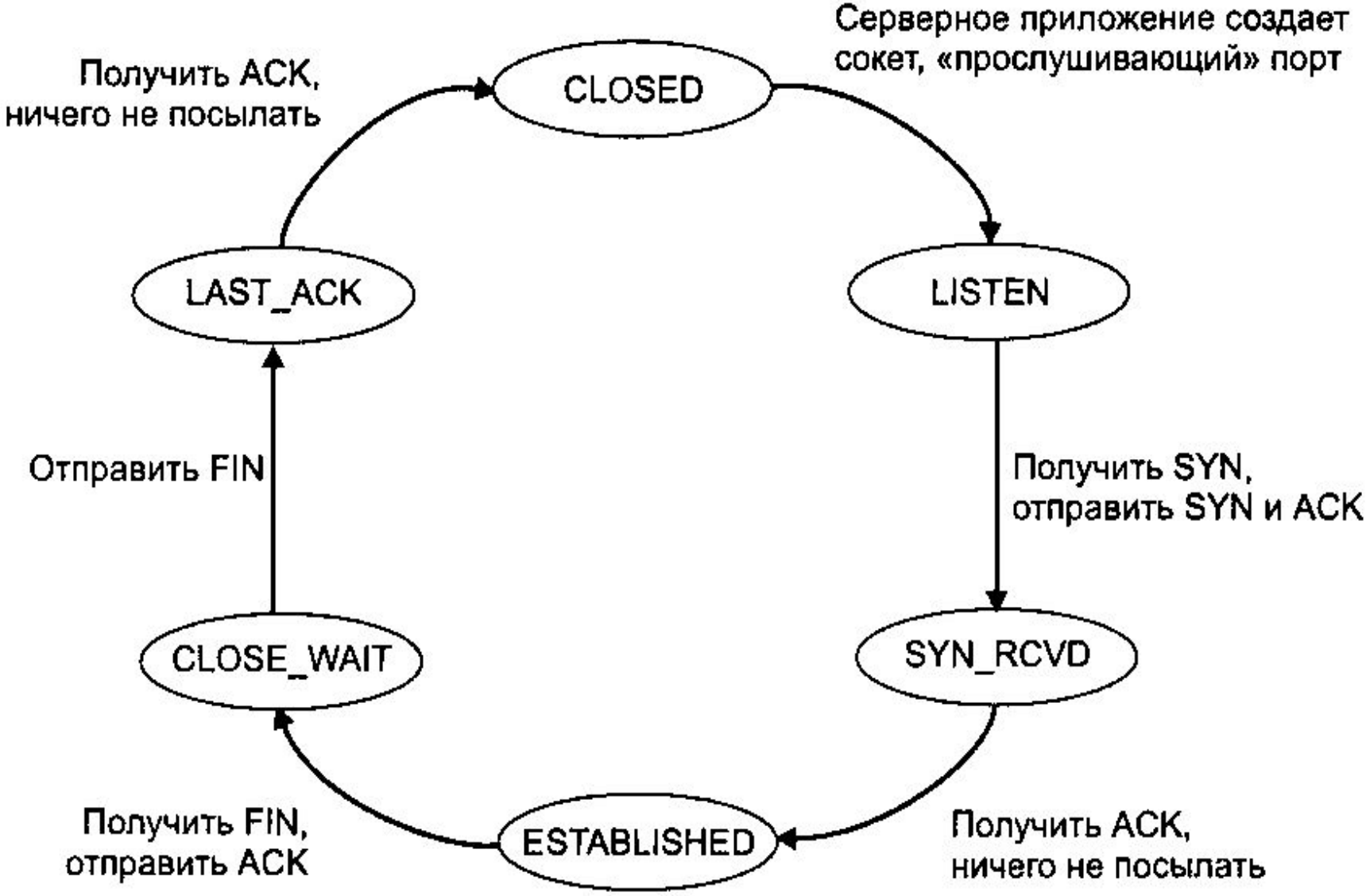


Рис. 3.37. Типичная последовательность состояний TCP-сервера

Установка соединения

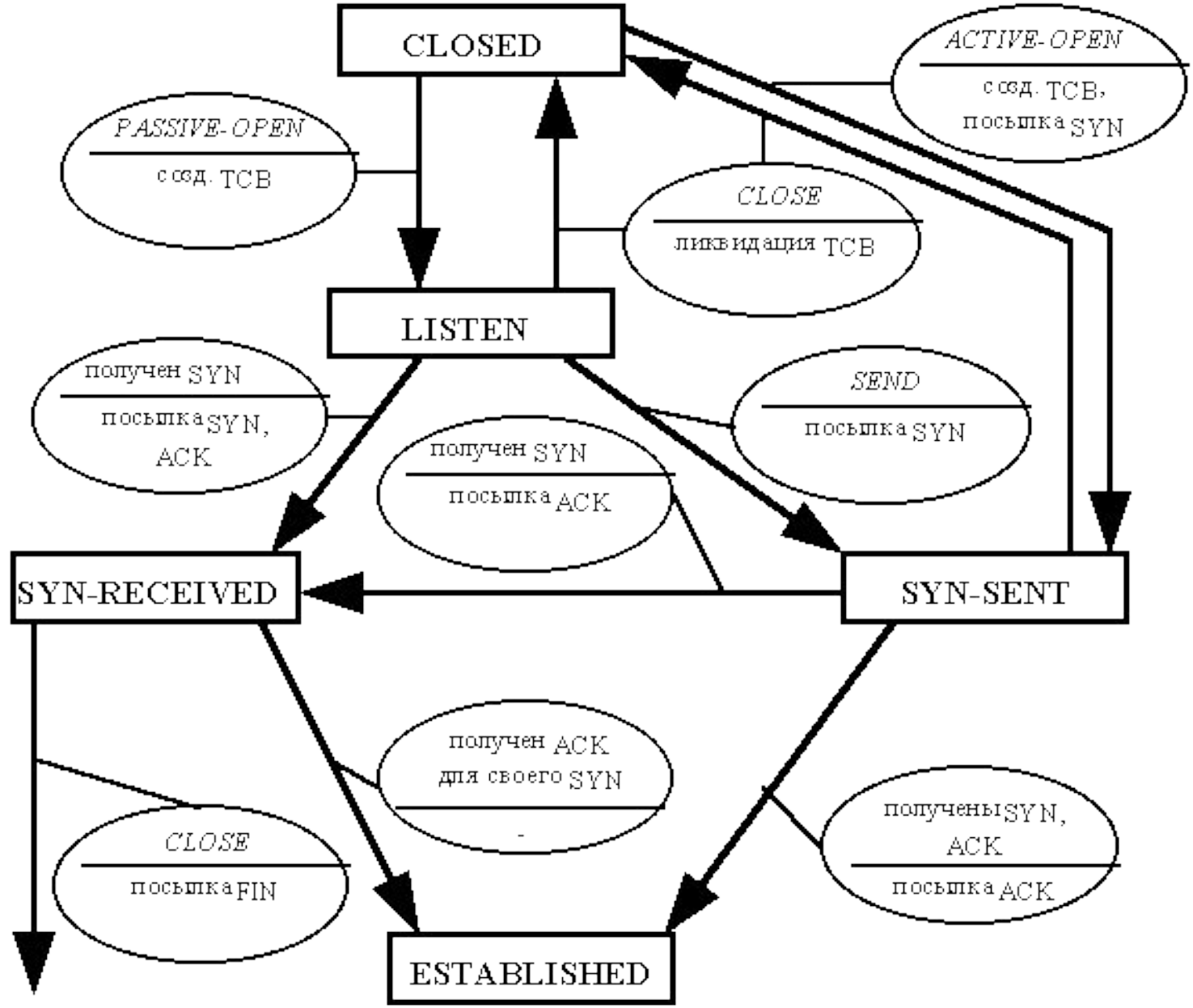
Процесс начала сеанса TCP называется «тройным рукопожатием»:

1. Клиент, который намеревается установить соединение, посылает серверу сегмент с номером последовательности и флагом SYN.
   - Сервер получает сегмент, запоминает номер последовательности и пытается создать сокет (буферы и управляющие структуры памяти) для обслуживания нового клиента.
     - В случае успеха сервер посылает клиенту сегмент с номером последовательности и флагами SYN и ACK, и переходит в состояние SYN-RECEIVED.
     - В случае неудачи сервер посылает клиенту сегмент с флагом RST.

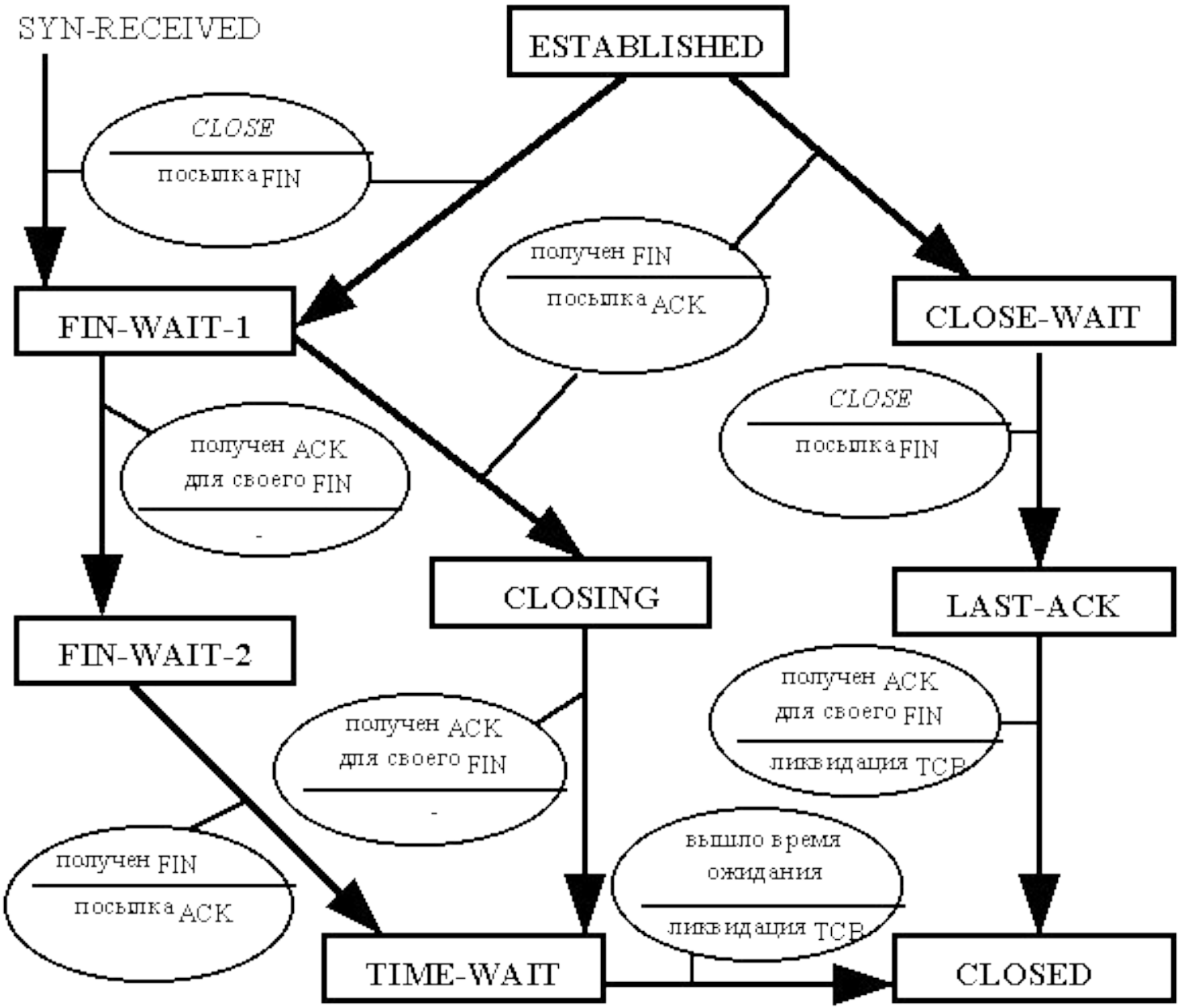
2. Если клиент получает сегмент с флагом SYN, то он запоминает номер последовательности и посылает сегмент с флагом ACK.
   - Если он одновременно получает и флаг ACK (что обычно и происходит), то он переходит в состояние ESTABLISHED.
   - Если клиент получает сегмент с флагом RST, то он прекращает попытки соединиться.
   - Если клиент не получает ответа в течение 10 секунд, то он повторяет процесс соединения заново.

3. Если сервер в состоянии SYN-RECEIVED получает сегмент с флагом ACK, то он переходит в состояние ESTABLISHED.
   - В противном случае после таймаута он закрывает сокет и переходит в состояние CLOSED.

Фаза установления соединения



Фаза закрытия соединения



Функции TCP

Базовая передача данных

- Модуль TCP выполняет передачу непрерывных потоков данных между своими клиентами в обоих направлениях. Клиентами TCP являются прикладные процессы, вызывающие модуль TCP при необходимости получить или отправить данные процессу-клиенту на другом узле.
- Протокол TCP рассматривает данные клиента как непрерывный неинтерпретируемый поток октетов. TCP разделяет этот поток на части для пересылки на другой узел в TCP-сегментах некоторого размера. Для отправки или получения сегмента модуль TCP вызывает модуль IP.
- Немедленное отправление данных может быть затребовано процессом-клиентом от TCP-модуля с помощью специальной функции PUSH, иначе TCP сам будет решать, как накапливать и когда отправлять данные клиента или когда передавать клиенту полученные данные.

Достоверность

- Модуль TCP обеспечивает защиту от повреждения, потери, дублирования и нарушения очередности получения данных.
- Для выполнения этих задач все октеты в потоке данных сквозным образом пронумерованы в возрастающем порядке. Заголовок каждого сегмента содержит число октетов данных в сегменте и порядковый номер первого октета той части потока данных, которая пересылается в данном сегменте. Например, если в сегменте пересылаются октеты с номерами от 2001 до 3000, то номер первого октета в данном сегменте равен 2001, а число октетов равно 1000.
- Номер первого байта в потоке определяется на этапе установления соединения и обозначается ISN+1 . Например, ISN+1=1.
- Также для каждого сегмента вычисляется контрольная сумма, позволяющая обнаружить повреждение данных.
- При удачном приеме октета данных принимающий модуль посылает отправителю подтверждение о приеме - номер удачно принятого октета. Если в течение некоторого времени отправитель не получит подтверждения, считается, что октет не дошел или был поврежден, и он посылается снова. Этот механизм контроля надежности называется PAR (Positive Acknowledgment with Retransmission). В действительности подтверждение посылается не для одного октета, а для некоторого числа последовательных октетов

Разделение каналов

- Протокол TCP обеспечивает работу одновременно нескольких соединений. Каждый прикладной процесс идентифицируется *номером порта*. Заголовок TCP-сегмента содержит номера портов процесса-отправителя и процесса-получателя. При получении сегмента модуль TCP анализирует номер порта получателя и отправляет данные соответствующему прикладному процессу.
- Все распространенные сервисы Интернет имеют стандартизованные номера портов. Например, номер порта сервера электронной почты - 25, сервера FTP - 21. Список стандартных номеров портов можно найти в файле /etc/services (Unix).
- Совокупность IP-адреса и номера порта называется *сокетом*. Сокет уникально идентифицирует прикладной процесс в Интернет. Например, сокет сервера электронной почты на хосте 194.84.124.4 обозначается как 194.84.124.4.25; часто номер порта отделяется двоеточием.

Управление потоком

- Протокол TCP дает средства получателю управлять количеством данных, посылаемых ему отправителем. Это достигается возвратом так называемого "окна" (window) вместе с каждым подтверждением, которое указывает диапазон приемлемых номеров, следующих за номером последнего успешно принятого сегмента. Окно определяет количество октетов, которое отправитель может послать до получения дальнейших указаний.

Работа с соединениями

- Механизмы управления потоком и обеспечения достоверности, описанные выше, требуют, чтобы программы протокола TCP инициализировали и поддерживали определенную информацию о состоянии каждого потока данных. Набор такой информации, включающий сокеты, номера очереди, размеры окон, называется соединением. Каждое соединение уникальным образом идентифицируется парой сокетов на двух концах.
- Если два процесса желают обмениваться информацией, соответствующие программы протокола TCP должны сперва установить соединение (на каждой стороне инициализировать информацию о статусе). По завершении обмена информацией соединение должно быть закрыто, чтобы освободить ресурсы для предоставления другим пользователям.
- Поскольку соединения должны устанавливаться между ненадежными хост-компьютерами и через ненадежную коммуникационную систему Internet, то во избежание ошибочной инициализации соединений используется механизм подтверждения связи с хронометрированными номерами очереди.

Работа с соединениями

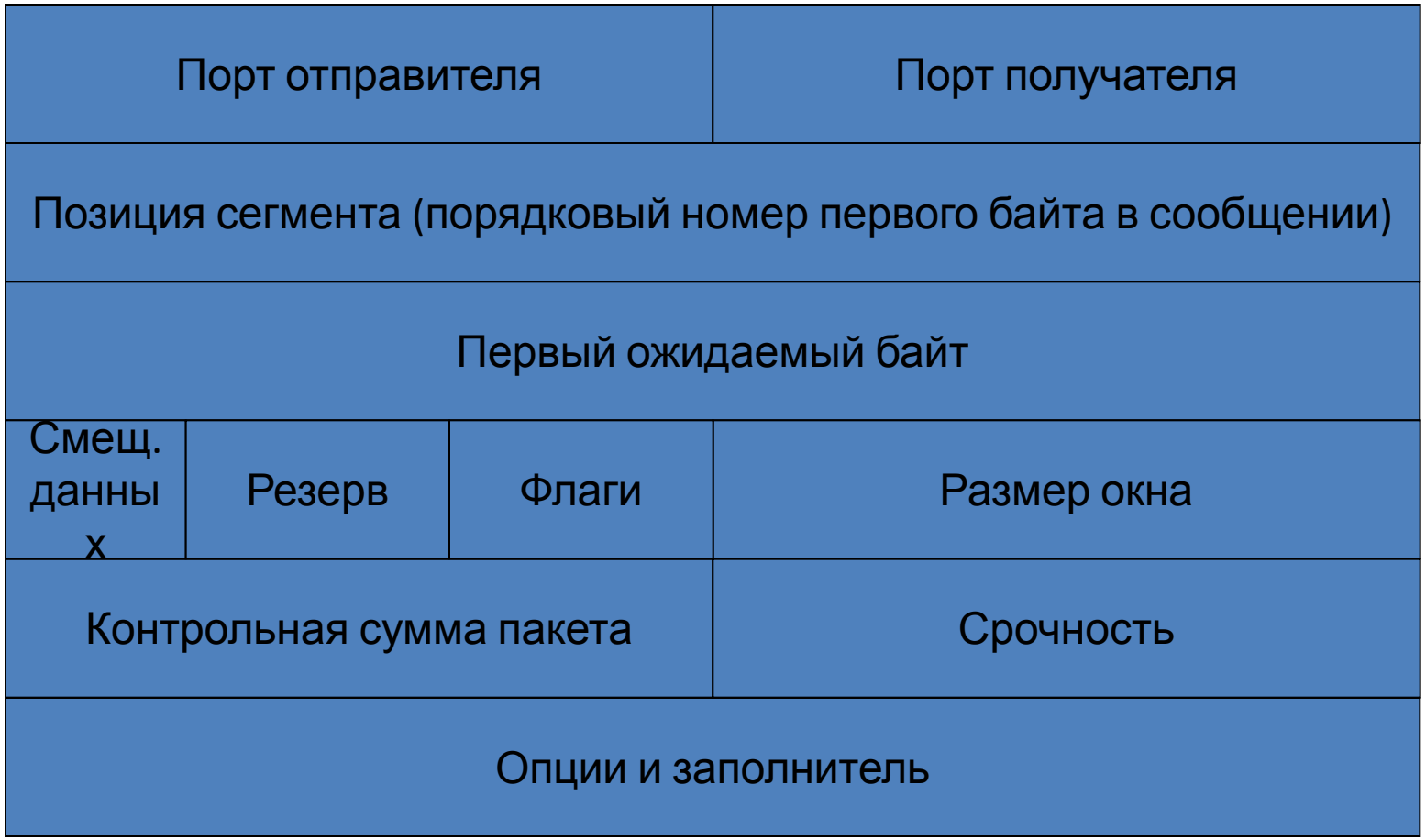
- Механизмы управления потоком и обеспечения достоверности, описанные выше, требуют, чтобы программы протокола TCP инициализировали и поддерживали определенную информацию о состоянии каждого потока данных. Набор такой информации, включающий сокеты, номера очереди, размеры окон, называется соединением. Каждое соединение уникальным образом идентифицируется парой сокетов на двух концах.
- Если два процесса желают обмениваться информацией, соответствующие программы протокола TCP должны сперва установить соединение (на каждой стороне инициализировать информацию о статусе). По завершении обмена информацией соединение должно быть закрыто, чтобы освободить ресурсы для предоставления другим пользователям.
- Поскольку соединения должны устанавливаться между ненадежными хост-компьютерами и через ненадежную коммуникационную систему Internet, то во избежание ошибочной инициализации соединений используется механизм подтверждения связи с хронометрированными номерами очереди.

Приоритет и безопасность

- Пользователи протокола TCP могут затребовать для своего соединения приоритет и безопасность. Предусмотрены принимаемые по умолчанию характеристики соединений, когда такие параметры не требуются.

Структура заголовка TCP

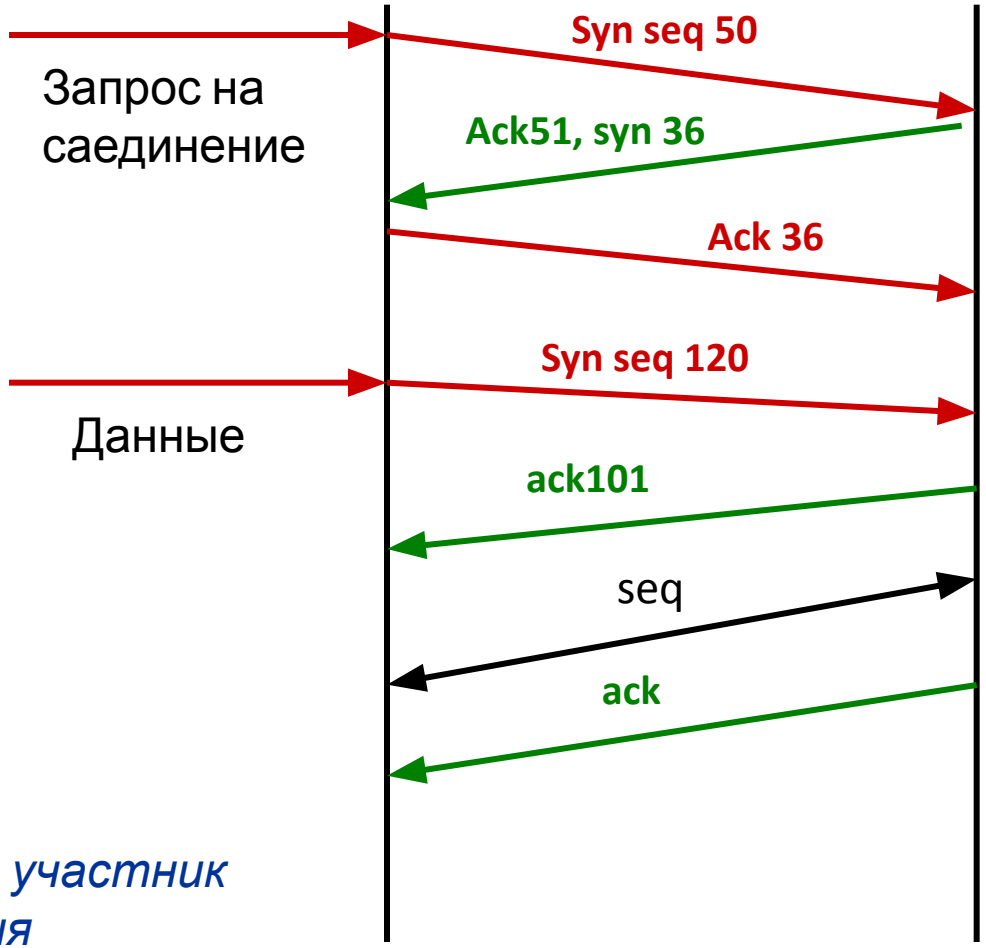
4 8 16 32 бита



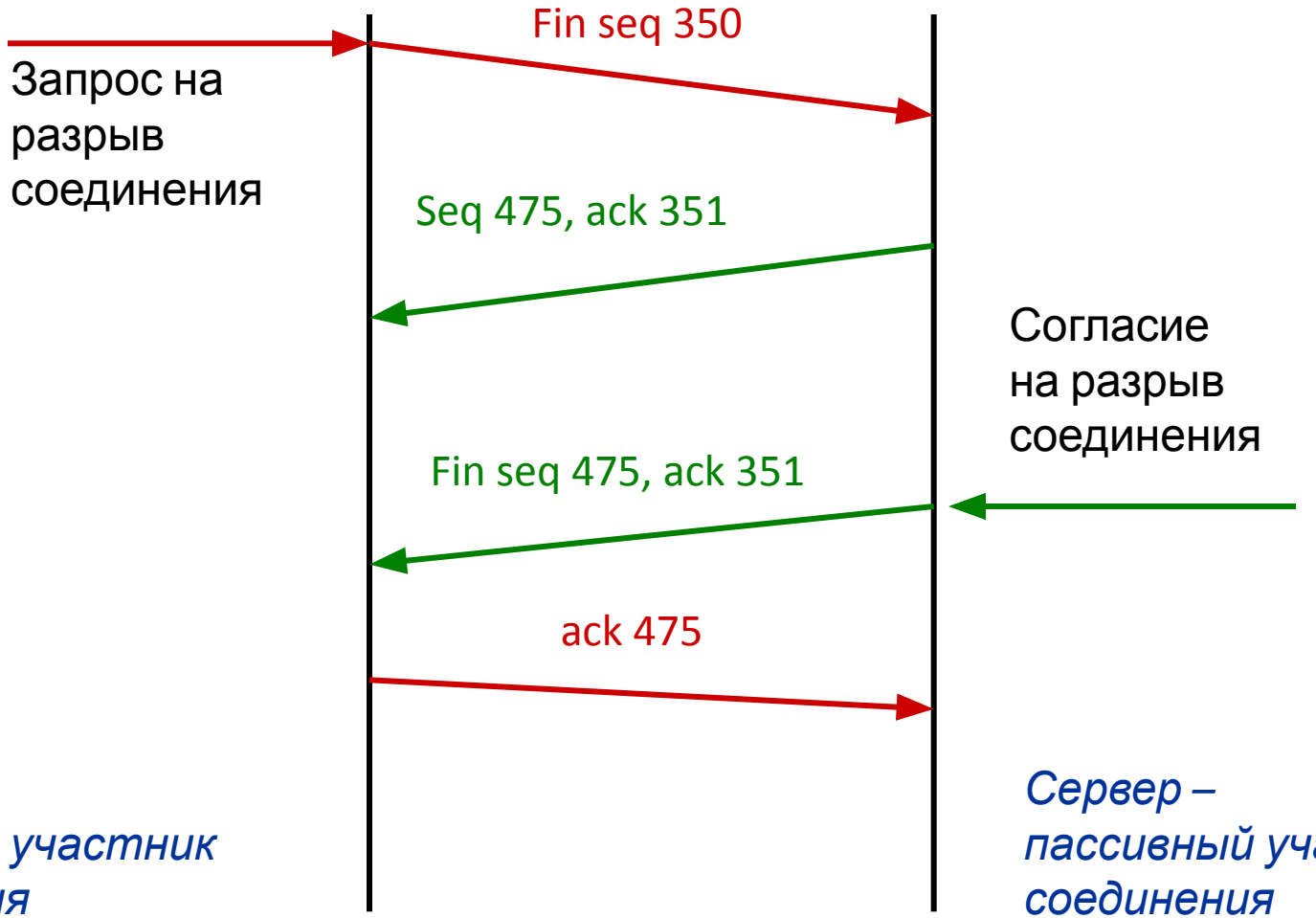
- Контрольная сумма TCP также рассчитывается с использованием псевдозаголовка (поле «тип протокола» 00000110) – структуру см. псевдозагловок UDP.
- TCP-сегмент: поток байтов разбивается на сегменты, каждый из которых передается в одном IP-пакете.
- MSS – максимальный размер сегмента. Определяется через рекомендуемый размер IP-пакетов (MTU).
- Динамическое окно – передача сразу нескольких сегментов (представляемых в виде байтов) до получения подтверждения на них. В частности, используется для реализации механизма медленного старта.
- RTT – полное время доставки пакетов (от момента пересылки в сеть до получения подтверждения).
- ACK – подтверждение (квитанция) о доставке пакета.

Установление и закрытие соединения

1. Установление соединения



2. Закрытие соединения



Sequence Number (SN) (32 бита) - порядковый номер первого октета в поле данных сегмента среди всех октетов потока данных для текущего соединения, то есть если в сегменте пересылаются октеты с 2001-го по 3000-й, то SN=2001.

Если в заголовке сегмента установлен бит SYN (фаза установления соединения), то в поле SN записывается начальный номер (ISN), например, 0. Номер первого октета данных, посылаемых после завершения фазы установления соединения, равен ISN+1.

Acknowledgment Number (ACK) (32 бита) - если установлен бит ACK, то это поле содержит порядковый номер октета, который отправитель данного сегмента желает получить. Это означает, что все предыдущие октеты (с номерами от ISN+1 до ACK-1 включительно) были успешно получены.

Control Bits (6 бит) - управляющие биты; активным является положение “бит установлен”.

- **URG** - поле срочного указателя (Urgent Pointer) задействовано;
- **ACK** - поле номера подтверждения (Acknowledgment Number) задействовано;
- **PSH** - осуществить “проталкивание” - если модуль TCP получает сегмент с установленным флагом PSH, то он немедленно передает все данные из буфера приема процессу-получателю для обработки, даже если буфер не был заполнен;
- **RST** - перезагрузка текущего соединения;
- **SYN** - запрос на установление соединения;
- **FIN** - нет больше данных для передачи