

КПИ им. Игоря Сикорского, каф. микроэлектроники.

М. Р. Домбругов. Информатика-1.

Персональные компьютеры и основы сетевых технологий



Лекция 6.

4-й (транспортный) уровень модели OSI

2 ноября 2020

7-уровневая модель OSI

Тип данных	Уровень (layer)	Функции
Данные	7. Прикладной (application)	Доступ к сетевым службам
	6. Представительский (presentation)	Представление и шифрование данных
	5. Сеансовый (session)	Управление сеансом связи
Сегменты	4. Транспортный (transport)	Прямая связь между конечными пунктами и надежность
Пакеты	3. Сетевой (network)	Определение маршрута и логическая адресация
Кадры	2. Канальный (data link)	Физическая адресация
Биты	1. Физический (physical)	Работа со средой передачи, сигналами и двоичными данными

4-й уровень модели OSI



Прямая связь между конечными пунктами и надежность

- Порты и сокеты
- стек протоколов TCP/IP
- NAT
- Протокол UDP
- Протокол TCP

Порты и сокет

Идентификация процессов. Программные порты

Обычно на компьютере выполняется несколько программных процессов (напр., различные окна браузера, прием электронной почты, skype...). Компьютер периодически получает по сети IP-пакет, предназначенный для одного из процессов.

Для идентификации разных процессов на одном компьютере используются «**программные порты**» (не путать с физическими разъемами на корпусе компьютера!): каждое сетевое приложение регистрирует за собой какой-то **номер порта** (2-байтный идентификатор).

Порты и сокеты

Номер порта

Числовой идентификатор от 0 до $2^{16} - 1 = 65535$ (64k номеров).

Все порты разделены на три диапазона:

1 и 2: Общеизвестные (well-known)

и зарегистрированные (registered):

0 ... 49151 (48k номеров, $\frac{3}{4}$ от общего количества).

Эти номера для конкретных целей регистрирует IANA (Internet Assigned Numbers Authority).

Первые 1k номеров (0 ... 1023) наз. общеизвестными, поскольку их номера закрепились исторически для самых популярных протоколов и приложений.

Напр., любой веб-сервер использует 80 порт.

Порты и сокет

3: Динамические или частные (dynamic or private):

49152 ... 65535 (16k номеров, $\frac{1}{4}$ часть от общего количества).

Эти порты используются временными (короткоживущими) соединениями «клиент – сервер».

Напр., при обращении к веб-серверу и запросе от него веб-страницы браузер выступает в качестве клиента.

ОС узла, где исполняется браузер, назначает ему (точнее, его окну) на время соединения с сервером какой-то порт из этого диапазона номеров.

Браузер обращается к 80 («общеизвестному») порту сервера. В ответ сервер со своего 80 порта присылает данные браузеру на тот порт, который он указал в своем запросе. После получения браузером требуемых данных сессия заканчивается. Когда окно браузера закрывается, динамический номер его порта высвобождается.

Стек протоколов TCP/IP

3-й (сетевой) уровень нужен для того чтобы информация достигла узла, а 4-й (транспортный) – чтобы информация попала нужному процессу в пределах этого узла.

Протоколы 4-го (транспортного) уровня, которые работают совместно с протоколом IP, образуют согласованный набор («стек») протоколов **TCP/IP**.

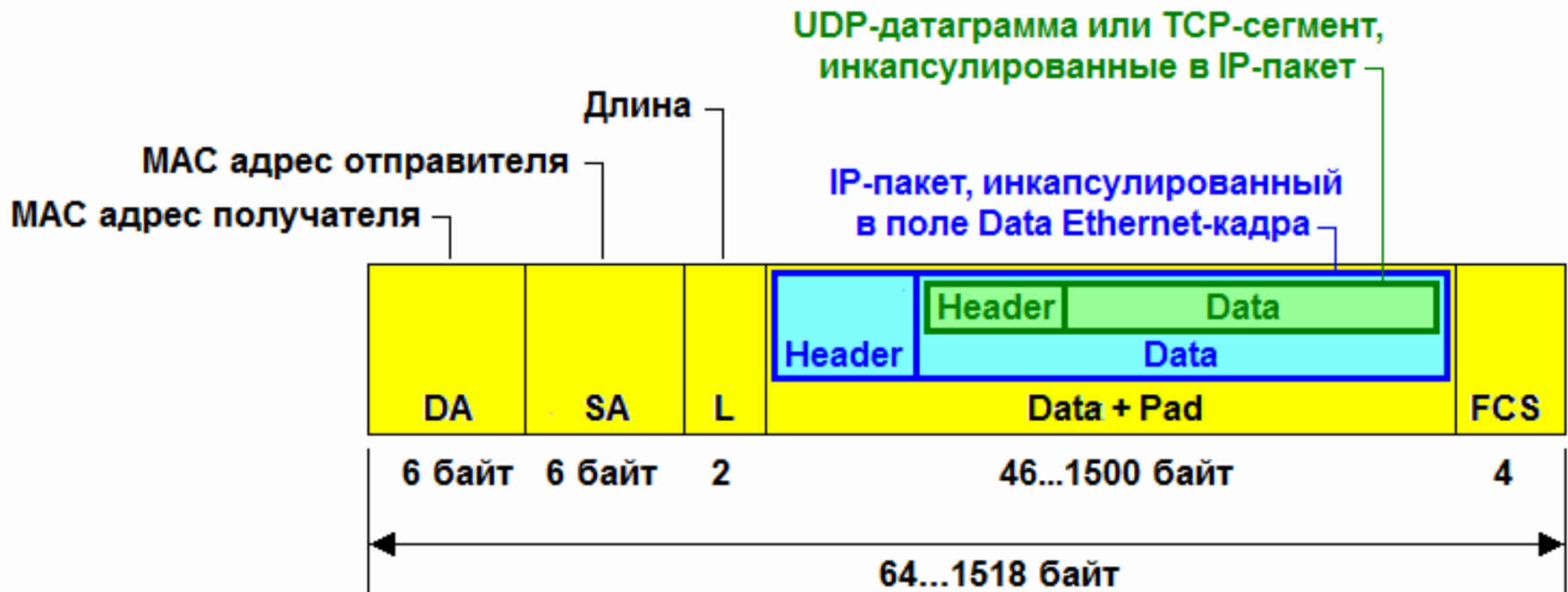
Два основных протокола из стека протоколов TCP/IP и их PDU (Protocol Data Unit, передаваемая порция данных):

- **UDP (User datagram protocol)**,
PDU – **UDP-датаграмма**;
- **TCP (Transmission Control Protocol)**,
PDU – **TCP-сегмент**.

Стек протоколов ТСП/ІР

Инкапсуляция в ТСП/ІР

UDP-датаграмма или ТСП-сегмент инкапсулируются в ІР-пакет, занимая в нем поле данных.

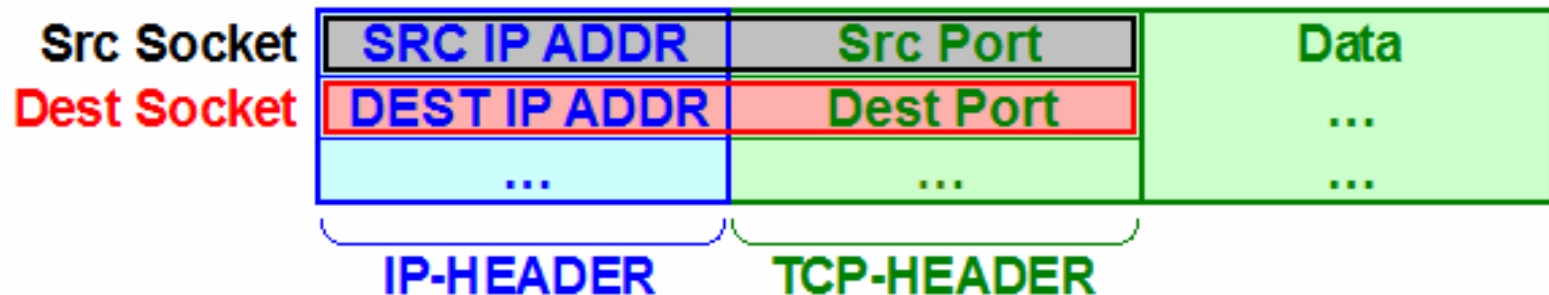


Стек протоколов TCP/IP

Сокет

Пара **IP-адрес + номер порта = сокет (socket, разъем)**.

В процессе обмена используется два сокета – сокет отправителя и сокет получателя.



На письме части сокета разделяются двоеточием, напр.:

212.111.212.227:80

Стек протоколов TCP/IP

- https://ru.wikipedia.org/wiki/Транспортный_уровень
- [https://ru.wikipedia.org/wiki/Порт_\(компьютерные_сети\)](https://ru.wikipedia.org/wiki/Порт_(компьютерные_сети))
- https://ru.wikipedia.org/wiki/Список_портов_TCP_и_UDP
- [https://ru.wikipedia.org/wiki/Сокет_\(программный_интерфейс\)](https://ru.wikipedia.org/wiki/Сокет_(программный_интерфейс))
- <http://ciscotips.ru/transport-layer>
- <http://iptcp.net/porty-i-sokety.html>
- <http://datanets.ru/tipy-nomerov-portov-horosho-izvestnye-zaregistrirovannye-dinamicheskie-chastnye.html>

NAT



Из-за дефицита IP-адресов (~4 млрд) организации используют частные IP адреса **10.0.0.0/8**, **172.16.0.0/12** и **192.168.0.0/16**. Они не уникальны и не маршрутизируются в интернете.

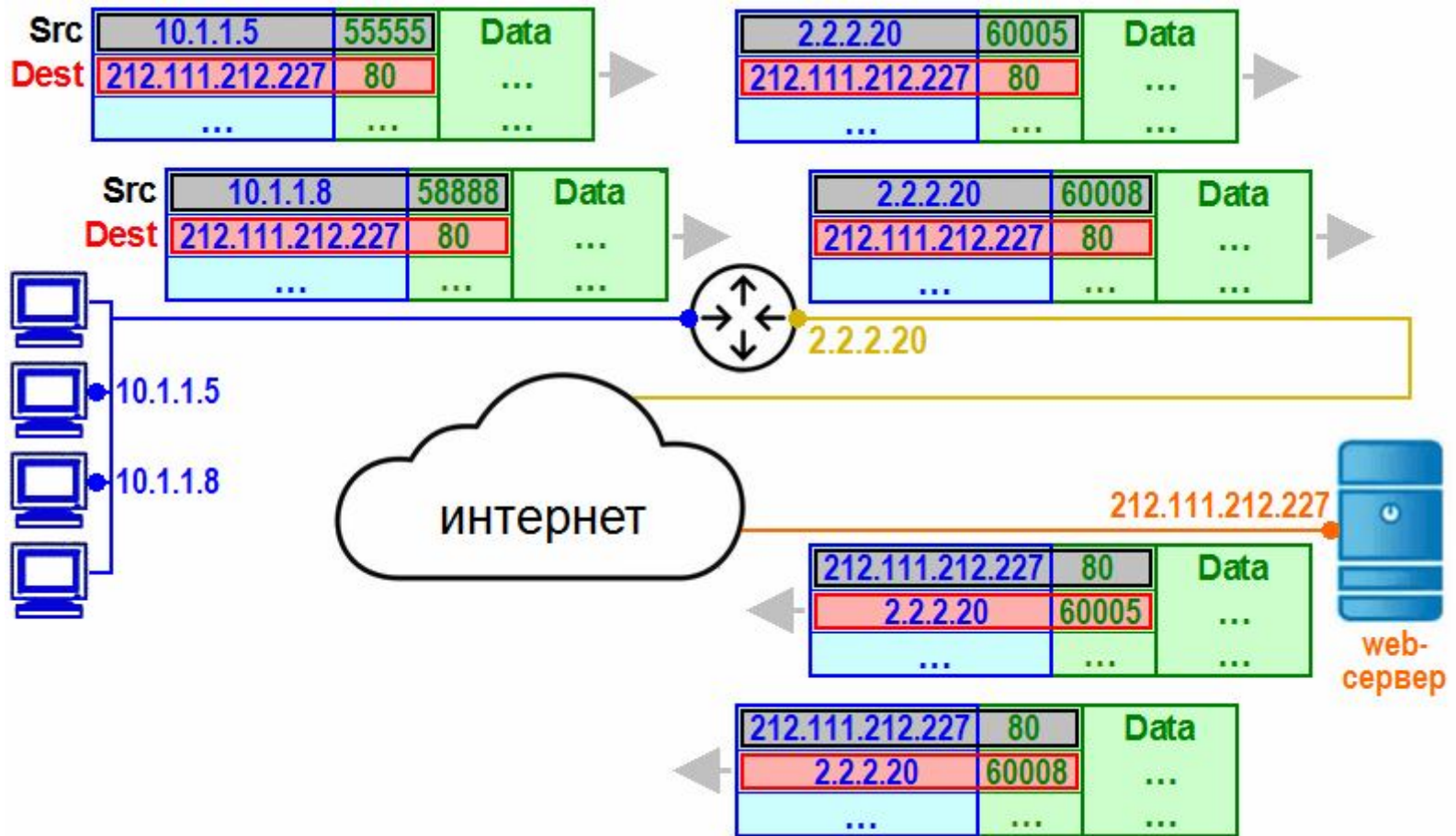
Поэтому IP-пакет из частной сети на уникальный адрес в интернет отправить можно, а из интернета в частную сеть – нельзя.

Для частичного преодоления этой проблемы применяют технику **NAT (Network Address Translation)** или иначе **IP Masquerading**.

Маршрутизатор «на лету» подменяет обратный сокет: прописывает свой внешний (видимый из интернета) IP-адрес в заголовок IP-пакета, а в заголовок TCP-сегмента ставит номер порта из своего динамического набора в 16k номеров.

Тогда в приходящих в ответ IP-пакетах по номеру порта в TCP-сегменте можно сделать обратную подстановку и тем самым различать ответные пакеты для разных локальных компьютеров.

NAT



NAT



Комбинацию, нужную для обратной подстановки, маршрутизатор сохраняет у себя во временной таблице:

Динамический порт в соquete с внешним IP-адресом 2.2.2.20 NAT-маршрутизатора	Сокет с IP-адресом из локальной сети и динамическим портом локальной машины
60005	10.1.1.5 : 55555
60008	10.1.1.8 : 58888

Через некоторое время после того, как клиент и сервер закончат обмениваться пакетами, роутер сотрет у себя в таблице запись о динамическом порте за сроком давности.

Если хостов за NATом несколько, то общее число возможных сокетов у всех отправителей > числа портов маршрутизатора, поэтому количество локальных узлов за NATом ограничено.

NAT

Пример: частный IPv4-адрес в домашней сети

```
C:\Windows\system32\cmd.exe
C:\>ipconfig /all

Адаптер беспроводной локальной сети Беспроводное сетевое соединение:
    DNS-суффикс подключения . . . . . :
    Описание. . . . . : Адаптер беспроводных сетей Atheros AR5007
EG Wireless
    Физический адрес. . . . . : 00-21-63-56-51-7C
    DHCP включен. . . . . : Да
    Автонастройка включена. . . . . : Да
    Локальный IPv6-адрес канала . . . : fe80::d5f7:8eee:194f:dbe6%12(Основной)
    IPv4-адрес. . . . . : 192.168.1.2(Основной)
    Маска подсети . . . . . : 255.255.255.0
    Аренда получена. . . . . : 10 октября 2020 г. 16:30:53
    Срок аренды истекает. . . . . : 12 октября 2020 г. 09:47:05
    Основной шлюз. . . . . : 192.168.1.1
    DHCP-сервер. . . . . : 192.168.1.1
    IAID DHCPv6 . . . . . : 218112355
    DUID клиента DHCPv6 . . . . . : 00-01-00-01-26-E2-6F-F1-00-13-77-93-65-27

    DNS-серверы. . . . . : 192.168.1.1
    NetBios через TCP/IP. . . . . : Включен
```

NAT



NAT не позволяет иметь доступ извне к внутренним узлам: все соединения иницируются изнутри, устройством из-за NAT. Так как отображение адресов задается **исходящими** пакетами, то пока они не отправлены, **входящие** пакеты не принимаются.

При использовании NAT нарушается одно из фундаментальных правил построения многоуровневых протоколов: уровень k не должен предполагать ничего относительно того, что именно уровень $k+1$ поместил в поле полезных данных.

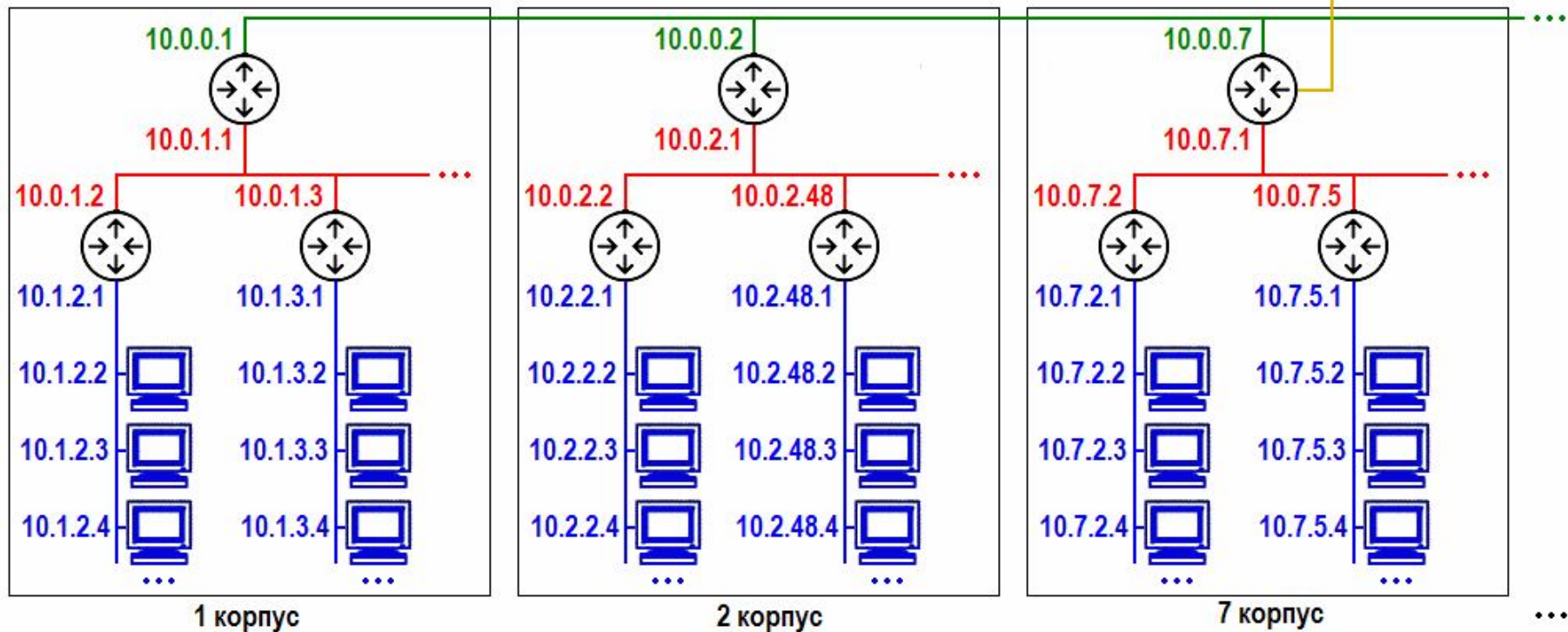
NAT

План IP-адресов в сети КПИ

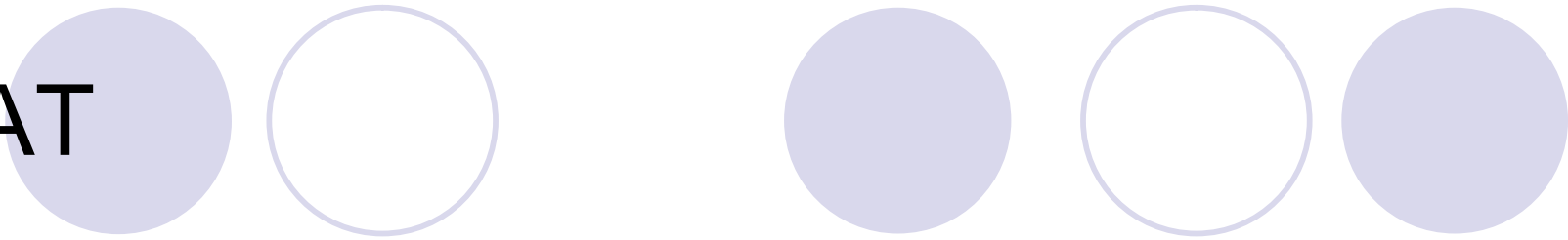
10.0.0.корпус

10.0.корпус.подразделение

10.корпус.подразделение.компьютер



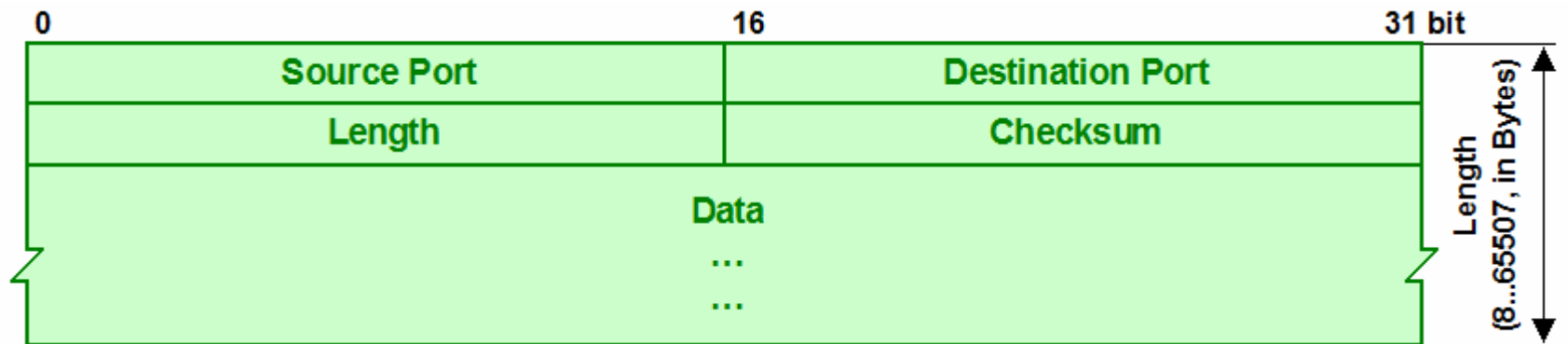
NAT



- <https://ru.wikipedia.org/wiki/NAT>
- <https://wiki.merionet.ru/seti/13/nat-na-palcax-cto-eto/>
- <https://moxa.pro/articles/articles/cto-takoe-nat-osobennosti-v-moxa/>
- <https://neerc.ifmo.ru/wiki/index.php?title=NAT>
- <https://sonikelf.ru/vse-cto-vy-xoteli-znat-o-nat-no-boyalis-sprosit-nat-pat-snat-dnat/>
- <https://linkmeup.gitbook.io/sdsm/5.-acl-i-nat/01-nat>

UDP

Структура UDP-датаграммы



- Source Port – порт отправителя;
- Destination Port – порт получателя;
- Length – длина датаграммы в байтах;
- Checksum – контрольная сумма датаграммы;
- Data – пересылаемые данные.

UDP



Функционирование UDP похоже на IP, за исключением введения портов.

Протокол UDP

- не обеспечивает подтверждение доставки датаграмм,
- не сохраняет их порядок следования,
- может их терять или дублировать.

Назначение UDP – максимально быстрая доставка.

Чувствительные ко времени приложения (напр. голосовая связь) часто используют UDP, так как предпочтительнее сбросить уже полученные данные, чем ждать задержавшиеся: если UDP-датаграмма придёт позже, чем она нужна для вставки в речь, она уже не нужна.

UDP не подходит для передачи бинарных файлов, где потеря одного фрагмента может привести к искажению всего файла.

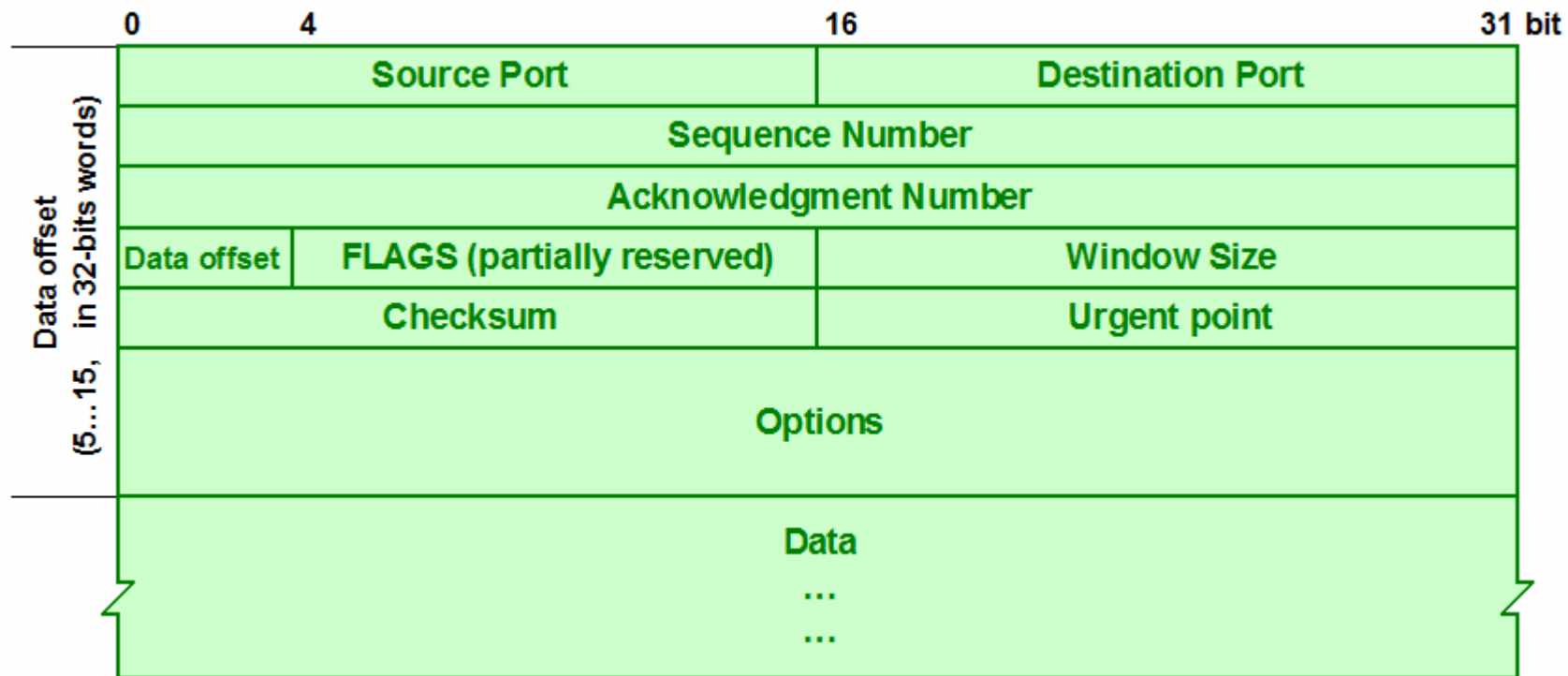
UDP



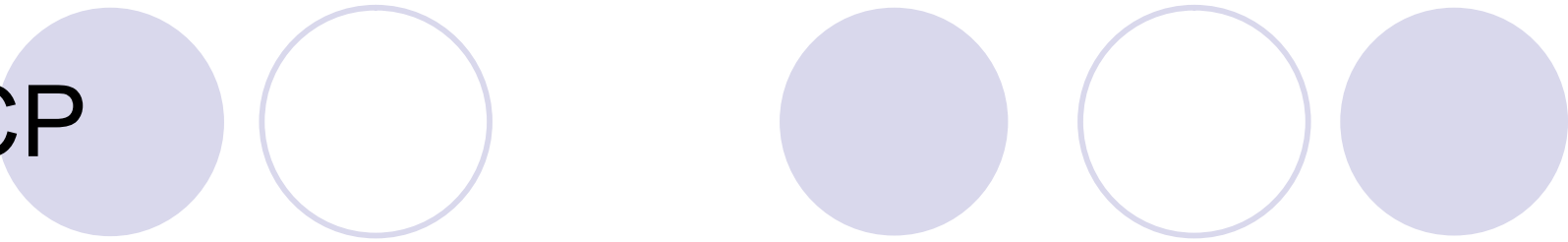
- <https://ru.wikipedia.org/wiki/UDP>
- <http://iptcp.net/protokol-udp-i-udp-deitagrammy.html>
- http://book.itep.ru/4/44/udp_442.htm
- <http://just-networks.ru/seti-tcp-ip/protokol-udp>
- https://docstore.mik.ua/tcp_ip/gl11.htm

TCP

Структура TCP-сегмента



TCP



Структура TCP-сегмента

Поля

- Source Port;
- Destination Port;
- Checksum;
- Data

имеют тот же смысл, что и в UDP-датаграмме.

Поля

- Sequence Number – порядковый номер;
- Acknowledgment Number – номер подтверждения;
- FLAGS – флаги (управляющие биты),
в частности, три флага: ACK, SYN, FIN;
- Window Size – размер окна

обеспечивают дополнительный функционал протокола TCP.

TCP



Основные задачи протокола TCP

Так же, как UDP:

- **идентификация приложений**,
передающих и принимающих данные;

Дополнительно:

- **сегментация данных** –
разбиение данных на сегменты для отсылки по сети,
сборка этих сегментов после получения;
- **организация надежного двустороннего соединения**
между узлами, при котором гарантируется доставка
пакетов данных, причём в правильной
последовательности.
- **контроль за скоростью передачи**
позволяет корректировать скорость отправки данных
в зависимости от возможностей получателя.

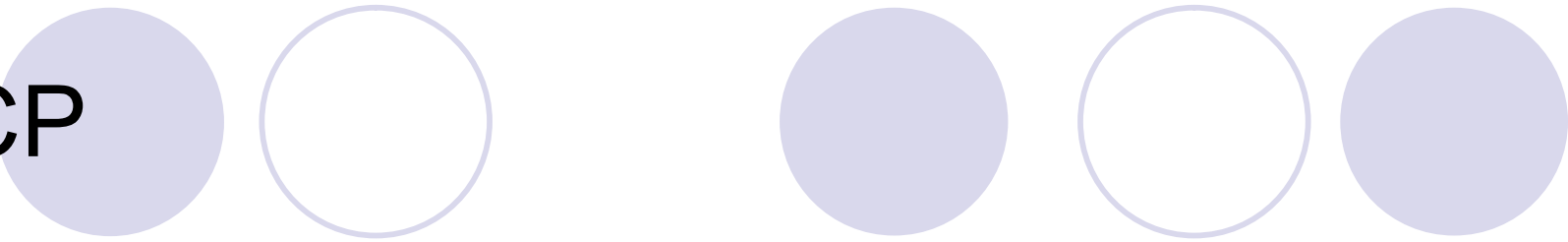
Двустороннее соединение

- Sequence Number (**SN**) – порядковый номер (в байтах). Каждый переданный байт данных увеличивает его на 1.
Изначальный порядковый номер – ISN (Initial Sequence Number) генерируется случайно от 0 до $2^{32}-1$ (~ 4 млрд). Первый байт полезных данных в устанавливаемой сессии будет иметь номер ISN+1;
- Acknowledgment Number (**AN**) – номер подтверждения, порядковый номер байта, который отправитель желает получить. Это означает, что все предыдущие байты (с номерами от ISN+1 до AN-1 включительно) были успешно получены.

Каждая сторона подсчитывает свой SN для переданных данных и AN для полученных данных.

SN каждой из сторон соответствует AN другой стороны.

TCP



Три стадии TCP-соединения

В отличие от UDP, который может сразу же начать передачу данных, в TCP-соединении можно выделить 3 стадии:

- установка соединения;
- передача данных;
- завершение соединения.

Флаги для управления соединением

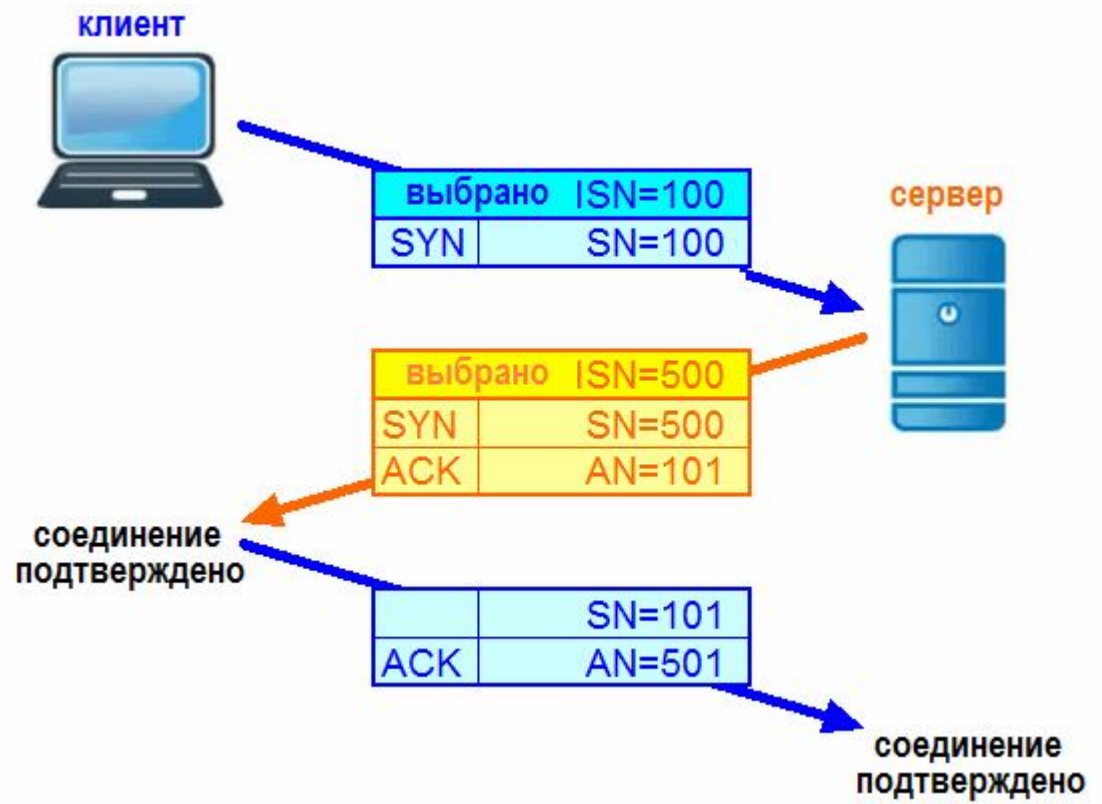
- **SYN** (Synchronize sequence numbers) – синхронизация SN и AN, выставляется при установлении сессии;
- **ACK** (Acknowledgement field is significant) – поле AN задействовано;
- **FIN** (Final) – выставляется чтобы завершить соединение.

TCP

Установка TCP-соединения

«трёхэтапное
рукопожатие»,
three way handshake

В заголовках
всех сегментов,
передаваемых
после установления
соединения, поле
AN заполняется,
а флаг ACK=1.



TSP



Сегментация данных

При отправке сегментов узлы последовательно нумеруют их и для каждого рассчитывают контрольную сумму.

Получатель по нумерации сегментов может установить, что какие-то из них отсутствуют, а при несовпадении контрольной суммы – что сегмент был повреждён при передаче. В этом случае отправляется запрос на повторную отправку сегмента.

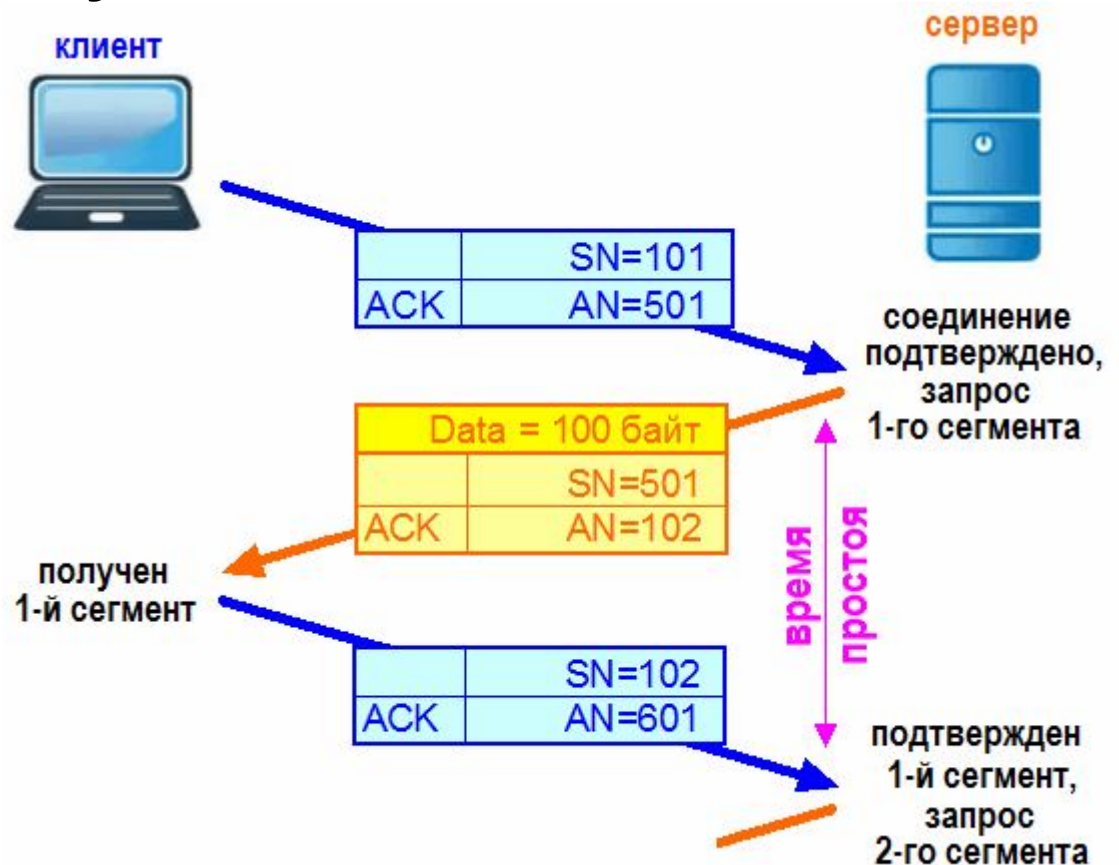
Для совместимости с протоколами 2-го уровня (Ethernet, телефонные линии, радиорелейные, Wi-Fi, ...) максимальный размер поля данных TSP-сегмента обычно ограничивают величиной от 536 до 1460 байт.

TCP

Подтверждение получения сегмента

Отправитель начнет передачу следующей порции данных только тогда, когда получит подтверждение, что данные дошли до получателя.

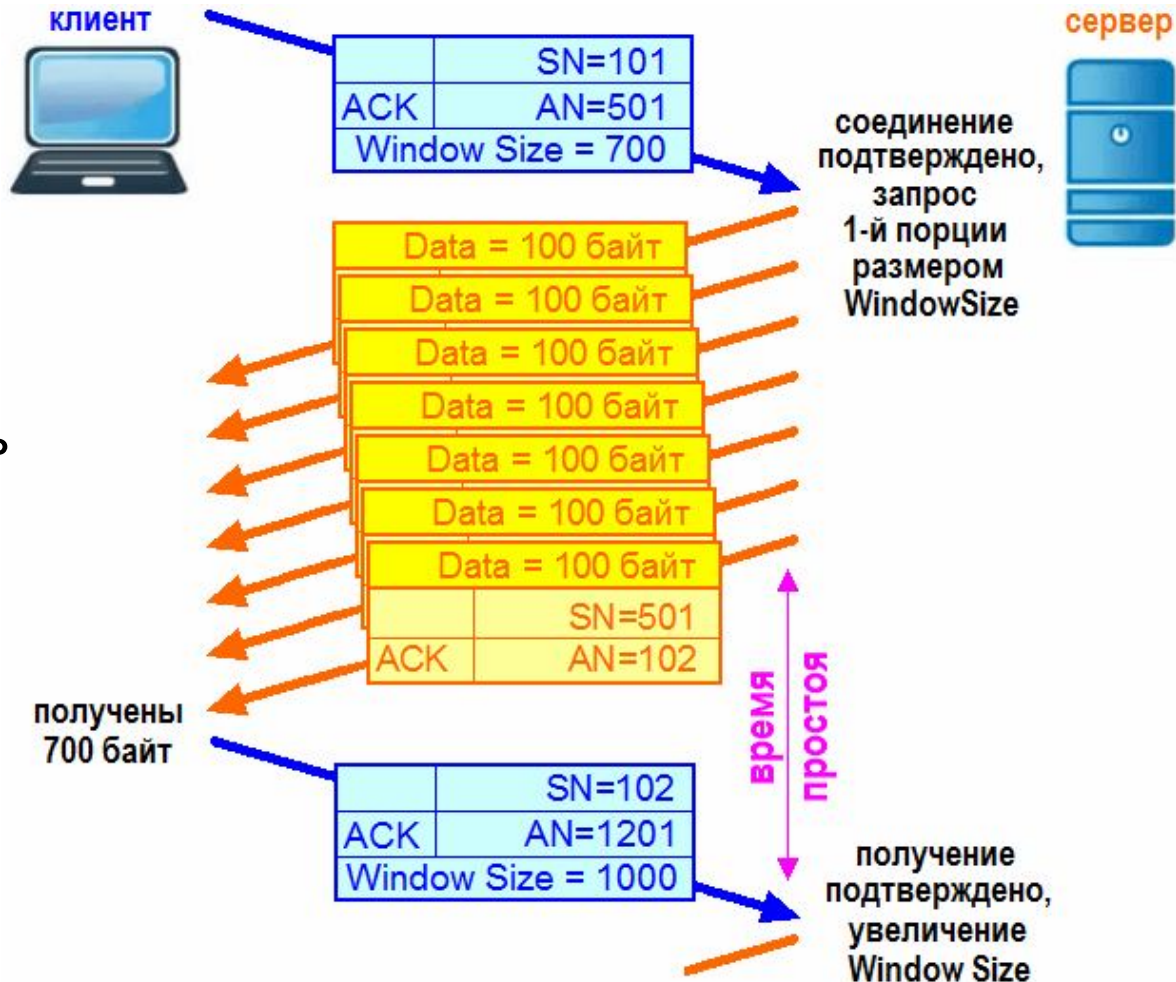
В результате в сети образуется простой длительностью ок. RTT (round-trip time).



TCP

○ **Window Size** –
размер окна

Поле определяет количество байтов данных, которые отправитель может отправить, прежде чем получит подтверждение от получателя.



TCP



В ответ на запрос на передачу отправитель отправляет все байты с номерами от SN до $SN + WindowSize - 1$.

WindowSize может быть длиной в несколько TCP-сегментов. Простой на линии будет после каждых WindowSize байт.

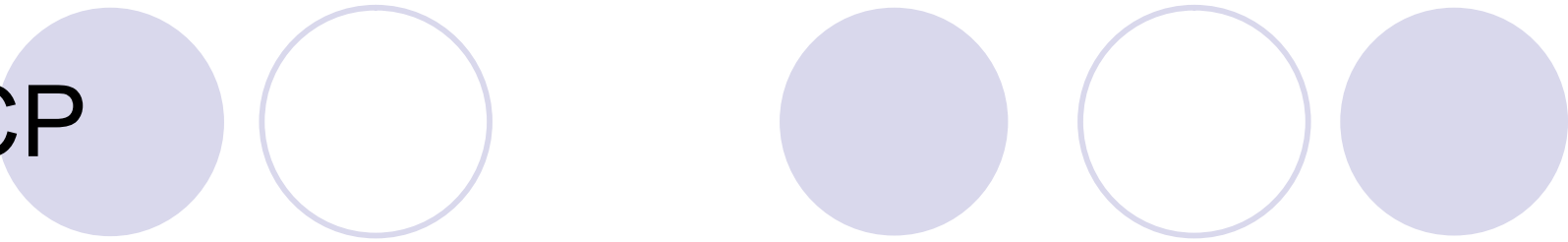
Получатель на своей стороне организует буфер длиной WindowSize для приема данных на случай, если сегменты придут не в правильной последовательности.

Когда приходят все ожидаемые данные, получатель отправляет **одно** подтверждение, увеличивая указатель SN на WindowSize.

Если сегменты теряются (ненадежная линия, приложение на компьютере клиента не успевает обрабатывать данные ...), получатель запрашивает повторную отправку всей порции данных.

Поэтому если сегменты приходят надежно, WindowSize автоматически увеличивается, если нет – уменьшается.

TCP



Завершение TCP-соединения

Инициатором разрыва соединения может быть любая сторона.

Для завершения соединения стороны обмениваются сегментами с выставленным флагом FIN и подтверждают друг другу их получение.

TCP

- https://ru.wikipedia.org/wiki/Transmission_Control_Protocol
- <http://ciscotips.ru/tcp>
- <https://hackware.ru/?p=12935#53>
- <https://www.osp.ru/nets/1999/01-02/143890>
- <https://iia-rf.ru/scarves/protokoly-ip-tcp-udp-naznachenie-otliche-protokol-udp-princip-raboty/>
- <https://studfile.net/preview/3545851/page:12/>
- <http://www.realcoding.net/articles/glava-2-osnovy.html>
- <https://networkguru.ru/kakie-parametri-vliyaut-na-proizvoditelnost-prilozheniy/>

4-й (транспортный) уровень модели OSI

Назначение:

Прямая связь между конечными пунктами и надежность

Оборудование:

реализуется программно

Тип обрабатываемых данных:

UDP-датаграммы и TCP-сегменты

Адресация:

сокет (IP-адрес + программный порт)