Протокол TCP

Протокол управления передачей информации – **Transmission Control Protocol (TCP)** был разработан для поддержки интерактивной связи между компьютерами. Протокол TCP обеспечивает надежность и достоверность обмена данными между процессами на компьютерах, входящих в общую сеть.

С одной стороны, протокол TCP взаимодействует с прикладным протоколом пользовательского приложения, а с другой стороны – с протоколом, обеспечивающим «низкоуровневые» функции маршрутизации и адресации пакетов, выполняемые протоколом Интернет.

Интерфейс с пользовательской частью состоит из набора вызовов, во многом аналогичных вызовам, которые операционная система обеспечивает для прикладного процесса манипулирования файлами. Например, имеются вызовы для открытия и закрытия соединений, приема и передачи данных через установленные соединения. Также ожидается асинхронный режим связи с прикладными программами. В целом, TCP обеспечивает логическую связь между прикладными процессами, обеспечивая следующие базовые функции:

- Основная передача данных;
- Надежность передачи;
- Контроль потока данных;

- Мультиплексирование;

Далее приведем краткое описание приведенных выше функций:

- **Основная передача данных**

Обеспечивается дуплексная передача данных. Определенное количество октетов упаковывается в сегменты. Передающий пользователь обеспечивает индикацию на передаваемые пользовательские данные.

- **Надежность передачи**

Каждому передаваемому октету распределяется *последовательный номер* (номер октета в последовательности октетов), требующий от TCP приемной части положительного подтверждения (ACK.). Если положительное подтверждение не принято в период интервала таймаута, то данные передаются повторно. В приемнике последовательные номера используются для корректного упорядочивания сегментов, которые могут быть приняты вне последовательности и с целью устранения дублирования информации. Искажения пакетов обрабатываются посредством добавления *контрольной суммы* к каждому переданному сегменту и *проверяемой в приемнике.* При этом искаженные сегменты отбрасываются.

- **Контроль потока**

TCP обеспечивает средства управления приемником общим объемом передаваемых передатчиком данных. Это достигается посредством передачи значения размера «окна»

с каждым *подтверждающим сообщением («ACK»)*, указывающим на диапазон допустимых последовательных номеров, не считая последнего успешно принятого сегмента. Это окно указывает на *допустимое количество октетов, которое передатчик может передать перед приемом последующего разрешения передачи.*

- **Мультиплексирование**

Многие процессы в пределах отдельного хоста могут одновременно использовать услугу TCP соединения. Для этого TCP в пределах каждого хоста обеспечивает множество адресов или портов. Сочлененные с адресами хоста и сети, это множество образует *комплект связи («socket»).* Пара комплектов уникально идентифицируют каждое соединение, а именно, комплект связи может одновременно использоваться множеством соединений. Связь портов с процессами независимо обрабатываются каждым хостом.

- **Соединение**

Когда два процесса желают установить между собой связь, то их подсистемы TCP должны первоначально установить соединение (инициализировать на каждой стороне информацию о статусе). Когда сеанс связи завершается, то соединение завершается или закрывается, чтобы освободить ресурсы для другого использования. Во избежание ошибочных инициализаций соединений, используется механизм «рукопожатия» с

последовательными номерами, основанными на тактовых отсчетах.

- *Надежность передачи* достигается за счет использования последовательных номеров и подтверждений. Номер подтверждения является следующим ожидаемым последовательным номером октета данных на прием. Когда TCP передает сегмент содержащий данные, он помещает его копию в буфер повторной передачи и устанавливает таймер. Когда на соответствующий сегмент поступает подтверждение, то он удаляется из очереди. Если до поступления сегмента истекает таймаут, то сегмент передается повторно.

Подтверждение TCP не гарантирует факт доставки данных к оконечному пользователю, а только означает, что противоположная подсистема TCP взяла на себя ответственность за процесс доставки сообщения.

- **Идентификация соединений**

Идентификация соединения на каждом окончании – *IP адрес* плюс *идентификатор порта.* Таким образом образуется комплект, являющийся уникальным для всех секций сигнального соединения.

Формат заголовка TCP

Сегменты TCP передаются как датаграммы Интернет. Заголовок протокола Интернет переносит несколько информационных полей, включая адреса хоста источника и

пункта назначения пакета. За заголовком TCP следует заголовок Интернет, обеспечивая специфическую для протокола TCP информацию. Это разделение допускает существование протоколов уровня хоста, отличных от TCP.

На нижеследующем рисунке представлен заголовок пакета TCP.

Формат заголовка TCP

Назначение полей протокола является следующим:

- **Последовательный номер**

Последовательный номер первого октета данных в этом сегменте (за исключением присутствия SYN). Если в сообщении присутствует SYN, то последовательный номер

является начальным номером последовательности (ISN), а первым октетом данных является ISN + 1.

- **Номер подтверждения**

Если бит управления «ACK» установлен, то это значение содержит значение следующего последовательного номера передатчика сегмента, ожидаемое на приеме. Этот номер всегда передается, как только устанавливается соединение.

- **Смещение данных (4 бита)**

Число 32-х битовых слов в заголовке TCP. Это значение указывает на начало данных. Заголовок TCP (даже при включении опций) является целочисленным значением размерностью 32 бита.

- **Резервирование (6 бит) (заполняется нулями).**
- **Биты управления (6 бит слева направо)**

URG: значащее поле указателя срочности;

ACK: значащее поле подтверждения;

PSH: функция PUSH;

RST: сброс соединения;

SYN: синхронизировать последовательные номера;

FIN: дальнейшие данные от передатчика отсутствуют;

- **Окно (16 бит)**

Количество октетов данных, начинающееся с «1», указанное в поле подтверждения, которое передатчик этого сегмента желает принять.

- **Проверочная сумма (16 бит)**

Охватывает 96 бит псевдозаголовка, концептуально присоединенного к заголовку TCP. Этот псевдозаголовок включает в себя *адрес источника*, *адрес пункта назначения, протокол* и *длину TCP*, что позволяет защитить TCP от некорректной маршрутизации сегментов. Эта информация переносится в протоколе Интернет и передается через интерфейс TCP/сеть в аргументах или результатах вызовов, посредством TCP по IP.

Длина TCP является *длиной заголовка TCP* плюс *длиной данных в октетах.*

- **Указатель срочности (16 бит)**

Это поле связывает текущее значение указателя срочности, как положительное смещение от последовательного номера в этом сегменте. Указатель срочности указывает на *последовательный номер октета,* следующий за срочными данными. Это поле интерпретируется только в сегментах с установленным битом управления «URG».

Терминология

- **Запись по соединению**

сохраняется в *блоке управления передачей (TCB)*, который включает в себя номера локального и удаленного комплекта, информацию по безопасности и прохождению соединения, указатели на *пользовательские буферы*

передачи и приема, указатели на *очередь повторной передачи* и на *текущий сегмент.*

- **Переменные последовательности передачи**

SND.UNA – неподтвержденная передача;

SND.NXT - последующая передача;

SND.WND – передача окна;

SND.UP – передача указателя срочности;

SND.WL 1 – последовательный номер сегмента, использованный для последнего обновления окна;

SND.WL 2 - номер подтверждения сегмента, использованный для последнего подтверждения окна;

ISS – начальный последовательный номер передачи;

- **Переменные последовательности приема**

RCV.NXT – прием последующего сегмента;

RCV.WND – окно приема;

RCV.UP – прием указателя срочности;

IRS – начальный последовательный номер на прием;

Текущие переменные сегмента

SEG.SEQ – последовательный номер сегмента;

SEG.ACK – номер подтверждения сегмента;

SEG.LEN – длина сегмента;

SEG.WND – окно сегмента;

SEG.UP – указатель срочности сегмента;

SEG.PRC – значение прохождения сегмента;

Краткое описание состояний:

- **LISTEN –** отражает ожидание запроса соединения от любой удаленной системы TCP и порта;
- **SYN-SENT** – отражает *ожидание* согласующего запроса соединения после передачи запроса соединения;
- **SYN-RECEIVED –** отражает *ожидание подтверждения* согласующего запроса соединения после того, как оба участника соединения приняли и передали запрос соединения;
- **ESTABLISHED –** отражает открытое соединение, принятые данные могут быть доставлены к пользователю. Нормальное состояние для фазы передачи данных соединения.
- **FIN-WAIT-1 –** отражает ожидание запроса завершения соединения от удаленной подсистемы TCP, или подтверждение ранее переданного запроса завершения соединения.

- **FIN-WAIT-2-** отражает ожидание запроса завершения соединения от удаленной подсистемы TCP.
- **CLOSE-WAIT –** отражает ожидание запроса завершения соединения от удаленной подсистемы TCP.
- **CLOSING –** отражает ожидание подтверждения запроса завершения соединения от удаленной подсистемы TCP.
- **LAST-ACK –** отражает ожидание подтверждения ранее переданного к удаленной подсистеме TCP запроса завершения соединения, которое включает в себя подтверждение его запроса завершения соединения.

Пользовательскими вызовами являются «OPEN», «SEND», «RECEIVE», «CLOSE», «ABORT» и «STATUS».

Диаграмма переходов приведенных выше состояний приведена на нижеследующем рисунке «Диаграмма переходов состояний процесса установления TCP соединения».

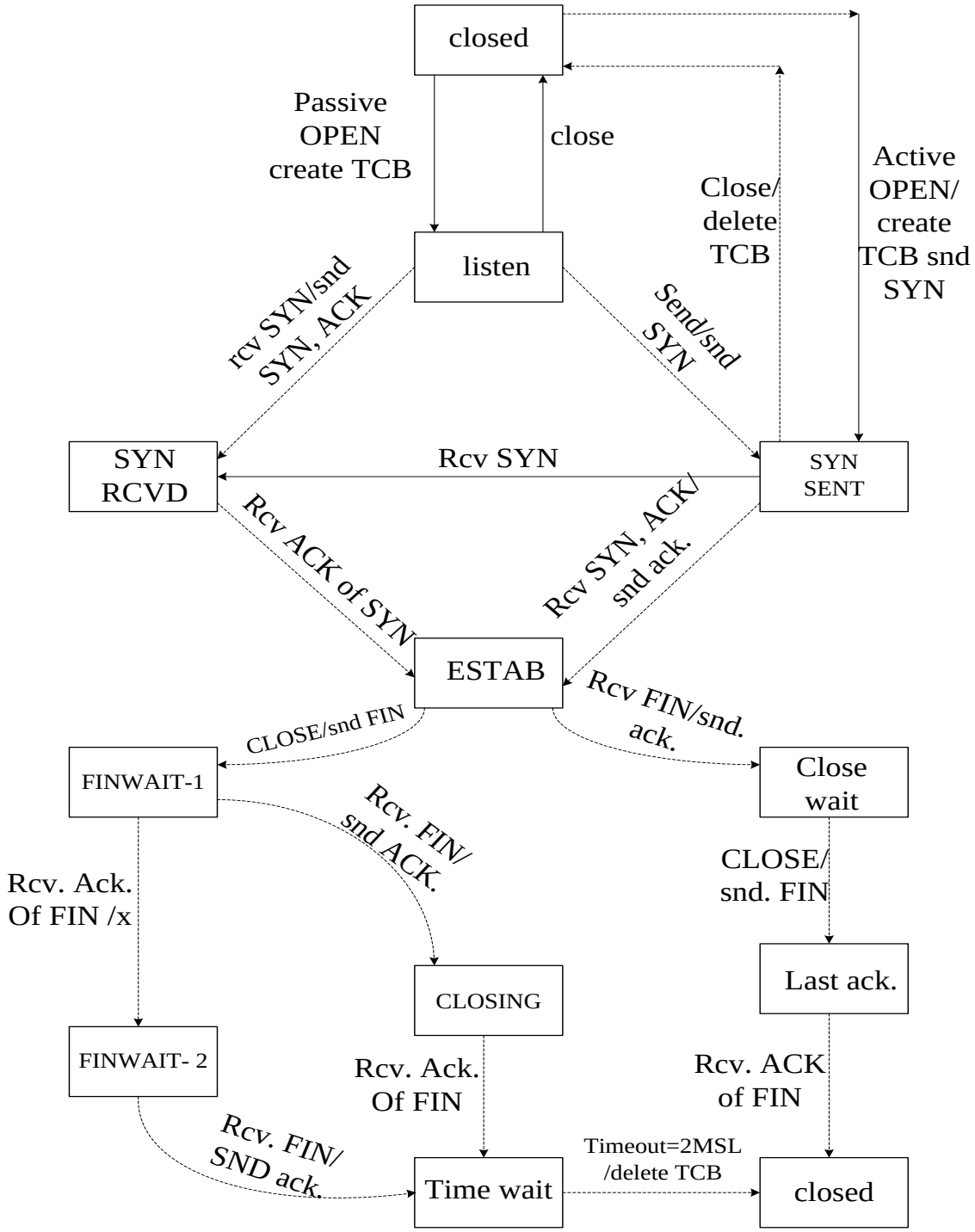


Диаграмма переходов состояний TCP соединения

Порядок обработки октетов и последовательных порядковых номеров

Нумерация октетов в пределах сегмента заключается в том, что первый октет данных, следующих за заголовком,

имеет наименьший номер, а следующие октеты нумеруются последовательно.

Действия, производимые TCP над последовательными номерами являются следующими:

a) Определение, что подтверждение относится к некоторому последовательному номеру, переданному но еще не подтвержденному.

b) Определение, что все последовательные номера, занятые сегментом, были подтверждены (например, чтобы удалить сегмент из очереди повторной передачи).

c) Определение, что входящий сегмент содержит последовательные номера, которые ожидаются (например, сегмент перекрывает «окно» на прием);

Для обработки подтверждений TCP используются следующие сравнения:

SND.UNA – старейший неподтвержденный последовательный номер;

SND.NXT – последующий последовательный номер на передачу;

SEG.ACK – подтверждение от принимающей подсистемы TCP (следующий последовательный номер, ожидаемый от противоположной TCP);

SEG.SEQ – первый последовательный номер сегмента;

SEG.LEN – число октетов, занятых данными в сегменте (считая SYN и FIN);

SEG.SEQ + SEG.LEN-1 – последний последовательный номера сегмента;

Новое подтверждение (названное «преемлимое подтверждение») является подтверждением, для которого удовлетворяется приведенное ниже неравенство:

$$SND.UNA < SEG.ACK \leq SND.NXT$$

Иными словами, старейший неподтвержденный последовательный номер должен быть меньше следующего последовательного номера, ожидаемого от удаленной TCP, который, в свою очередь, менее или равен последующего последовательного номера на передачу.

Сегмент в очереди повторной передачи полностью подтверждается, если сумма его последовательного номера и длины *менее или равна* значению подтверждения во входящем сегменте.

При приеме данных требуются следующие сравнения:

RCV.NXT – следующий последовательный номер, ожидаемый во входящих сегментах, *являющийся левой и нижней границей окна на прием;*

RCV.NXT + RCV.WND -1 – последний значащий номер, ожидаемый по входящему сегменту, являющейся правой или верхней границей окна на прием;

SEG.SEQ – первый последовательный номер занятый входящим сегментом;

SEG.SEQ + SEG.LEN - 1 – последний значащий номер, занятый входящим сегментом.

Сегмент предназначается для занятия части пространства достоверной последовательности на прием, если

$RCV.NXT \leq (SEG.SEQ + SEG.LEN - 1) < RCV.NXT + RCV.WND$

Таким образом, следующий ожидаемый номер на прием меньше или равен последнему последовательному номеру в сегменте, который, в свою очередь, совокупному значению последнего последовательного номера на прием и значению размера окна на прием.

Форматы пользовательских команд

Открытие соединения

OPEN (local port, foreign socket, active/passive [timeout][,precedence] [security/compartment][,options]) -> local connection name;

Допускается, что локальная TCP информирована о идентификации обслуживаемых процессов и проверяет уполномоченность процессов использовать определенное

соединение. В зависимости от реализации TCP, локальная вычислительная сеть и идентификаторы TCP для адреса источника будут обеспечиваться или TCP или протоколом нижнего уровня, например, IP.

Если флаг «active/passive» установлен на «passive», то это вызов «LISTEN» для входящего соединения. Пассивное открытие соединения может иметь, либо полностью определенный удаленный комплект для ожидания определенного соединения или неопределенный удаленный комплект для ожидания любого вызова.

Полностью определенный пассивный вызов может быть преобразован в активный, посредством последующего выполнения «SEND».

Блок управления передачей (TCB) создается и частично наполняется данными команды «OPEN». При активной команде «OPEN» TCP сразу начнет *процедуру синхронизации* (т.е. установления) этого соединения. *Таймаут,* если определено его значение, позволяет вызывающему абоненту установить таймаут для всех данных, предоставляемых TCP. Если за период действия таймаута, доставка данных к пункту назначения завершилась неудачей, то TCP прерывает соединение. Установленное значение таймаута равно по умолчанию 5 мин.

TCP и некоторые компоненты операционной системы верифицируют полномочия пользователя открыть соединение с определенными параметрами *«precedence»* или *«security/compartment».* Если данные параметры отсутствуют, то в вызове «OPEN» задаются значения по

умолчанию. Пользователю TCP на передачу приведенной выше команды OPEN ( ), возвращается наименование локального соединения *(local connection name).*

Передача (SEND)

Формат пользовательской команды SEND является следующим: *SEND (имя локального соединения, адрес буфера, подсчет байтов, флаг «Push», флаг «urgent» [, таймаут]).* Этот вызов обуславливает передачу данных по указанному соединению, содержащихся в указанном пользовательском буфере. Если соединение было открыто, то сообщение «SEND» рассматривается как ошибка. Для некоторых реализаций пользователям разрешается первоначальная передача «SEND». В этом случае автоматически реализуется передача сообщения «OPEN». Если вызывающий процесс не уполномочен использовать это соединение, то в ответ на передачу команды возвращается ошибка.

Если установлен флаг «PUSH», то данные должны быть немедленно переданы к приемнику и бит «PUSH» будет установлен в последний сегмент TCP, созданный от буфера. Если флаг «PUSH» не установлен, то данные могут быть комбинированы с последующими сообщениями «SEND» для эффективности передачи.

Если установлен флаг «URGENT», сегменты переданные к TCP назначения будут иметь установленный *флаг срочности.* Принимающая TCP будет сигнализировать об условиях срочности принимающему процессу, если

указатель срочности указывает, что данные следующие за указателем срочности не были использованы принимающим процессом. Целью передачи TCP-B индикации срочности является стимуляция приемника обработать срочные данные и указать приемнику на момент приема всех известных к настоящему моменту времени срочных данных.

Если в сообщении «OPEN» не был определен никакой удаленный комплект, но соединение установлено (например, в случае спецификации соединения в состоянии «LISTENING», при поступлении удаленного сегмента к локальному комплекту), то назначенный буфер передается к удаленному комплекту. *Адрес удаленного комплекта* не обязательно должен быть явно определен в вызове «SEND», однако, если сетью реализуется попытка выдачи «SEND», перед определением удаленного комплекта будет возвращена ошибка.

Для определения статуса соединения, пользователи могут использовать операцию «STATUS». Если определен таймаут, то текущий таймаут пользователя заменяется существующим. В простейшей реализации, «SEND» не может возвращать управление передающему процессу до завершения передачи или превышения таймаута. Однако, этот простой метод может привести к блокировкам (например, обе стороны соединения могли бы осуществить попытку передачи «SEND» перед активацией каждым окончанием операции «RECEIVE» и предполагает слабые технические характеристики). Поэтому данный метод не рекомендуется. Неявно допускается асинхронный пользовательский интерфейс, в котором «SEND» позже

извлекает некоторый тип «SIGNAL» или псевдо прерывания от обслуживающего TCP. Альтернативой является немедленный возврат отклика. Например, SEND мог бы возвращать немедленное локальное подтверждение, даже если даже если переданный сегмент не был подтвержден удаленной TCP. В реализациях этого типа (синхронных), там еще будут некоторые асинхронные сигналы, но они будут относиться к самому соединению, а не к определенным сегментам или буферам.

Команда «принять» («RECEIVE»)

Команда «*RECEIVE» имеет следующий формат:*

RECEIVE (local connection name, buffer address, byte count, urgent flag, push flag)

Эта команда распределяет приемный буфер, ассоциированный с определенным соединением. Если данной команде не предшествует команда «OPEN», или вызывающий процесс не уполномочен использовать это соединение, то возвращается ошибка.

В простейшей реализации TCP, управление могло бы не возвращаться к вызывающей программе до момента полного заполнения буфера, или события некоторой ошибки, но эта схема очень подвержена блокировкам. В более сложных реализациях допускается одновременная выдача нескольких неподтвержденных сообщений «RECEIVE». Если для заполнения буфера поступает

достаточно данных, перед видимостью «PUSH», флаг «PUSH» не устанавливается в отклик на «RECEIVE». Буфер заполняется максимальным количеством данных. Если «PUSH» видим перед заполнением буфера, то буфер возвращается частично заполненным и индицируется PUSH.

Команда «Закрыть» (CLOSE)

Команда «CLOSE» имеет следующий формат: CLOSE (имя локального соединения). Эта команда обуславливает закрытие локального соединения. Если соединение не открыто, или вызывающий процесс не авторизован использовать это соединение, то возвращается ошибка. При этом все переданные к этому моменту команды SEND будут подтверждены и при необходимости переданы повторно до успешного завершения их обслуживания, насколько это позволяет управление потоком. Сообщение «CLOSE» интерпретируется сетью, как *«Данные для дальнейшей передачи отсутствуют»,* но не интерпретируются сетью, как *«Запрет дальнейшего приема данных».*

Может иметь место ситуация, когда закрывающая сеанс связи сторона не может избавиться от ее данных перед завершением таймаута. В этом случае, сообщение «CLOSE» сменяет сообщение «ABORT» и управление передается TCP, закрывающей соединение. Пользователь может закрыть соединение в любое время по своей собственной инициативе, или же в отклик на различные подсказки от

TCP, например, «Выполнено удаленное задание», «превышен таймаут передачи», «пункт назначения не доступен».

Попытки повторно открыть соединение перед тем, как TCP передаст отклик на команду «CLOSE», вызовет ошибочный отклик.

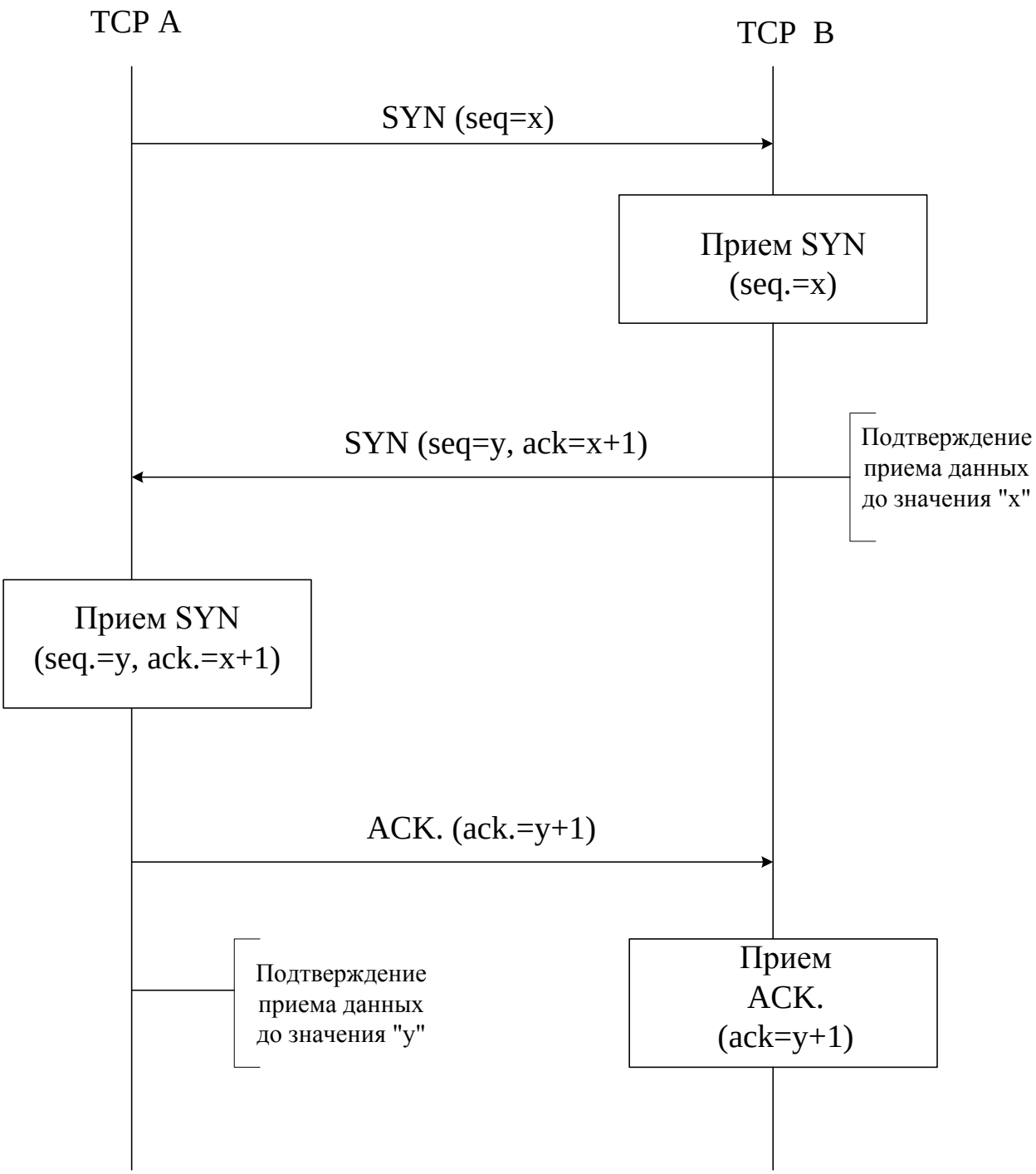
Команда «STATUS»

Команда «STATUS» имеет следующий формат:

STATUS (local connection name) ---→ данные статуса;

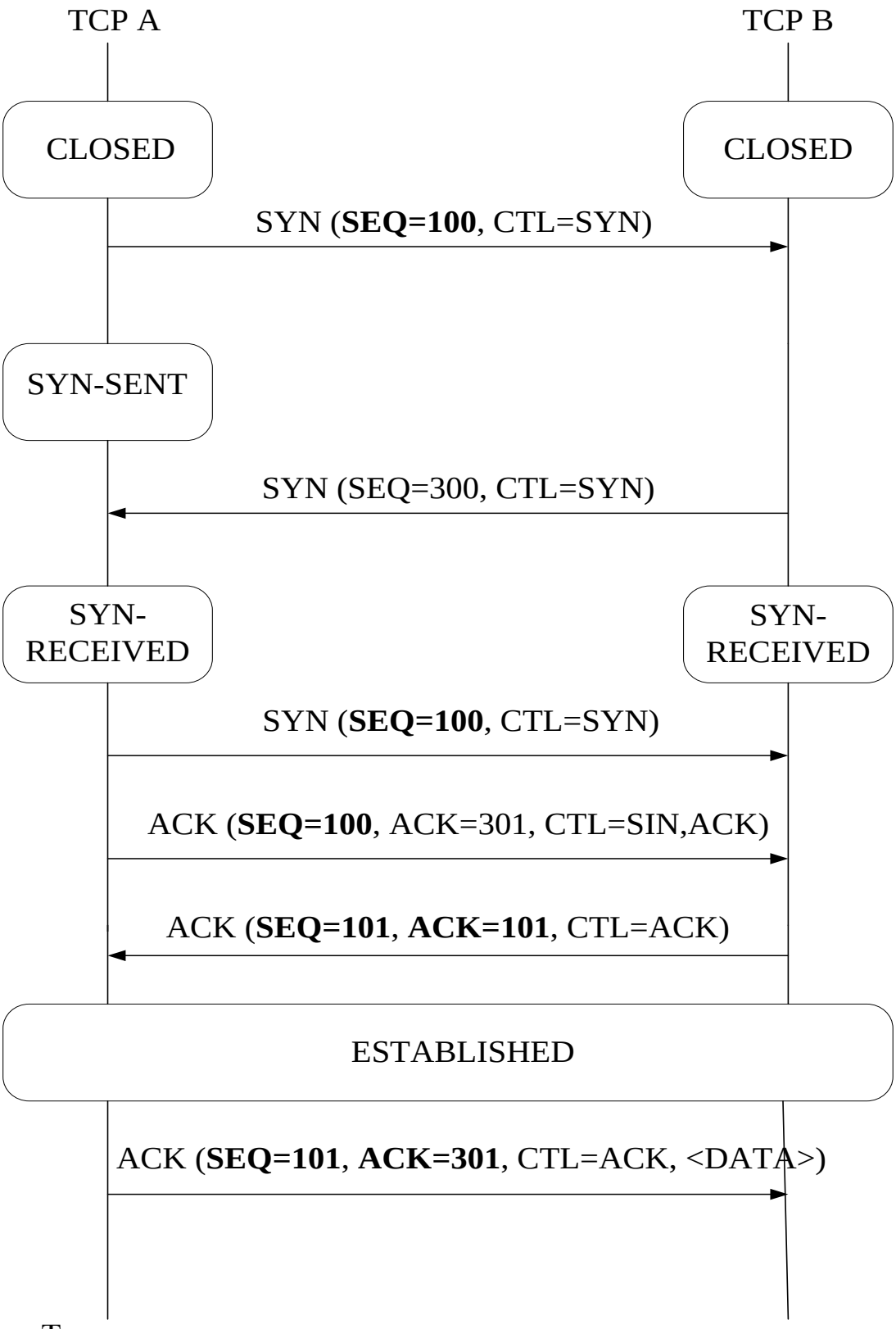
Эта команда возвращает блок данных, содержащий информацию о статусе (местный комплект, удаленный комплект, наименование местного соединения, окно на прием, окно на передачу, состояние соединения, количество буферов, ожидающих подтверждения, количество буферов прерывания приема, состояние срочности, предпочтение, безопасность, таймаут передачи.

Ниже представлен ряд стрелочных диаграмм, представляющих основные особенности установления соединений протокола TCP/IP.

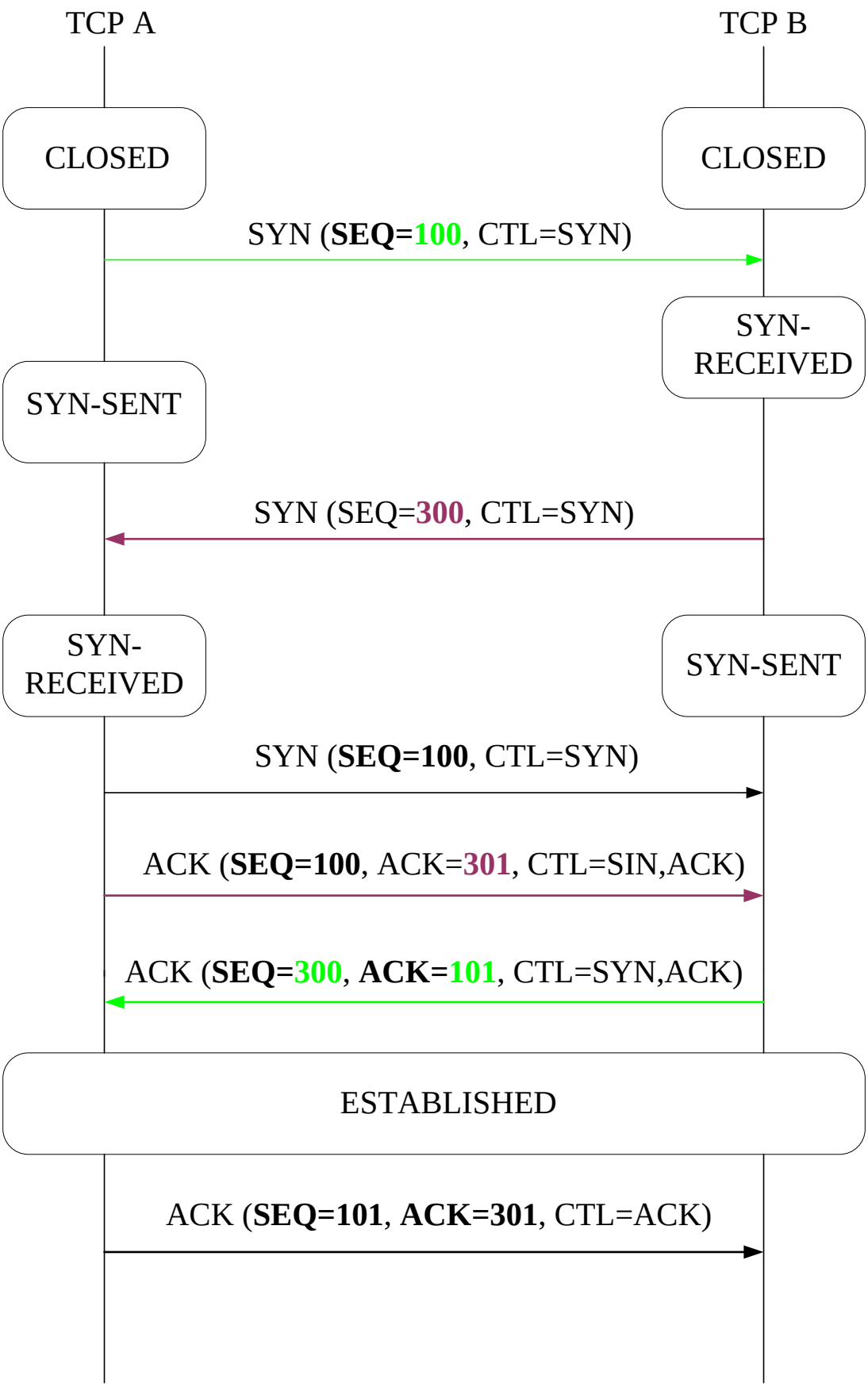


Стрелочная диаграмма трехсторонней квитации

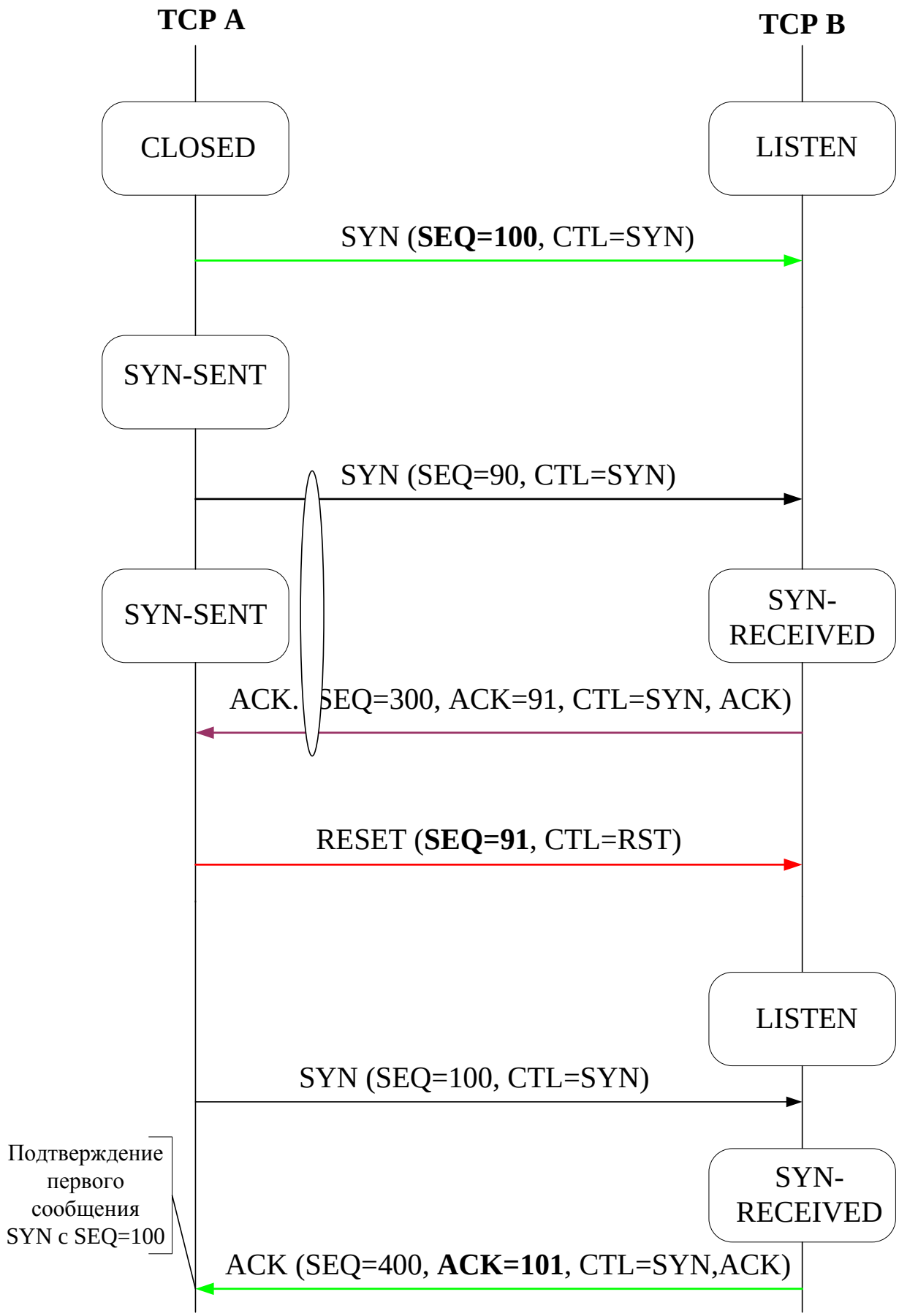
Установление TCP соединения

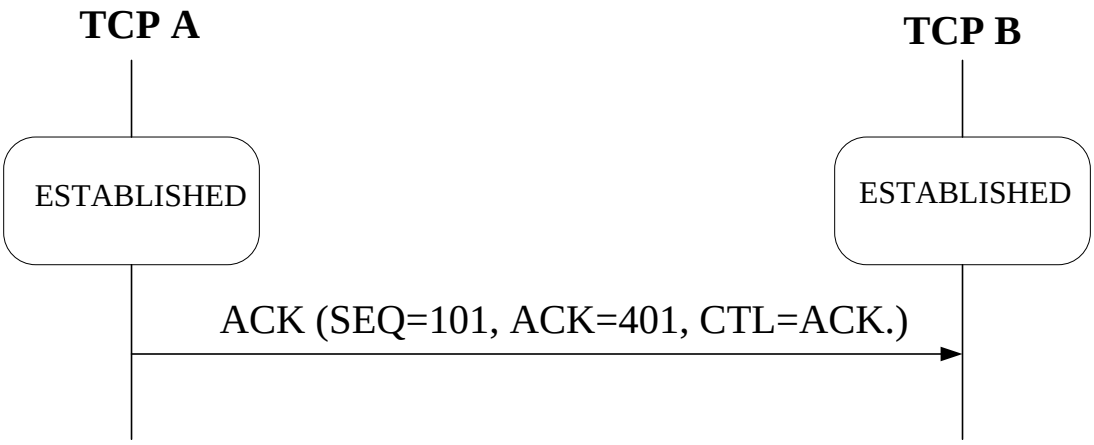


Трехсторонняя процедура рукопожатия для синхронизации соединения



Одновременная синхронизация соединения


TCP A
TCP B
CLOSED
LISTEN
SYN (**SEQ=100**, CTL=SYN)
SYN-SENT
SYN (SEQ=90, CTL=SYN)
SYN-SENT
SYN-RECEIVED
ACK. SEQ=300, ACK=91, CTL=SYN, ACK)
RESET (**SEQ=91**, CTL=RST)
LISTEN
SYN (SEQ=100, CTL=SYN)
Подтверждение первого сообщения SYN с SEQ=100
SYN-RECEIVED
ACK (SEQ=400, **ACK=101**, CTL=SYN,ACK)



Дублирование сообщений TCP

Последнее дублирующее сообщение (SEQ=90) сбрасывается посредством передачи от TCP A сообщения RESET (SEQ=91), а переданное ранее сообщение SYN (SEQ=100) передается на обработку в TCP-B, подтверждающая его сообщением ACK (..ACK=101), после чего TCP-A и TCP-B переходят в состояние «ESTABLISHED».

При одновременном инициировании соединений, местная и удаленная подсистема TCP реализует циклы «CLOSED – SYN-SENT – SYN-RECEIVED-ESTABLISHED»,что представлено выше на диаграмме «Одновременное инициирование соединений».

Передача данных

Передатчик данных отслеживает следующий используемый порядковый номер в переменной *SND.NXT (следующий последовательный номер на передачу).* Приемник данных отслеживает следующий ожидаемый порядковый номер в переменной *RCV.NXT (следующий последовательный номер, ожидаемый во входящих сегментах, являющийся левой и нижней границей окна на прием).*

Передатчик данных отслеживает последний неподтвержденный порядковый номер в переменной *SND.UNA (старейший неподтвержденный последовательный номер).* Если поток данных моментально сводится к нулю и все переданные данные были подтверждены, то все приведенные выше переменные будут равны.

Когда передатчик создает сегмент и передает его, то переменная *SND.NXT* увеличивает свое значение. Когда приемник принимает сегмент, то он увеличивает значение RCV.NXT и передает подтверждение. Когда передатчик данных принимает подтверждение, он увеличивает значение SND.UNA. Расширение, в рамках которого эти переменные различаются, является мерой задержки в связи. Общее количество шагов приращения переменных зависит от длины поля «Данные» сегмента. При переходе процесса установления сеанса связи TCP в состояние *«Established»,* все сегменты должны переносить текущую информацию подтверждения.

Управление окном.

Поле «окно» передается в каждом сегменте. Оно указывает на диапазон последовательных номеров, который передатчик этого поля подготовил для приема. Предполагается наличие свободного для установления соединения пространства буфера данных. При поступлении большего объема данных, по сравнению с указанным в «окне», они отбрасываются. Это является результатом избыточного количества повторных передач, увеличивающих нагрузку TCP и сети. Передающая подсистема TCP должна быть готова к приему от пользователя и передачи, по крайней мере одного октета новых данных даже в случае нулевого размера окна.

Таймаут повторной передачи

Вследствие разнообразия сетей, образующих систему Интернет и широкого диапазона использований TCP соединений, таймаут повторной передачи должен определяться динамически.

Обработка сообщения RESET

Во всех состояниях, исключая «SYN-SENT», все сегменты сообщения RESET проверяются по их полям последовательных номеров (SEQ). Сообщение RESET является достоверным, если его последовательный номер находится в пределах окна. В состоянии «SYN-SENT» (RST

принят в качестве отклика на начальное сообщение SYN), RST принимается, если поле «ACK» подтверждает сообщение SYN.

Обработка команды «ABORT»

Эта команда вызывает прекращение всех задержанных операций «SEND» и «RECEIVE», удаление TCB и выдачу к подсистеме TCP удаленного окончания сообщения RESET. В зависимости от конкретной реализации, пользователи могут принимать индикации «ABORT» для каждой переданной операции SEND и RECEIVE или же могут просто принимать подтверждения «ABORT».

Закрытие соединения

Сообщение «CLOSE» поступившее к местной подсистеме TCP со стороны пользователя, означает «У меня больше нет данных для передачи». Диаграмма переходов состояний процесса закрытия TCP соединения, приведена выше на рисунке *«Диаграмма переходов состояний TCP соединения».*

Пользователь, закрывающий соединение, может продолжать передавать данные, пока он не будет информирован о закрытии соединения на противоположной стороне. Пользователь информируется об операции закрытия соединения встречным TCP, что дает ему возможность корректно завершить сеанс связи.

Имеется три основных случая закрытия канала.

1. Инициированное одним из участников соединения;
2. Удаленная TCP инициирует закрытие соединения посредством передачи сигнала управления FIN.
3. Оба пользователя одновременно закрывают соединение.

Случай 1

Локальный пользователь инициирует закрытие соединения. В этом случае может быть образован сегмент FIN и помещен в очередь исходящего сегмента. От пользователя TCP более не принимаются никакие дальнейшие операции «SEND». TCP переходит в состояние «FIN-WAIT-1». В данном состоянии допускается прием информации («RECEIVE»). Все переданные и включающие FIN сегменты будут повторно передаваться до их подтверждения. Когда противоположная подсистема TCP подтверждает сообщение TCP и передает собственное сообщение FIN, то исходящая TCP-A подтверждает данное сообщение FIN. При этом, подсистема TCP, принимающая сообщение FIN, подтверждает его, но собственное значение FIN передает только после приема от пользователя команды закрытия соединения.

Случай 2

Если со стороны сети поступает сообщение FIN, то принимающая подсистема TCP может его подтвердить и информировать пользователя о закрытии соединения. Пользователь передает отклик “CLOSE”, при наличии

которого, TCP может передать FIN к противоположной TCP взаимодействия после передачи любых оставшихся данных. Далее, TCP ожидает подтверждения переданного сообщения FIN, при приеме которого, сообщение удаляется. Если по истечению таймаута подтверждений от TCP-B не поступило, то соединение прерывается и пользователь об этом информируется.

Случай 3

Оба пользователя одновременно закрывают соединение.

Одновременная передача команды «CLOSE» пользователями на обоих окончаниях соединения, вызывает обмен сегментами FIN. Когда все сегменты, передающие эти сообщения FIN, были обработаны и подтверждены, каждая подсистема TCP может подтвердить принятое сообщение FIN.