**TCP** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Transmission Control Protocol* — протокол управления передачей) — один из основных [протоколов передачи данных](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB_%D0%BF%D0%B5%D1%80%D0%B5%D0%B4%D0%B0%D1%87%D0%B8_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) интернета. Предназначен для управления [передачей данных интернета](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B5%D0%B4%D0%B0%D1%87%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85). Пакеты в TCP называются *сегментами***.**

В стеке протоколов [TCP/IP](https://ru.wikipedia.org/wiki/TCP/IP) выполняет функции [транспортного уровня](https://ru.wikipedia.org/wiki/%D0%A2%D1%80%D0%B0%D0%BD%D1%81%D0%BF%D0%BE%D1%80%D1%82%D0%BD%D1%8B%D0%B9_%D1%83%D1%80%D0%BE%D0%B2%D0%B5%D0%BD%D1%8C) [модели OSI](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_OSI).

Механизм TCP предоставляет [поток данных](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%82%D0%BE%D0%BA_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) с предварительной установкой соединения, осуществляет повторный запрос данных в случае потери данных и устраняет дублирование при получении двух копий одного пакета, гарантируя тем самым (в отличие от [UDP](https://ru.wikipedia.org/wiki/UDP)) целостность передаваемых данных и уведомление отправителя о результатах передачи.

Реализации TCP обычно встроены в [ядра](https://ru.wikipedia.org/wiki/%D0%AF%D0%B4%D1%80%D0%BE_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%BE%D0%B9_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D1%8B) [ОС](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0). Существуют реализации TCP, работающие в [пространстве пользователя](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%81%D1%82%D1%80%D0%B0%D0%BD%D1%81%D1%82%D0%B2%D0%BE_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8F).

Когда осуществляется передача от компьютера к компьютеру через Интернет, TCP работает на верхнем уровне между двумя конечными системами, например, [браузером](https://ru.wikipedia.org/wiki/%D0%91%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80) и веб-сервером. TCP осуществляет надёжную передачу потока байтов от одного процесса к другому.

|  |
| --- |
| **Структура заголовка** |
| [**Бит**](https://ru.wikipedia.org/wiki/%D0%91%D0%B8%D1%82) | **0 — 3** | **4 — 6** | **7 — 15** | **16 — 31** |
| **0** | Порт источника, **Source Port**  | Порт назначения, **Destination Port** |
| **32** | Порядковый номер, **Sequence Number (SN)** |
| **64** | Номер подтверждения, **Acknowledgment Number (ACK SN)** |
| **96** | Длина заголовка, (**Data offset**) | Зарезервировано | Флаги | Размер Окна, **Window size** |
| **128** | Контрольная сумма, **Checksum** | Указатель важности, **Urgent Point** |
| **160** | Опции (необязательное, но используется практически всегда) |
| **160/192+** | Данные |

**Порт источника, Порт назначения**[[править](https://ru.wikipedia.org/w/index.php?title=TCP&veaction=edit&section=2) | [править код](https://ru.wikipedia.org/w/index.php?title=TCP&action=edit&section=2)]

Эти 16-битные поля содержат номера [портов](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%80%D1%82_%28TCP/UDP%29) — числа, которые определяются по [специальному списку](https://ru.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B8%D1%81%D0%BE%D0%BA_%D0%BF%D0%BE%D1%80%D1%82%D0%BE%D0%B2_TCP_%D0%B8_UDP).

**Порт источника** идентифицирует приложение клиента, с которого отправлены пакеты. Ответные данные передаются клиенту на основании этого номера.

**Порт назначения** идентифицирует порт, на который отправлен пакет.

**Порядковый номер**[[править](https://ru.wikipedia.org/w/index.php?title=TCP&veaction=edit&section=3) | [править код](https://ru.wikipedia.org/w/index.php?title=TCP&action=edit&section=3)]

**Sequence number** (32 бита) — измеряется в байтах, и каждый переданный байт полезных данных (payload) увеличивает это значение на 1.

Если установлен флаг SYN (идёт установление сессии), то поле содержит изначальный порядковый номер — ISN (Initial Sequence Number). В целях безопасности это значение генерируется случайным образом и может быть равно от 0 до 232−1 (4294967295). Первый байт полезных данных в устанавливающейся сессии будет иметь номер ISN+1.

В противном случае, если SYN не установлен, первый байт данных, передаваемый в данном пакете, имеет этот порядковый номер.

Поскольку поток TCP в общем случае может быть длиннее, чем число различных состояний этого поля, то все операции с порядковым номером должны выполняться по модулю 232. Это накладывает практическое ограничение на использование TCP. Если скорость передачи коммуникационной системы такова, чтобы в течение MSL (максимального времени жизни сегмента) произошло переполнение порядкового номера, то в сети может появиться два сегмента с одинаковым номером, относящихся к разным частям потока, и приёмник получит некорректные данные.

**Номер подтверждения**[[править](https://ru.wikipedia.org/w/index.php?title=TCP&veaction=edit&section=4) | [править код](https://ru.wikipedia.org/w/index.php?title=TCP&action=edit&section=4)]

**Acknowledgment Number (ACK SN)** (32 бита) — если установлен флаг 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.

Каждая сторона подсчитывает свой Sequence number для переданных данных и отдельно Acknowledgement number для полученных данных. Sequence number каждой из сторон соответствует Acknowledgement number другой стороны.

**Длина заголовка (смещение данных)**[[править](https://ru.wikipedia.org/w/index.php?title=TCP&veaction=edit&section=5) | [править код](https://ru.wikipedia.org/w/index.php?title=TCP&action=edit&section=5)]

Длина заголовка (Data offset) занимает 4 бита и указывает значение длины заголовка, измеренное в 32-битовых [словах](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%BE%D0%B5_%D1%81%D0%BB%D0%BE%D0%B2%D0%BE). Минимальный размер составляет 20 байт (пять 32-битовых [слов](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%BE%D0%B5_%D1%81%D0%BB%D0%BE%D0%B2%D0%BE)), а максимальный — 60 байт (пятнадцать 32-битовых [слов](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%BE%D0%B5_%D1%81%D0%BB%D0%BE%D0%B2%D0%BE)). Длина заголовка определяет смещение полезных данных относительно начала сегмента. Например, Data offset равное 11112 говорит о том, что заголовок занимает пятнадцать 32-битных [слова](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%BE%D0%B5_%D1%81%D0%BB%D0%BE%D0%B2%D0%BE) (15 строк\*32 бита в каждой строке/8 бит = 60 байт).

**Зарезервировано**[[править](https://ru.wikipedia.org/w/index.php?title=TCP&veaction=edit&section=6) | [править код](https://ru.wikipedia.org/w/index.php?title=TCP&action=edit&section=6)]

Зарезервировано (3 бита) для будущего использования и должно устанавливаться в ноль.

**Флаги (управляющие биты)**[[править](https://ru.wikipedia.org/w/index.php?title=TCP&veaction=edit&section=7) | [править код](https://ru.wikipedia.org/w/index.php?title=TCP&action=edit&section=7)]

Это поле содержит 9 битовых флагов:

* **NS** (ECN-nonce) — Устойчивый механизм сигнализации насыщения с помощью ECN-nonce ([RFC 3540](https://tools.ietf.org/html/rfc3540))
* **CWR** (Congestion Window Reduced) — Поле «Окно перегрузки уменьшено» — флаг установлен отправителем, чтобы указать, что получен пакет с установленным флагом ECE ([RFC 3168](https://tools.ietf.org/html/rfc3168))
* **ECE** (ECN-Echo) — Поле «Эхо ECN» — указывает, что данный узел способен на ECN (явное уведомление перегрузки) и для указания отправителю о перегрузках в сети ([RFC 3168](https://tools.ietf.org/html/rfc3168))
* **URG** — поле *«Указатель важности»* задействовано ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Urgent pointer field is significant*). Когда узел отправляет сегмент с URG флагом, то узел-получатель принимает его на отдельном канале.
* **ACK** — поле *«Номер подтверждения»* задействовано ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Acknowledgement field is significant*)
* **PSH** — ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Push function*) инструктирует получателя протолкнуть данные, накопившиеся в приёмном буфере, в приложение пользователя. API для установки PSH флага нет. Обычно он устанавливается ядром, когда оно очищает буфер. Дело в том, что когда узел отправляет информацию, TCP сохраняет её в буфере и не передает её сразу другому узлу, ожидая, захочет ли узел-отправитель передать ещё. Такая же схема работает и у узла-получателя. Когда он получает информацию, TCP сохраняет её в буфере, чтобы не тревожить приложение из-за каждого байта полученной информации. Если узел отправляет сегмент с PSH флагом, это значит, что он отправил все, что было нужно.
* **RST** — оборвать соединения, сбросить буфер (очистка буфера) ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Reset the connection*)
* **SYN** — синхронизация номеров последовательности ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Synchronize sequence numbers*)
* **FIN** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *final*, бит) — флаг, будучи установлен, указывает на завершение соединения ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *FIN bit used for connection termination*).

**Размер окна**[[править](https://ru.wikipedia.org/w/index.php?title=TCP&veaction=edit&section=8) | [править код](https://ru.wikipedia.org/w/index.php?title=TCP&action=edit&section=8)]

Window Size самостоятельно определяет количество байт данных ([payload](https://ru.wikipedia.org/wiki/Payload)), после передачи которых отправитель ожидает подтверждения от получателя, что данные получены. Иначе говоря, получатель пакета располагает для приёма данных буфером длиной «размер окна» байт.

По умолчанию размер окна измеряется в байтах, поэтому ограничен 216 (65535) байтами. Однако благодаря TCP опции Window scale option этот размер может быть увеличен до 1 Гбайта. Чтобы задействовать эту опцию, обе стороны должны согласовать это в своих SYN сегментах.

**Контрольная сумма (Checksum)**[[править](https://ru.wikipedia.org/w/index.php?title=TCP&veaction=edit&section=9) | [править код](https://ru.wikipedia.org/w/index.php?title=TCP&action=edit&section=9)]

Поле контрольной суммы — это 16-битное дополнение к сумме всех 16-битных [слов](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%BE%D0%B5_%D1%81%D0%BB%D0%BE%D0%B2%D0%BE) заголовка (включая псевдозаголовок) и данных. Если сегмент, по которому вычисляется контрольная сумма, имеет длину не кратную 16-битам, то длина сегмента увеличивается до кратной 16-ти, за счёт дополнения к нему справа нулевых битов заполнения. Биты заполнения (0) не передаются в сообщении и служат только для расчёта контрольной суммы. При расчёте контрольной суммы значение самого поля контрольной суммы принимается равным 0.

**Указатель важности (Urgent pointer)**[[править](https://ru.wikipedia.org/w/index.php?title=TCP&veaction=edit&section=10) | [править код](https://ru.wikipedia.org/w/index.php?title=TCP&action=edit&section=10)]

16-битовое значение положительного смещения от порядкового номера в данном сегменте. Это поле указывает порядковый номер октета, которым заканчиваются важные (urgent) данные. Поле принимается во внимание только для пакетов с установленным флагом URG. Используется для [внеполосных данных](https://ru.wikipedia.org/wiki/%D0%92%D0%BD%D0%B5%D0%BF%D0%BE%D0%BB%D0%BE%D1%81%D0%BD%D1%8B%D0%B5_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D0%B5).

**Опции**[[править](https://ru.wikipedia.org/w/index.php?title=TCP&veaction=edit&section=11) | [править код](https://ru.wikipedia.org/w/index.php?title=TCP&action=edit&section=11)]

Могут применяться в некоторых случаях для расширения протокола. Иногда используются для тестирования. На данный момент в опции практически всегда включают 2 байта [NOP](https://ru.wikipedia.org/wiki/NOP) (в данном случае 0x01) и 10 байт, задающих [timestamps](https://ru.wikipedia.org/wiki/Timestamp). Вычислить длину поля опции можно через значение поля смещения.

Механизм действия протокола[[править](https://ru.wikipedia.org/w/index.php?title=TCP&veaction=edit&section=12) | [править код](https://ru.wikipedia.org/w/index.php?title=TCP&action=edit&section=12)]

В отличие от традиционной альтернативы — UDP, который может сразу же начать передачу пакетов, TCP устанавливает соединения, которые должны быть созданы перед передачей данных. TCP-соединение можно разделить на 3 стадии:

* Установка соединения
* Передача данных
* Завершение соединения

**Состояния сеанса TCP**[[править](https://ru.wikipedia.org/w/index.php?title=TCP&veaction=edit&section=13) | [править код](https://ru.wikipedia.org/w/index.php?title=TCP&action=edit&section=13)]

Упрощённая диаграмма состояний TCP. Более подробно в [TCP EFSM diagram](http://www.medianet.kent.edu/techreports/TR2005-07-22-tcp-EFSM.pdf) (на английском языке)

|  |
| --- |
| **Состояния сеанса TCP** |
| **CLOSED** | Начальное состояние узла. Фактически фиктивное |
| **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) |

**Установка соединения**[[править](https://ru.wikipedia.org/w/index.php?title=TCP&veaction=edit&section=14) | [править код](https://ru.wikipedia.org/w/index.php?title=TCP&action=edit&section=14)]

Процесс начала сеанса TCP (также называемый «рукопожатие» ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *handshake*)), состоит из трёх шагов.

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 (буфера 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 клиент 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 тайм-аута он закрывает сокет и переходит в состояние CLOSED.

Процесс называется «трёхэтапным рукопожатием» ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *three way handshake*), так как, несмотря на то, что возможен процесс установления соединения с использованием четырёх сегментов (SYN в сторону сервера, ACK в сторону клиента, SYN в сторону клиента, ACK в сторону сервера), на практике для экономии времени используется три сегмента.

Пример базового 3-этапного согласования:

TCP A TCP B

 1. CLOSED LISTEN

 2. SYN-SENT --> <SEQ=100><CTL=SYN> --> SYN-RECEIVED

 3. ESTABLISHED <-- <SEQ=300><ACK=101><CTL=SYN,ACK> <-- SYN-RECEIVED

 4. ESTABLISHED --> <SEQ=101><ACK=301><CTL=ACK> --> ESTABLISHED

 5. ESTABLISHED <-- <SEQ=301><ACK=101><CTL=ACK> <-- ESTABLISHED

В строке 2 TCP A начинает передачу сегмента SYN, говорящего об использовании номеров последовательности, начиная со 100;

В строке 3 TCP B передаёт SYN и подтверждение для принятого SYN в адрес TCP A. Поле подтверждения показывает ожидание TCP B приёма номера последовательности 101, подтверждающего SYN с номером 100;

В строке 4 TCP A отвечает пустым сегментом с подтверждением ACK для сегмента SYN от TCP B;

В строке 5 TCP B передаёт некоторые данные. Отметим, что номер подтверждения сегмента в строке 5 (ACK=101) совпадает с номером последовательности в строке 4 (SEQ=101), поскольку ACK не занимает пространства номеров последовательности (если это сделать, придётся подтверждать подтверждения — ACK для ACK).

Существуют экспериментальные расширения протокола TCP, сокращающие количество пакетов при установлении соединения, например [TCP Fast Open](https://ru.wikipedia.org/w/index.php?title=TCP_Fast_Open&action=edit&redlink=1)[[en]](https://en.wikipedia.org/wiki/TCP_Fast_Open). Ранее также существовало расширение [T/TCP](https://ru.wikipedia.org/wiki/T/TCP). Для прозрачного шифрования данных предлагается использовать расширение [tcpcrypt](https://ru.wikipedia.org/wiki/Tcpcrypt).

**Передача данных**[[править](https://ru.wikipedia.org/w/index.php?title=TCP&veaction=edit&section=15) | [править код](https://ru.wikipedia.org/w/index.php?title=TCP&action=edit&section=15)]

*См. также:*[*Алгоритм Нейгла*](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%9D%D0%B5%D0%B9%D0%B3%D0%BB%D0%B0)*и*[*Медленный старт*](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D0%B4%D0%BB%D0%B5%D0%BD%D0%BD%D1%8B%D0%B9_%D1%81%D1%82%D0%B0%D1%80%D1%82)

При обмене данными приёмник использует номер последовательности, содержащийся в получаемых сегментах, для восстановления их исходного порядка. Приёмник уведомляет передающую сторону о номере последовательности, до которой он успешно получил данные, включая его в поле «номер подтверждения». Все получаемые данные, относящиеся к промежутку подтверждённых последовательностей, игнорируются. Если полученный сегмент содержит номер последовательности больший, чем ожидаемый, то данные из сегмента буферизируются, но номер подтверждённой последовательности не изменяется. Если впоследствии будет принят сегмент, относящийся к ожидаемому номеру последовательности, то порядок данных будет автоматически восстановлен исходя из номеров последовательностей в сегментах.

Для того, чтобы передающая сторона не отправляла данные интенсивнее, чем их может обработать приёмник, TCP содержит средства управления потоком. Для этого используется поле «окно». В сегментах, направляемых от приёмника передающей стороне, в поле «окно» указывается текущий размер приёмного буфера. Передающая сторона сохраняет размер окна и отправляет данных не более, чем указал приёмник. Если приёмник указал нулевой размер окна, то передачи данных в направлении этого узла не происходит, пока приёмник не сообщит о большем размере окна.

В некоторых случаях передающее приложение может явно затребовать передать данные до некоторой последовательности принимающему приложению, не буферизируя их. Для этого используется флаг PSH. Если в полученном сегменте обнаруживается флаг PSH, то реализация TCP отдаёт все буферизированные на текущий момент данные принимающему приложению. «Проталкивание» используется, например, в интерактивных приложениях. В сетевых терминалах нет смысла ожидать ввода пользователя после того, как он закончил набирать команду. Поэтому последний сегмент, содержащий команду, обязан содержать флаг PSH, чтобы приложение на принимающей стороне смогло начать её выполнение.

**Завершение соединения**[[править](https://ru.wikipedia.org/w/index.php?title=TCP&veaction=edit&section=16) | [править код](https://ru.wikipedia.org/w/index.php?title=TCP&action=edit&section=16)]

Завершение соединения можно рассмотреть в три этапа:

1. Посылка серверу от клиента флага FIN на завершение соединения.
2. Сервер посылает клиенту флаги ответа ACK , FIN, что соединение закрыто.
3. После получения этих флагов клиент закрывает соединение и в подтверждение отправляет серверу ACK , что соединение закрыто.

Известные проблемы[[править](https://ru.wikipedia.org/w/index.php?title=TCP&veaction=edit&section=17) | [править код](https://ru.wikipedia.org/w/index.php?title=TCP&action=edit&section=17)]

**Максимальный размер сегмента**[[править](https://ru.wikipedia.org/w/index.php?title=TCP&veaction=edit&section=18) | [править код](https://ru.wikipedia.org/w/index.php?title=TCP&action=edit&section=18)]

TCP требует явного указания максимального размера сегмента ([MSS](https://ru.wikipedia.org/wiki/MSS)) в случае, если виртуальное соединение осуществляется через сегмент сети, где максимальный размер блока ([MTU](https://ru.wikipedia.org/wiki/Maximum_transmission_unit)) менее, чем стандартный MTU Ethernet (1500 байт).

В протоколах туннелирования, таких как [GRE](https://ru.wikipedia.org/wiki/GRE_%28%D0%BF%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB%29), [IPIP](https://ru.wikipedia.org/w/index.php?title=IPIP&action=edit&redlink=1), а также [PPPoE](https://ru.wikipedia.org/wiki/PPPoE) MTU туннель меньше, чем стандартный, поэтому сегмент TCP максимального размера имеет длину пакета больше, чем MTU. Это приводит к фрагментации и уменьшению скорости передачи полезных данных. Если на каком-либо узле фрагментация запрещена, то со стороны пользователя это выглядит как «зависание» соединений. При этом «зависание» может происходить в произвольные моменты времени, а именно тогда, когда отправитель использовал сегменты длиннее допустимого размера. Для решения этой проблемы на маршрутизаторах применяются правила Firewall-а, добавляющие параметр MSS во все пакеты, инициирующие соединения, чтобы отправитель использовал сегменты допустимого размера.

MSS может также управляться параметрами операционной системы.

**Обнаружение ошибок при передаче данных**[[править](https://ru.wikipedia.org/w/index.php?title=TCP&veaction=edit&section=19) | [править код](https://ru.wikipedia.org/w/index.php?title=TCP&action=edit&section=19)]

Хотя протокол осуществляет проверку контрольной суммы по каждому сегменту, используемый алгоритм считается слабым [[1]](http://portal.acm.org/citation.cfm?doid=347059.347561).

В общем случае распределенным сетевым приложениям рекомендуется использовать дополнительные программные средства для гарантирования целостности передаваемой информации[[2]](http://evanjones.ca/tcp-checksums.html).

**Атаки на протокол**[[править](https://ru.wikipedia.org/w/index.php?title=TCP&veaction=edit&section=20) | [править код](https://ru.wikipedia.org/w/index.php?title=TCP&action=edit&section=20)]

*См. также:*[*TCP hijacking*](https://ru.wikipedia.org/wiki/TCP_hijacking)*и*[*Атака TCP Reset*](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B0%D0%BA%D0%B0_TCP_Reset)

Недостатки протокола проявляются в успешных теоретических и практических атаках, при которых злоумышленник может получить доступ к передаваемым данным, выдать себя за другую сторону или привести систему в нерабочее состояние.

Реализация[[править](https://ru.wikipedia.org/w/index.php?title=TCP&veaction=edit&section=21) | [править код](https://ru.wikipedia.org/w/index.php?title=TCP&action=edit&section=21)]

**Псевдозаголовок**[[править](https://ru.wikipedia.org/w/index.php?title=TCP&veaction=edit&section=22) | [править код](https://ru.wikipedia.org/w/index.php?title=TCP&action=edit&section=22)]

TCP-заголовок не содержит информации об адресе отправителя и получателя, поэтому даже при совпадении порта получателя нельзя с точностью сказать, что сообщение пришло в нужное место. Поскольку назначением протокола TCP является надёжная доставка сообщений, то этот момент имеет принципиальное значение. Эту задачу можно было решить разными способами. Самый очевидный — добавить информацию об адресе назначения в заголовок TCP, однако это, во-первых, приводит к дублированию информации, что снижает долю полезной информации, переносимой TCP-сегментом, а во-вторых, нарушает принцип инкапсуляции модели OSI. Поэтому разработчики протокола пошли другим путём и использовали дополнительный псевдозаголовок:

*TCP-псевдозаголовок IPv4*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Биты** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** | **21** | **22** | **23** | **24** | **25** | **26** | **27** | **28** | **29** | **30** | **31** |
| **0-31** | IP-адрес отправителя (Source address) |
| **32-63** | IP-адрес получателя (Destination address) |
| **64-95** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Протокол (Protocol) | Длина TCP-сегмента (TCP length) |

*TCP-псевдозаголовок IPv6*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Биты** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** | **21** | **22** | **23** | **24** | **25** | **26** | **27** | **28** | **29** | **30** | **31** |
| **0-95** | IP-адрес отправителя (Source address) |
| **128-223** | IP-адрес получателя (Destination address) |
| **224-255** | Длина TCP-сегмента (TCP length) |
| **256-287** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Протокол верхнего уровня (Next header) |

* Протокол (Protocol)/Протокол верхнего уровня (Next header) — содержит в себе значение 6 (00000110 в двоичном виде, 0x6 — в шестнадцатеричном) — идентификатор TCP-протокола.
* Длина TCP-сегмента (TCP length) — содержит в себе длину TCP-сегмента в байтах (TCP-заголовок + данные; длина псевдозаголовка не учитывается).

Псевдозаголовок не включается в TCP-сегмент. Он используется для расчёта контрольной суммы перед отправлением сообщения и при его получении (получатель составляет свой псевдозаголовок, используя адрес хоста, с которого пришло сообщение, и собственный адрес, а затем считает контрольную сумму).

**Освобождение от расчёта контрольной суммы**[[править](https://ru.wikipedia.org/w/index.php?title=TCP&veaction=edit&section=23) | [править код](https://ru.wikipedia.org/w/index.php?title=TCP&action=edit&section=23)]

Многие реализации стека TCP/IP предоставляют возможности использования аппаратной поддержки для автоматического расчёта контрольной суммы в сетевом адаптере до передачи в сеть или после приёма из сети для верификации. Это может освобождать операционную систему от использования ценных тактов процессора при вычислении контрольной суммы.

Эта функция может приводить к тому, что [анализаторы трафика](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7%D0%B0%D1%82%D0%BE%D1%80_%D1%82%D1%80%D0%B0%D1%84%D0%B8%D0%BA%D0%B0), перехватывающие исходящие пакеты до их передачи в сетевой адаптер и не знающие о делегировании расчёта контрольной суммы сетевому адаптеру, могут сообщать об ошибке контрольной суммы в исходящих пакетах.