

Программирование на современном C++

Введение в разработку сетевых приложений.



Отметьтесь на портале!

- Посещение необязательное, но тем, кто пришёл, следует отмечаться на портале в начале каждого занятия
- Это позволяет нам анализировать, какие занятия были более или менее интересны студентам, и менять курс в лучшую сторону
- Также это даст возможность вам оставить обратную связь по занятию после его завершения


пн, 1 ноября	вт, 2 ноября	ср, 3 ноября	чт, 4 ноября	пт, 5 ноября	сб, 6 ноября
Нет занятий	<div>18:00 Углубленный C/C++ (w... п</div> <div>Обработка исключительных ситуаций. Шаблоны классов и методов. Обобщенное и безопасное программирование</div> <div>А. Халайджи</div> <div>18:00 Углубленный C/C++ (ML) п</div> <div>Обработка исключительных ситуаций. Шаблоны классов и методов. Обобщенное и безопасное программирование</div> <div>А. Халайджи</div>	Нет занятий	Нет занятий	Нет занятий	Нет занятий

Ссылка на Zoom РК сегодня в 18:00

Безопасность интернет-приложений (третий семестр)

Подключиться к конференции Zoom
<https://mailru.zoom.us/j/98586081818?pwd=eWxwSDdCbHhQNnNwQU5iblFvU2dTZz09>

Идентификатор конференции: 985 8608 1818
Код доступа: 785885

 Алексей Набережный 55 минут назад

★ 0 💬 0 ↓ 0 ↑

Запись прошлой лекции (+ что почитать перед РК)

Безопасность интернет-приложений (третий семестр)

Углублённый C/C++

Лекция 5

📍 Онлайн - ML

Отметьтесь, что вы пришли на занятие. Так вы улучшите свою посещаемость и вас увидит преподаватель в своём "Журнале посещений".

Отметиться

Оставьте отзыв о занятии и мы сможем улучшить учебный процесс.

Оставить отзыв

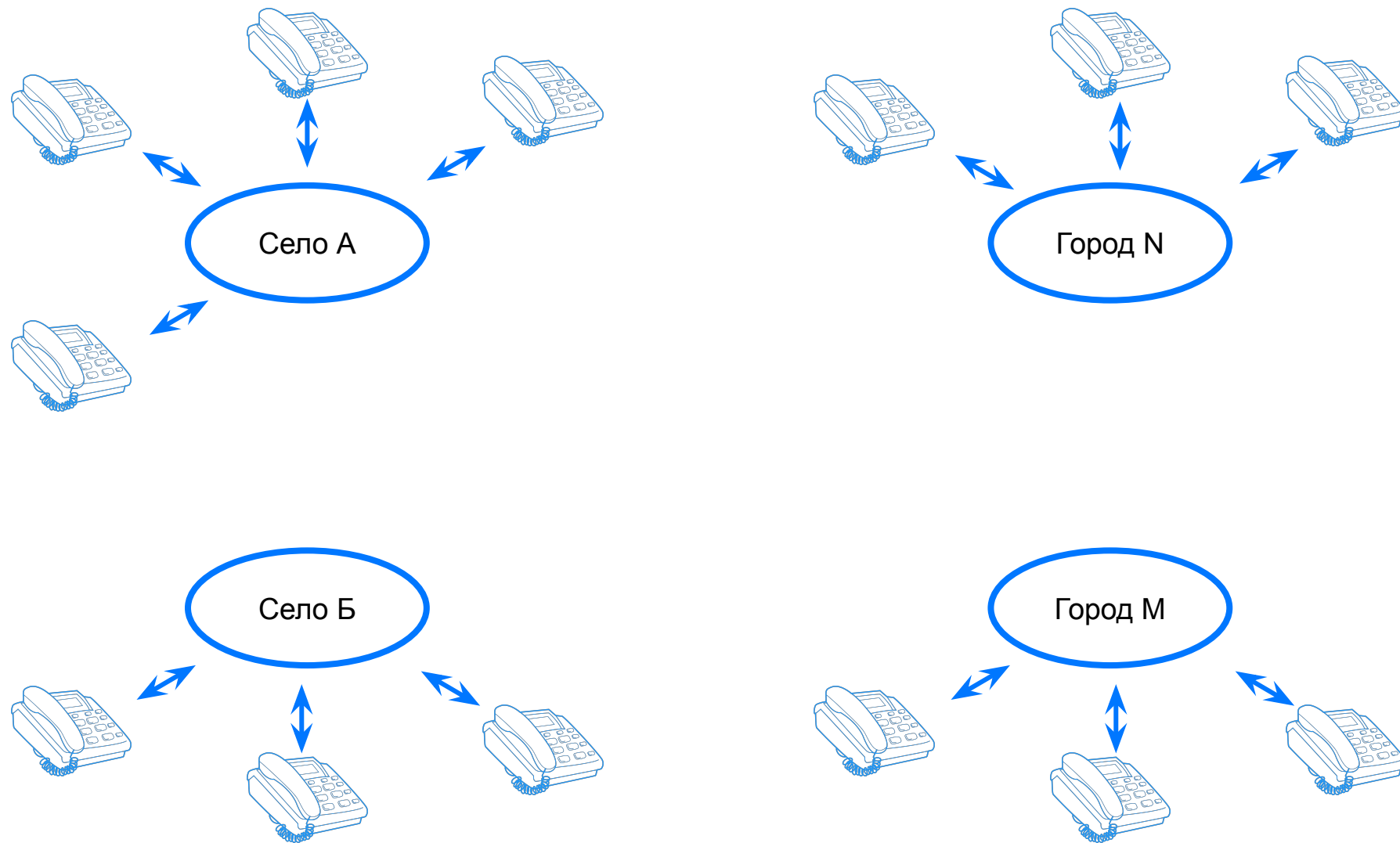
План лекции

- История интернета
- Модель OSI и TCP/IP
- Перерыв
- Сокеты Беркли

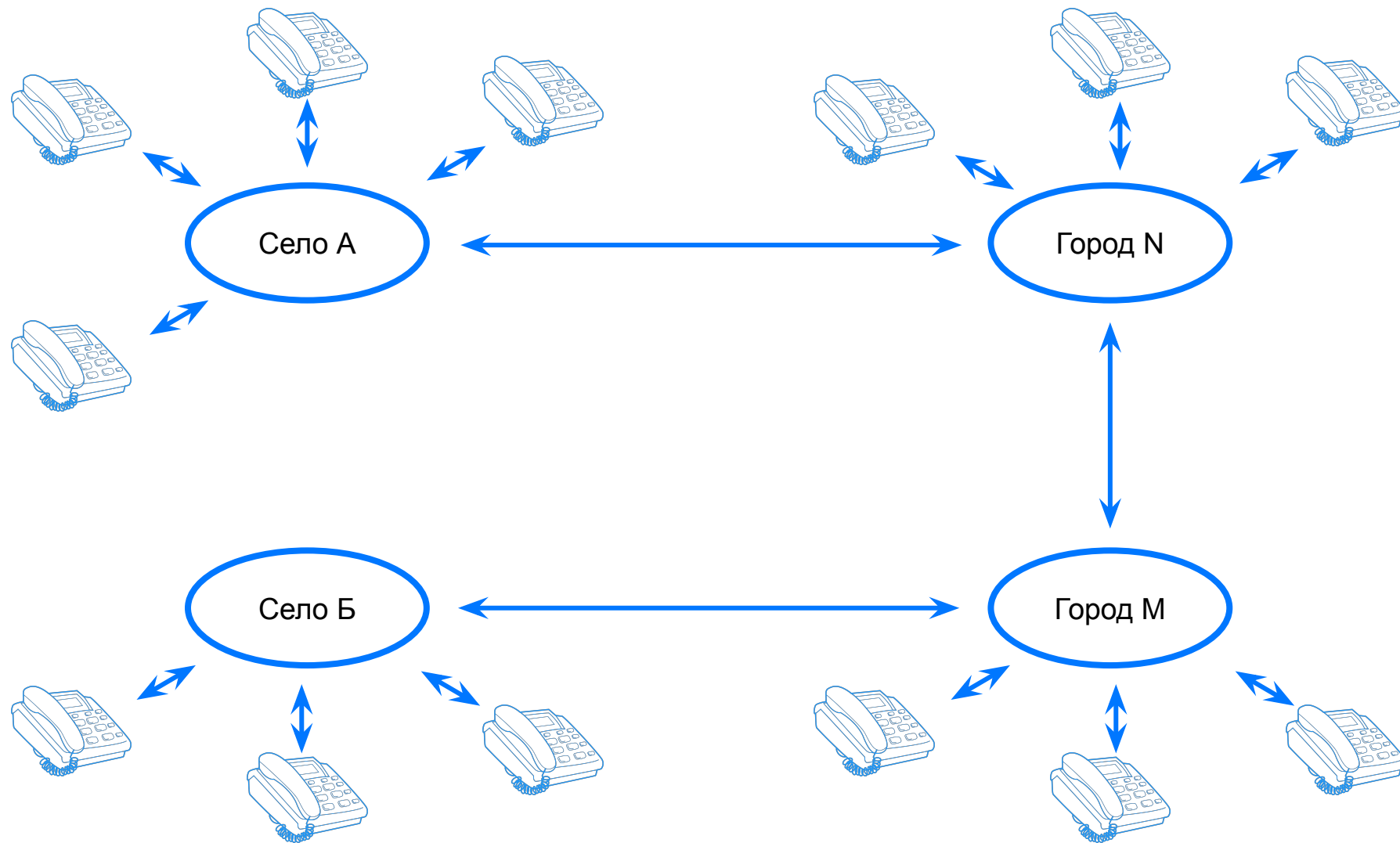
История создания сети Интернет

В 1969 г. создана сеть ARPANET Агентством Министерства обороны США по перспективным исследованиям.

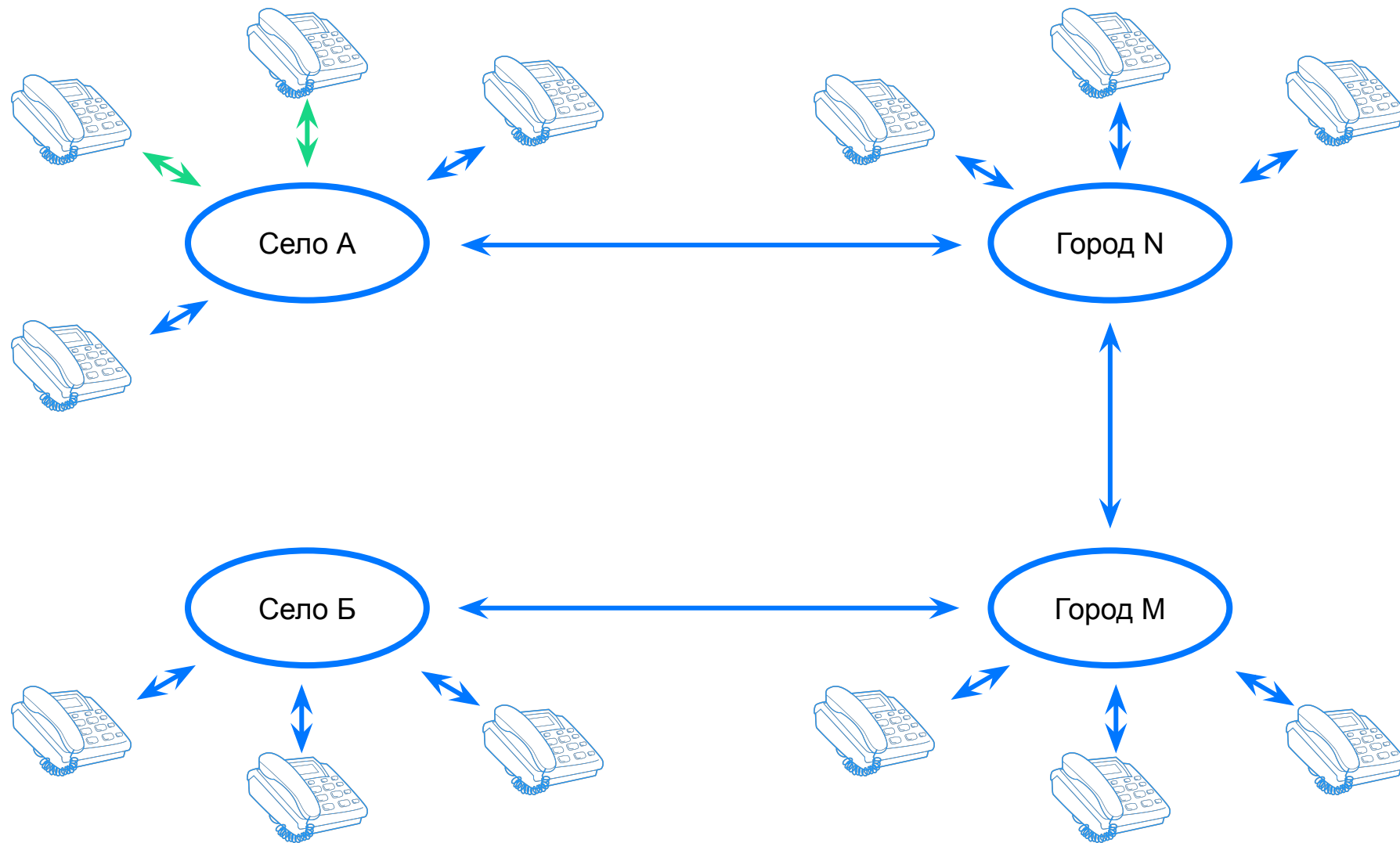
Топология телефонной связи



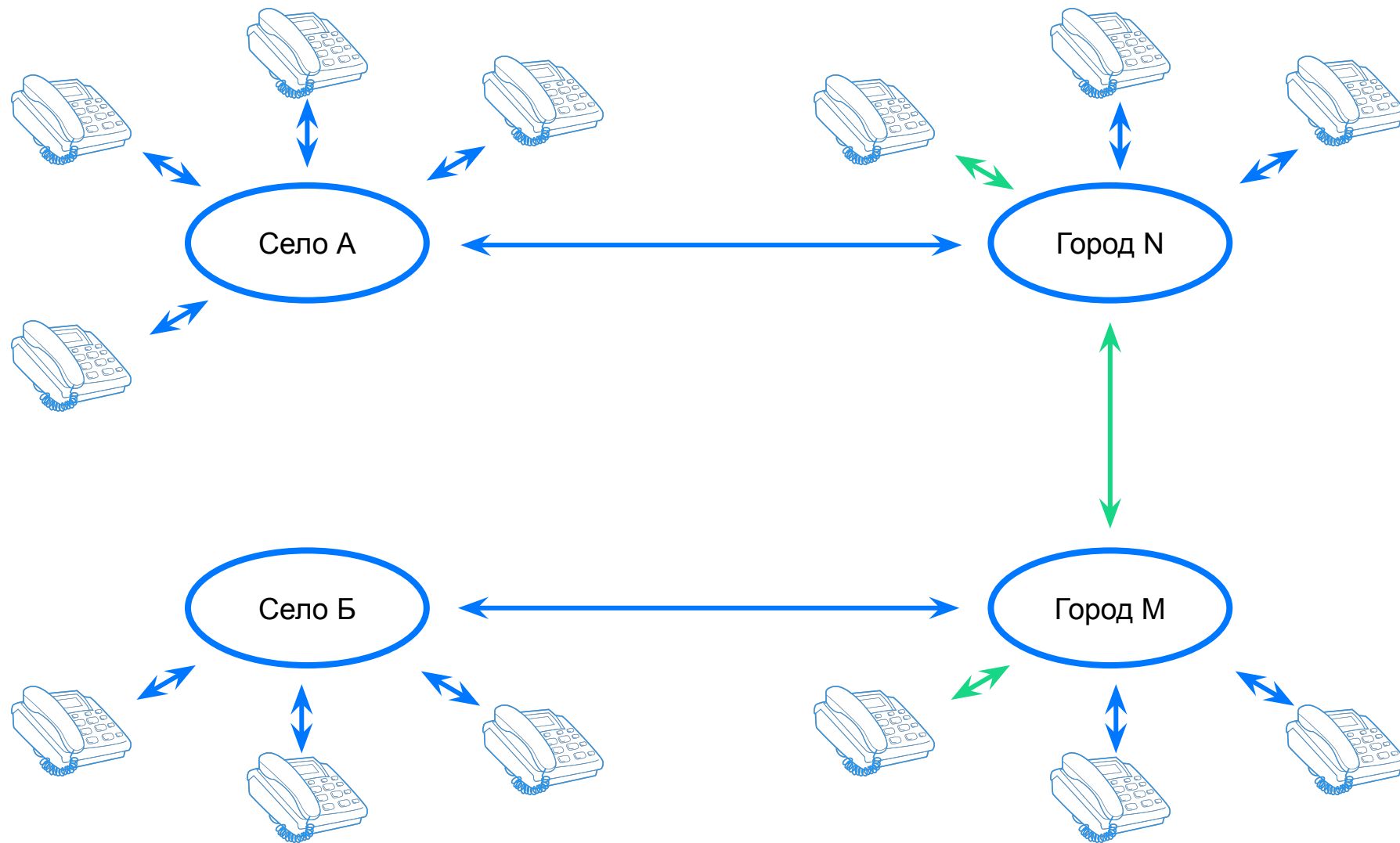
Топология телефонной связи



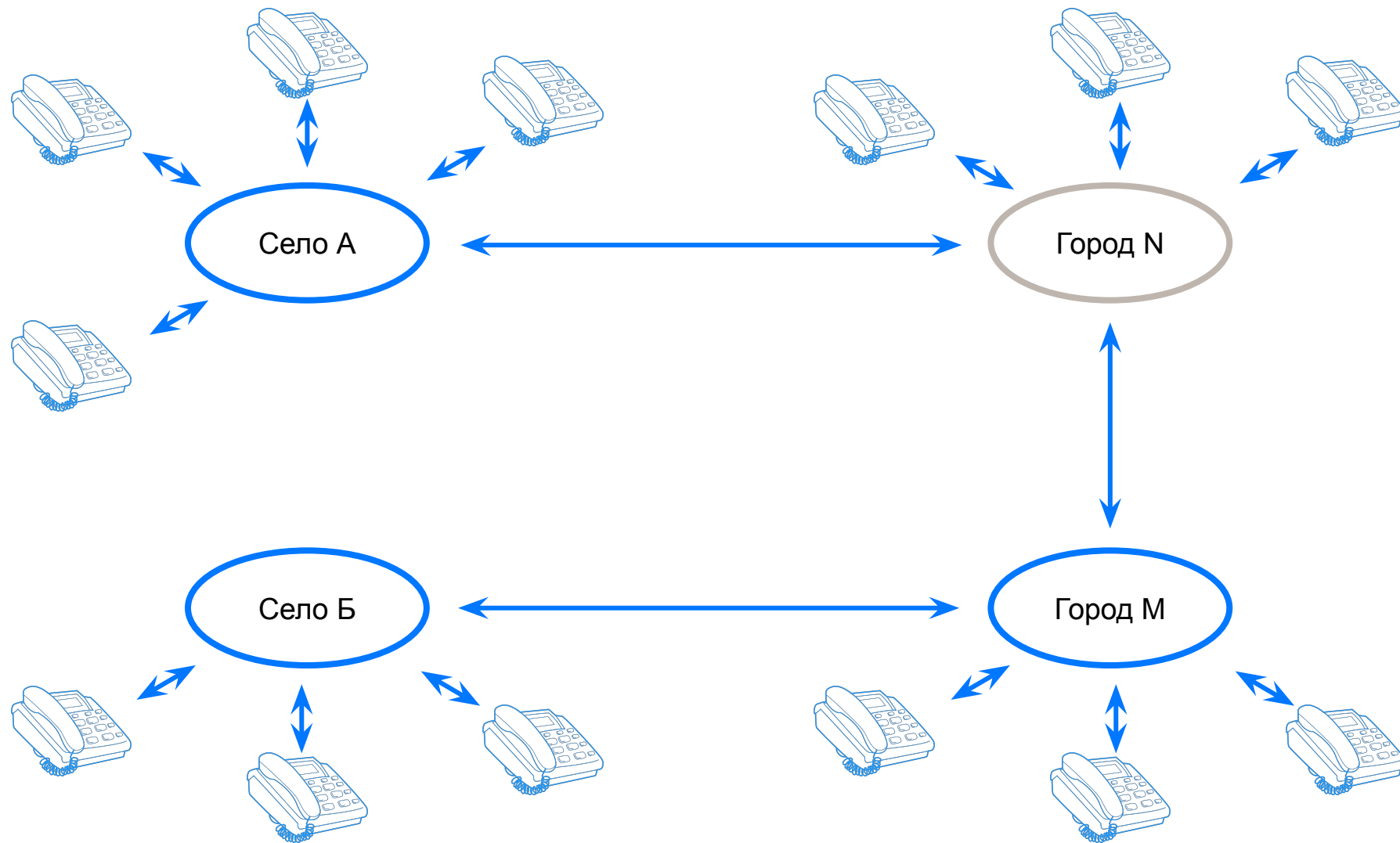
Топология телефонной связи



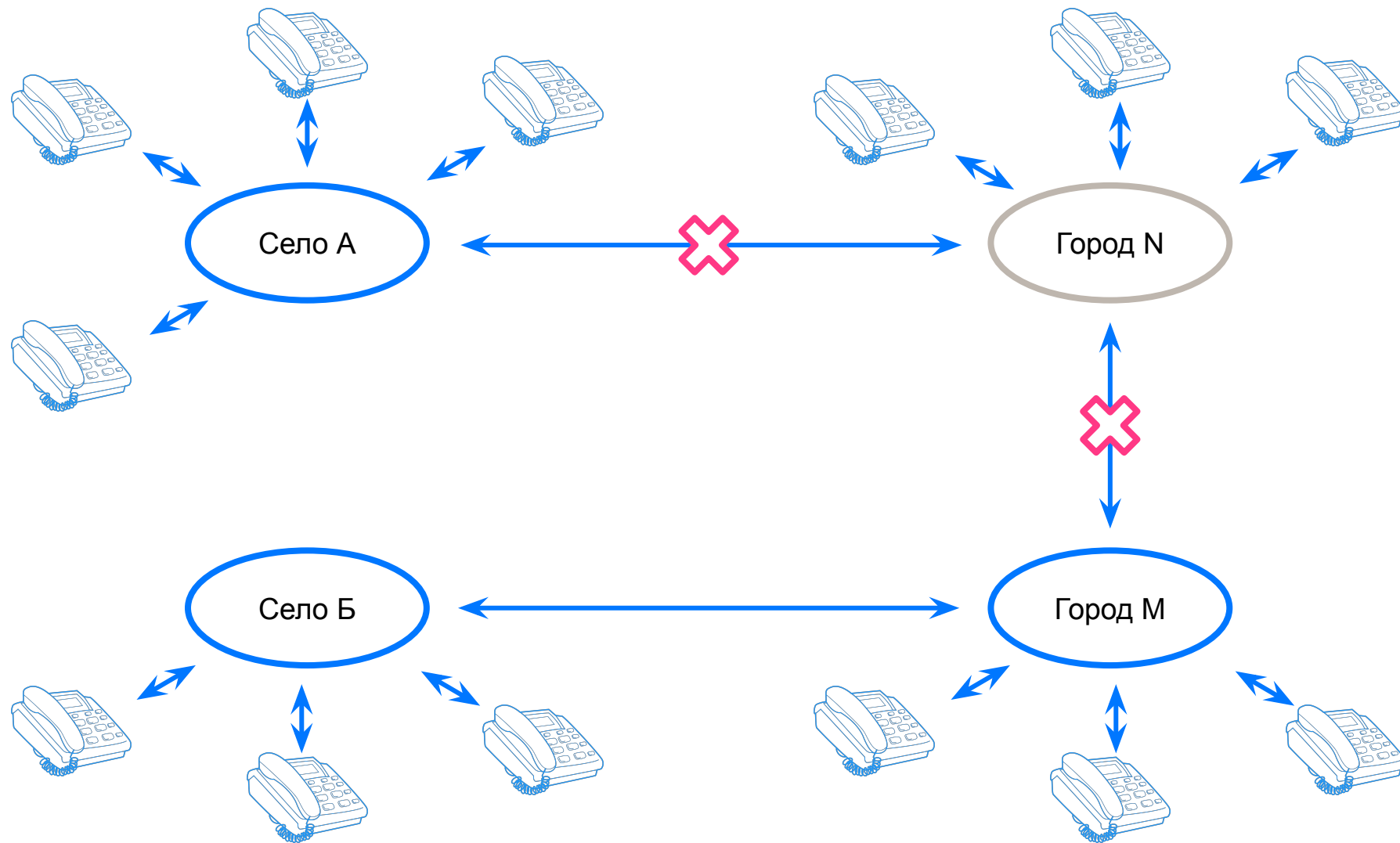
Топология телефонной связи



Топология телефонной связи



Топология телефонной связи

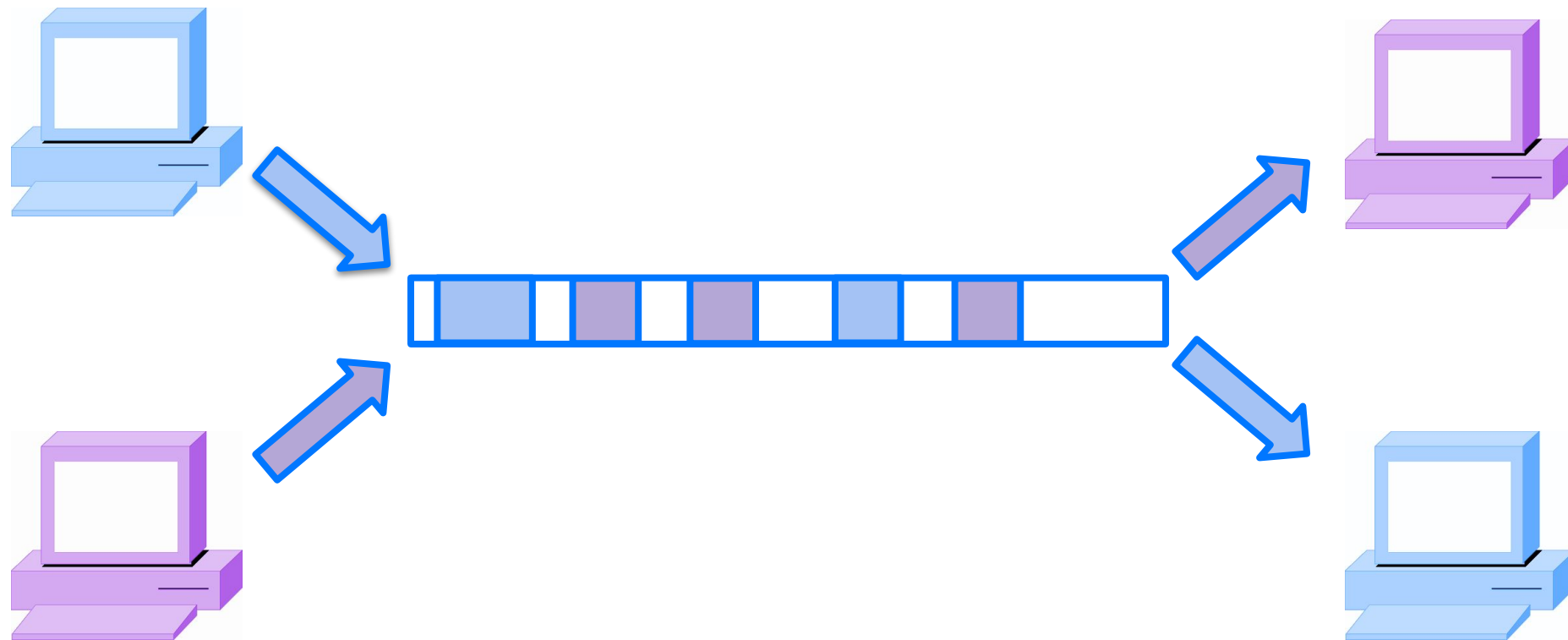


Пакетная передача данных

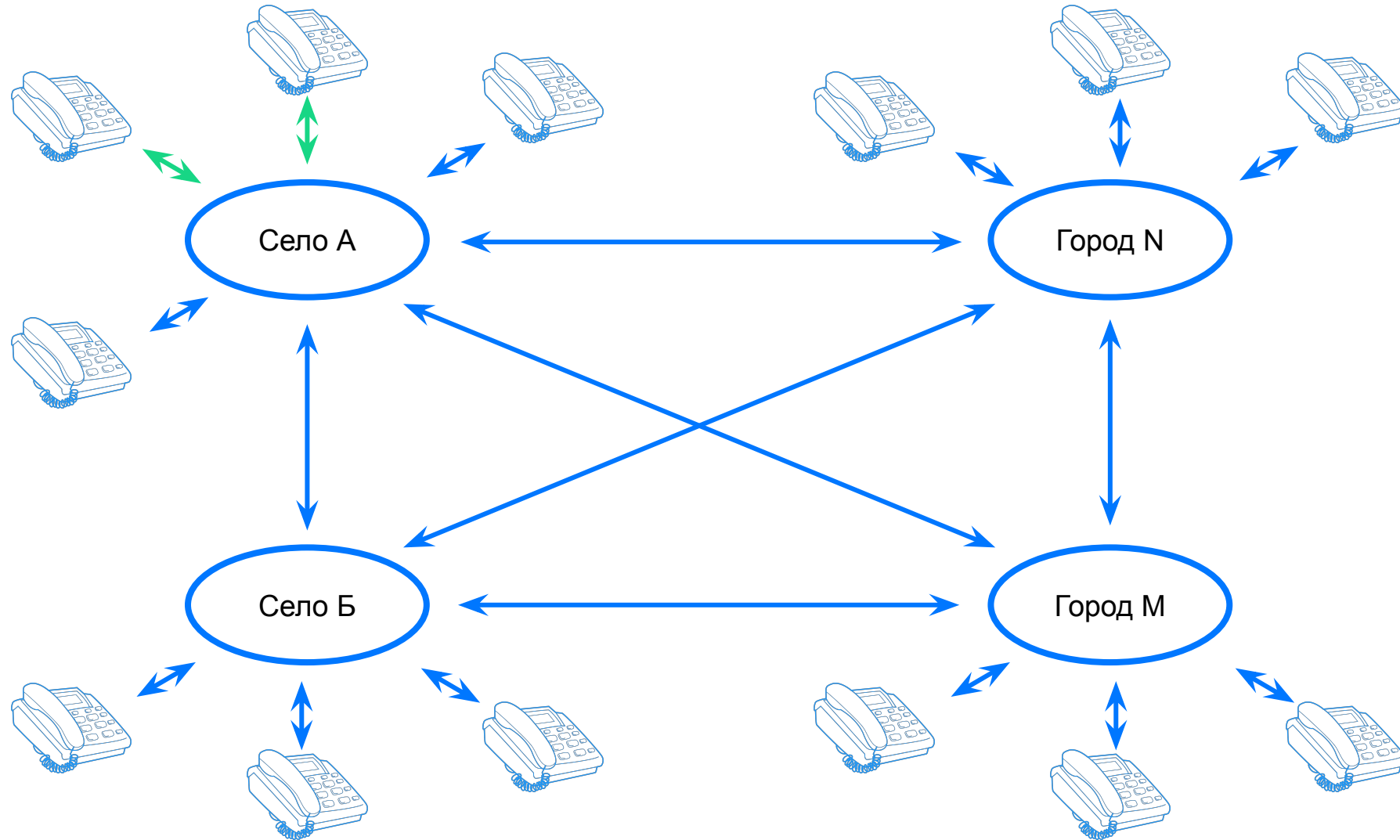
1 января 1983 года ARPANET стала первой в мире сетью, перешедшей на маршрутизацию пакетов данных

Пакетная передача данных

1 января 1983 года ARPANET стала первой в мире сетью, перешедшей на маршрутизацию пакетов данных



Пакетная передача данных



Сетевая модель OSI и TCP/IP

Уровень	Тип данных	Функции
7. Прикладной	Данные	
6. Представления		
5. Сеансовый		
4. Транспортный	Сегменты /Дейтаграммы (datagram)	Связь между конечными пунктами и надежность
3. Сетевой	Пакеты (packet)	Маршрутизация
2. Канальный	Кадры (frame)	Адресация
1. Физический	Биты	Передача сигнала

Сетевая модель OSI и TCP/IP

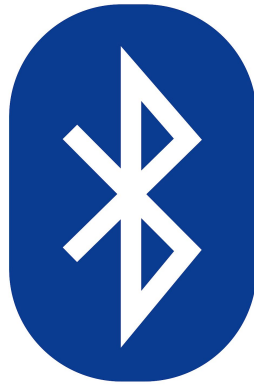
Уровень	Тип данных	Функции	TCP/IP
7. Прикладной	Данные		Прикладной
6. Представления			
5. Сеансовый			
4. Транспортный	Сегменты /Дейтаграммы (datagram)	Связь между конечными пунктами и надежность	Транспортный
3. Сетевой	Пакеты (packet)	Маршрутизация	Сетевой
2. Канальный	Кадры (frame)	Адресация	Канальный
1. Физический	Биты	Передача сигнала	

Модель OSI: Физический уровень

Как передать бит информации



1-Wire



Bluetooth



IrDA



IP посредством почтовых голубей RFC 1149

1 апреля 1990

опубликован как RFC

28 апреля 2001

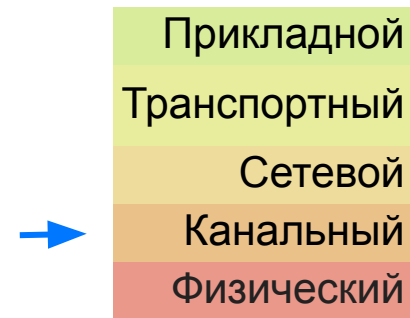
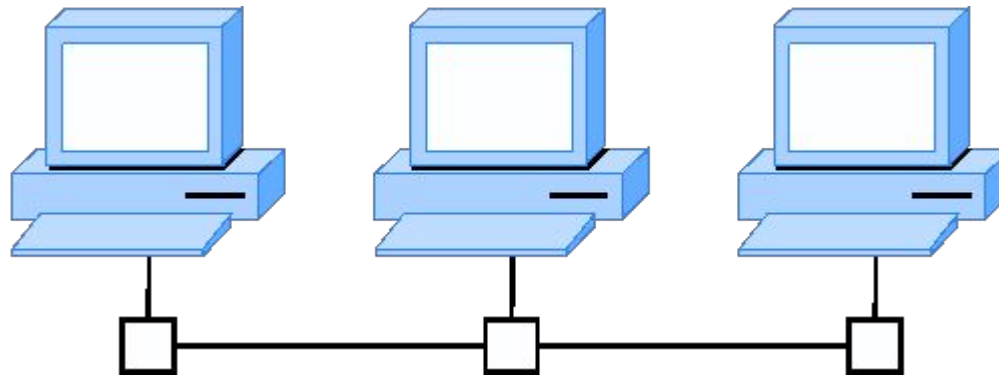
Реализован 9 ring

запросов в Норвегии



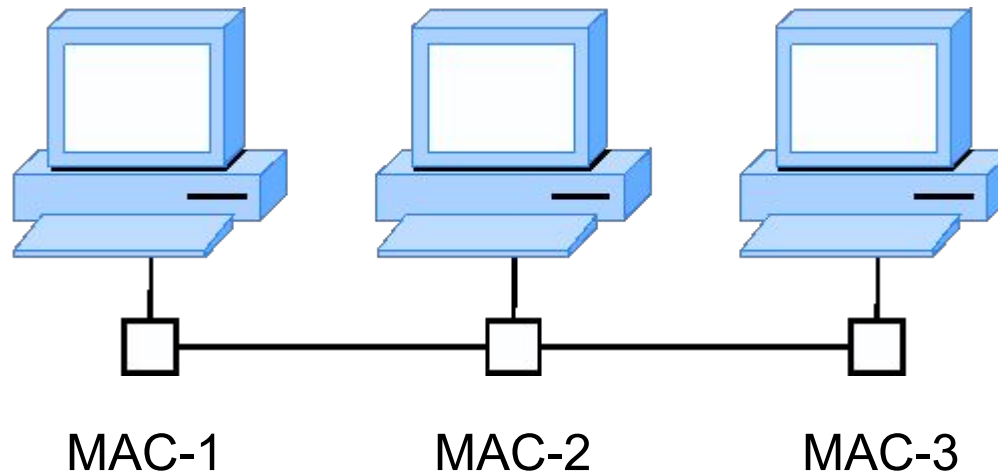
Канальный уровень

Как передать данные узлу (ПОЛУЧАТЕЛЮ ВНУТРИ 1-ОЙ СЕТИ)



Канальный уровень

Как передать данные узлу

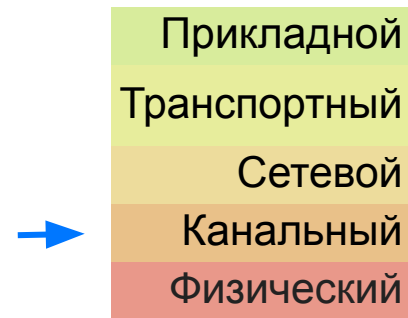


MAC Address: 10:e7:c6:f5:ee:dd

Широковещательный: ff:ff:ff:ff:ff:ff

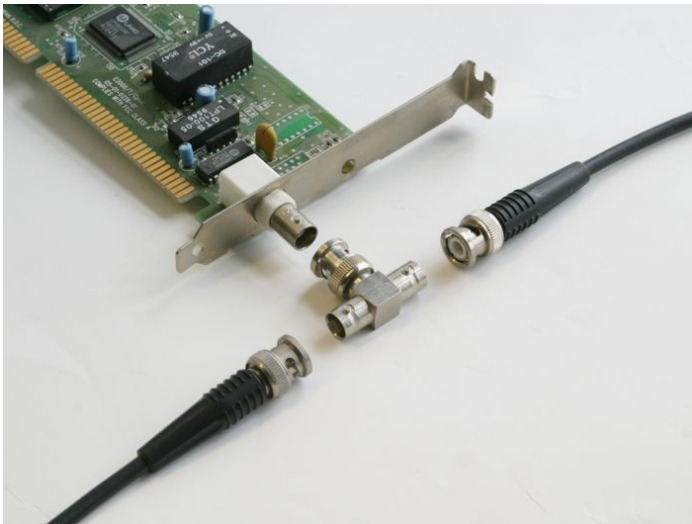


ЕСЛИ У ПАКЕТА ШИРОКОВЕЩАТЕЛЬНЫЙ АДРЕС, ТО КАЖДЫЙ УЗЕЛ В СЕТИ СЧИТАЕТ, ЧТО ПАКЕТ ПРЕДНАЗНАЧЕН ЕМУ



Примеры физического уровня

10BASE-2



Хаб(Не свитч)

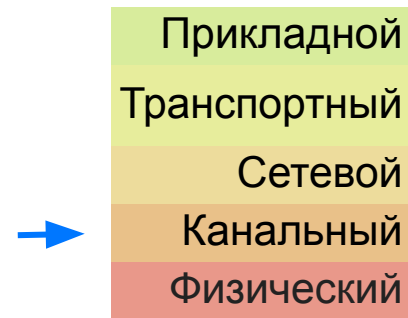
Дублирует трафик на все порты



Канальный уровень, протоколы

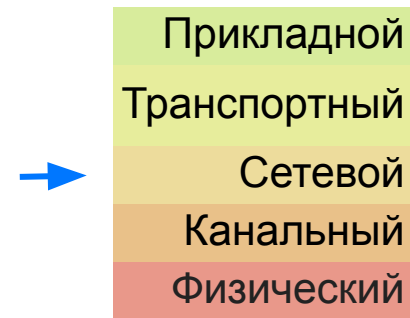
Ethernet 2 frame

MAC dest	MAC source	Ethertype (IPv4, IPv6, ARP, Wake-on-LAN)	Payload
6 Байт	6 Байт	2 Байта	46-1500 Байт



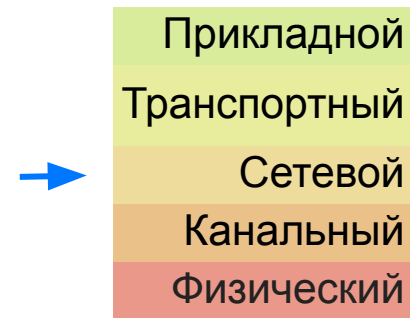
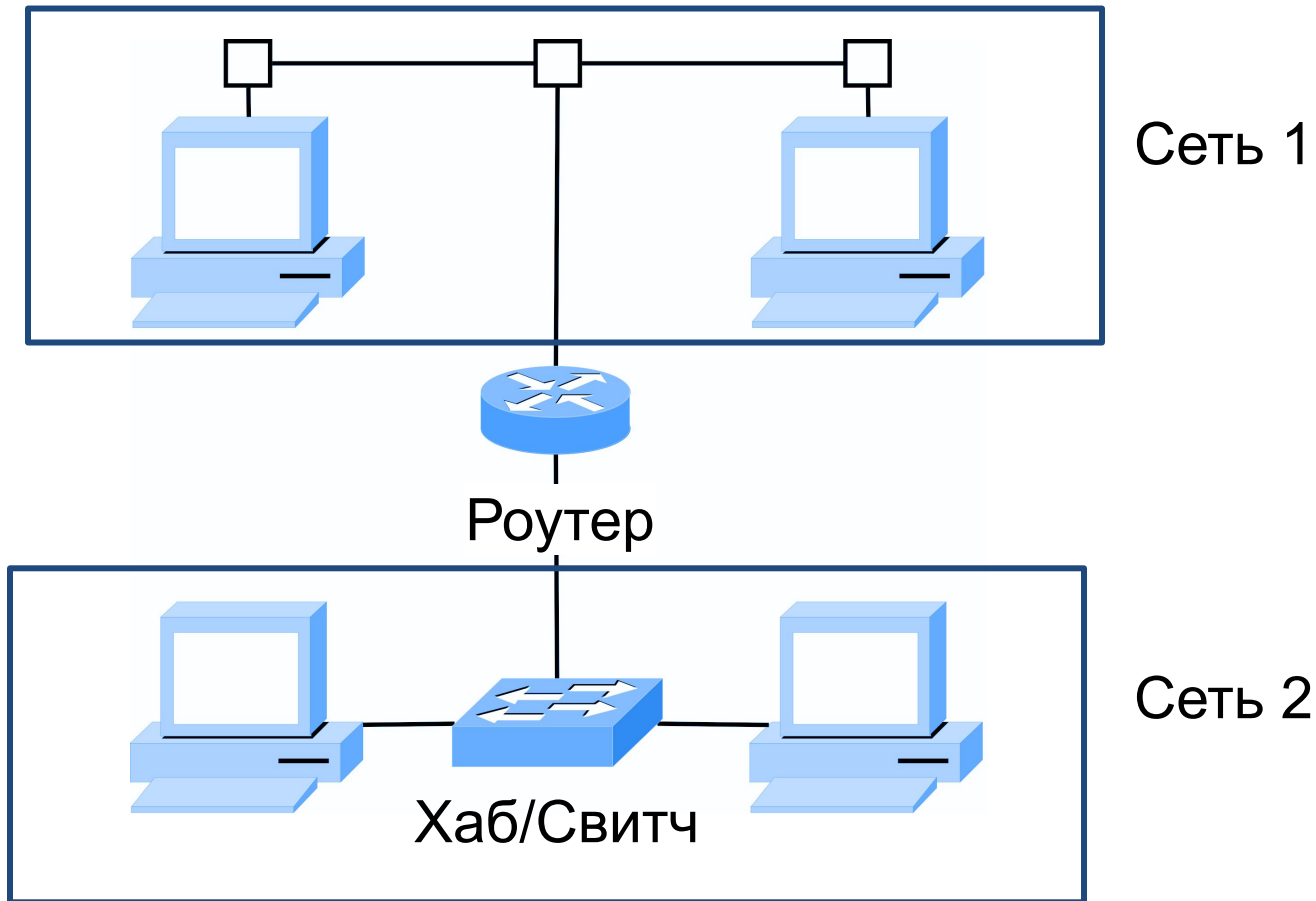
Сетевой уровень

Как послать пакет из одной сети в другую?



Сетевой уровень

Как послать пакет из одной сети в другую?

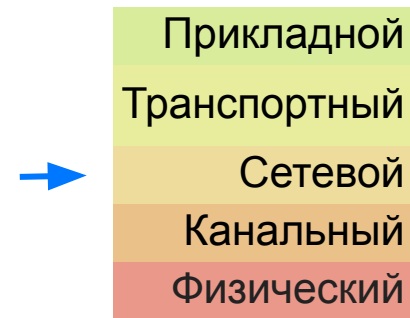


IP адрес

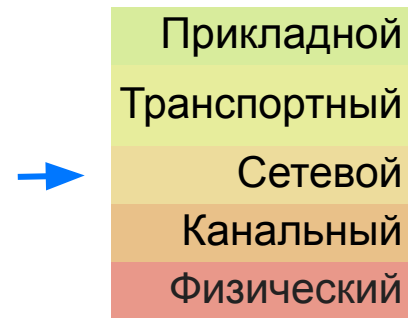
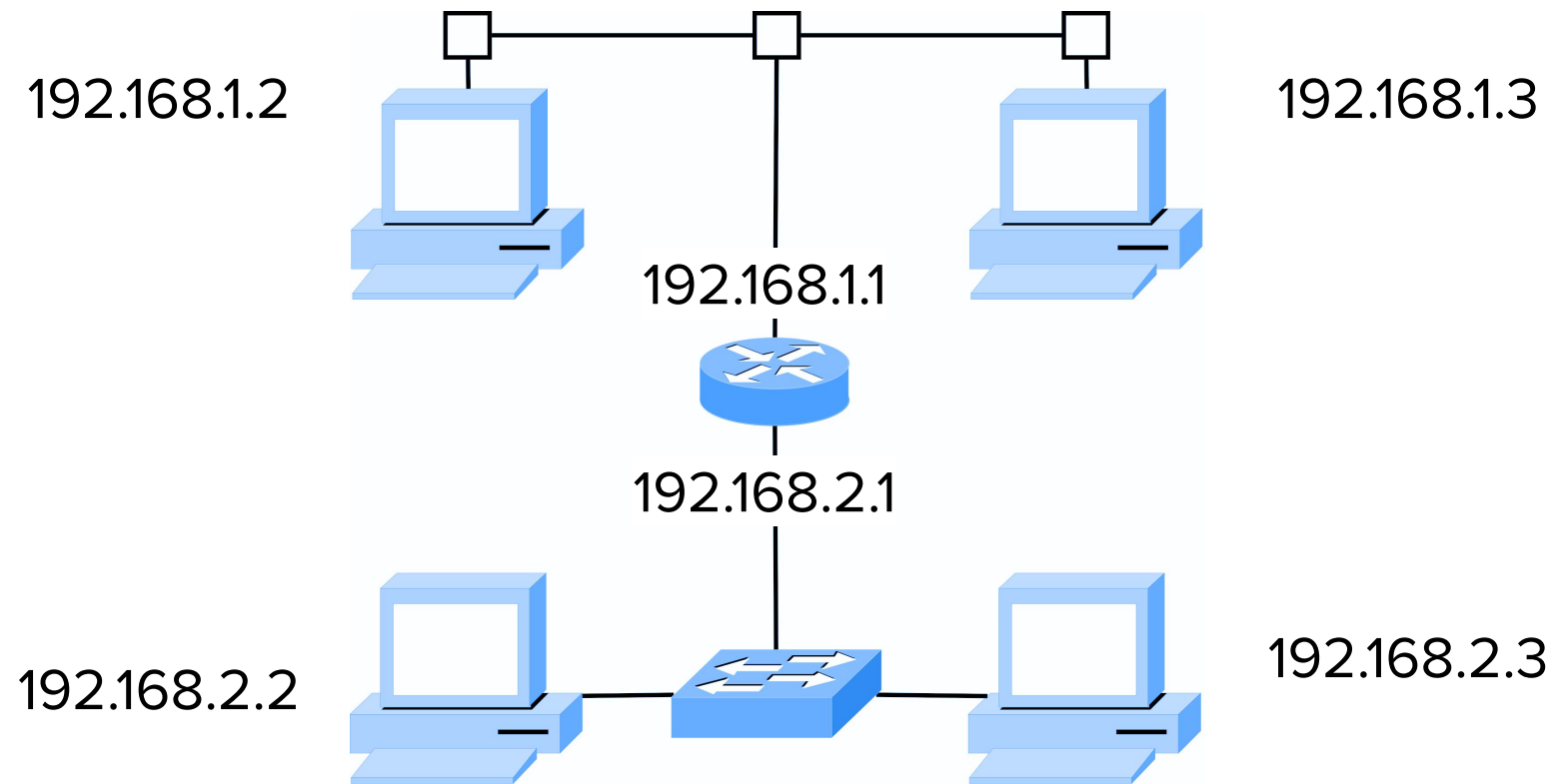
192.168.1.2 - 4 байтное число

Может быть назначен

Идентифицирует узел в сети

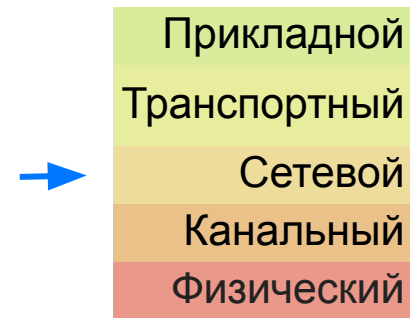
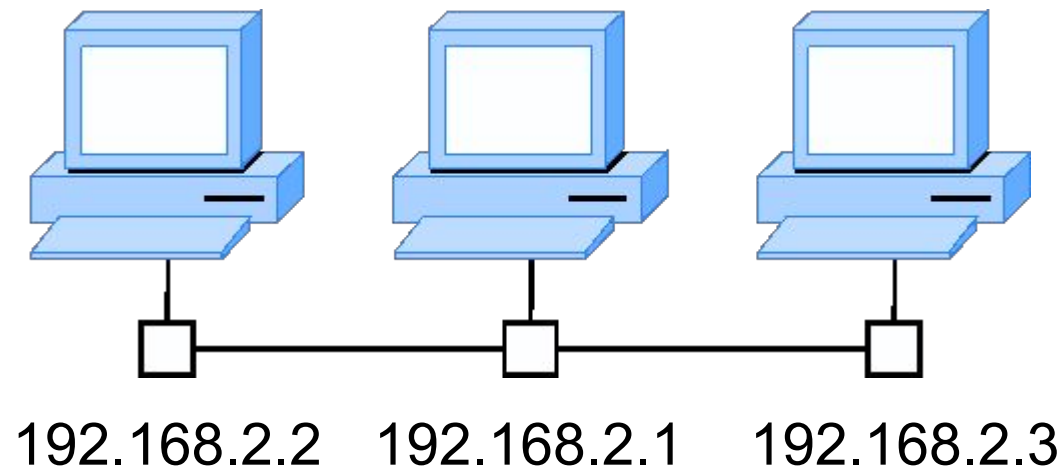


Internet Protocol



Передача пакета в своей подсети

Нужен MAC-адрес получателя



Как получить MAC по IP адресу

ARP (Address Resolution Protocol):

Узел отправляет участникам сети широковещательный запрос: Кто имеет такой адрес, сообщить отправителю (MAC + IP).

Я: "КОНКРЕТНЫЙ IP" + "КОНКРЕТНЫЙ MAC" --> ОТПРАВЛЯЕТСЯ ШИРОКОВЕЩАТЕЛЬНЫЙ ЗАПРОС(ПАКЕТ) НА ВСЕ MAC-АДРЕСА ПОДСЕТИ

Получивший отвечает отправителю

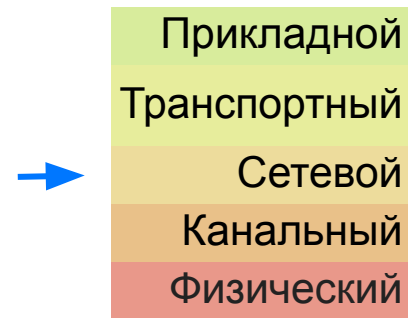
ТАК ВЫЯСНЯЕТСЯ MAC-АДРЕС

Посмотреть сетевые интерфейсы:

```
ifconfig
```

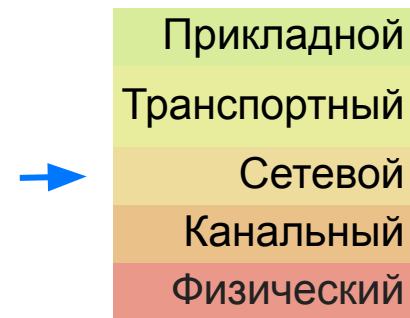
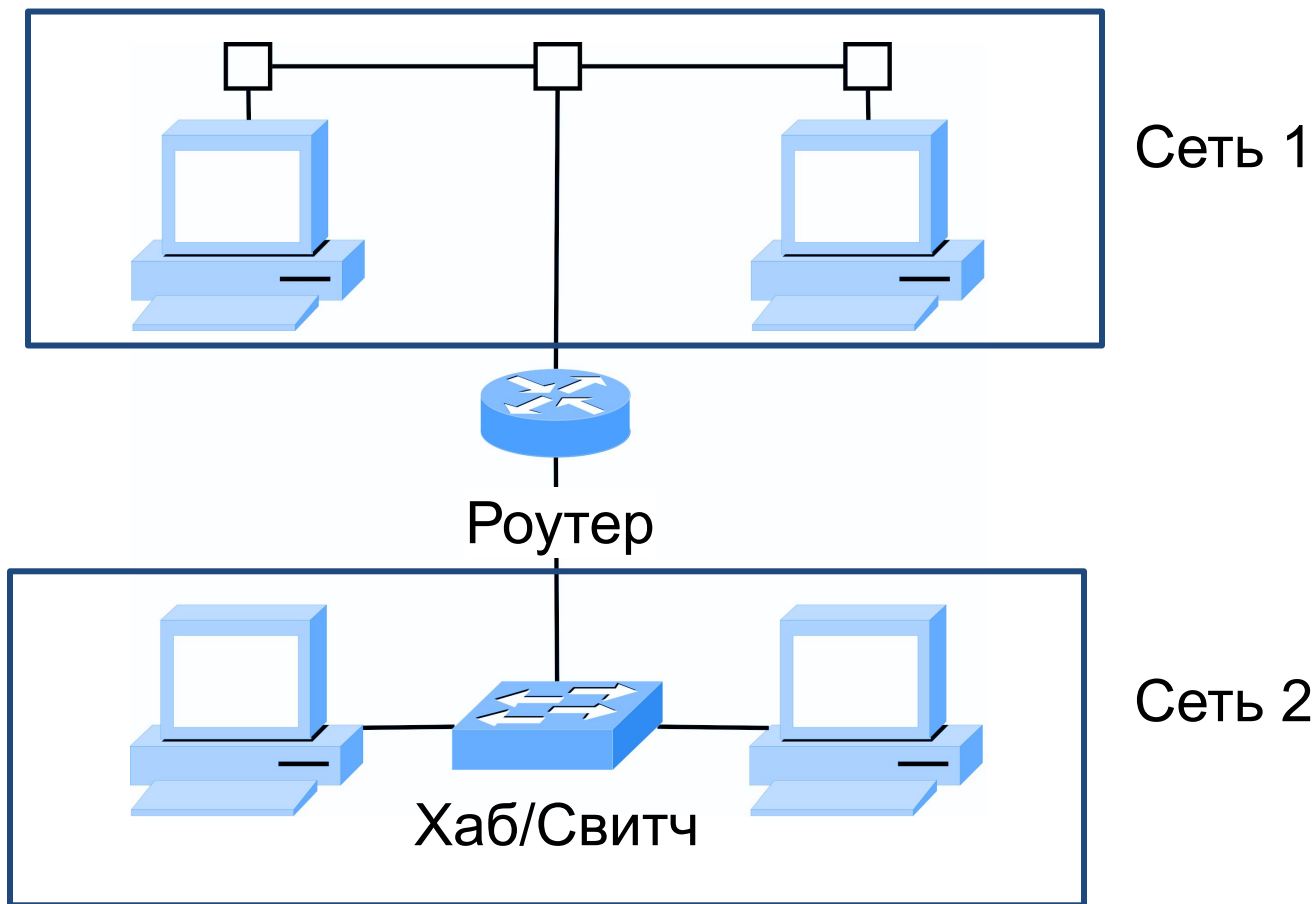
Запросить arp:

```
arping -I wlp3s0 192.168.1.1
```



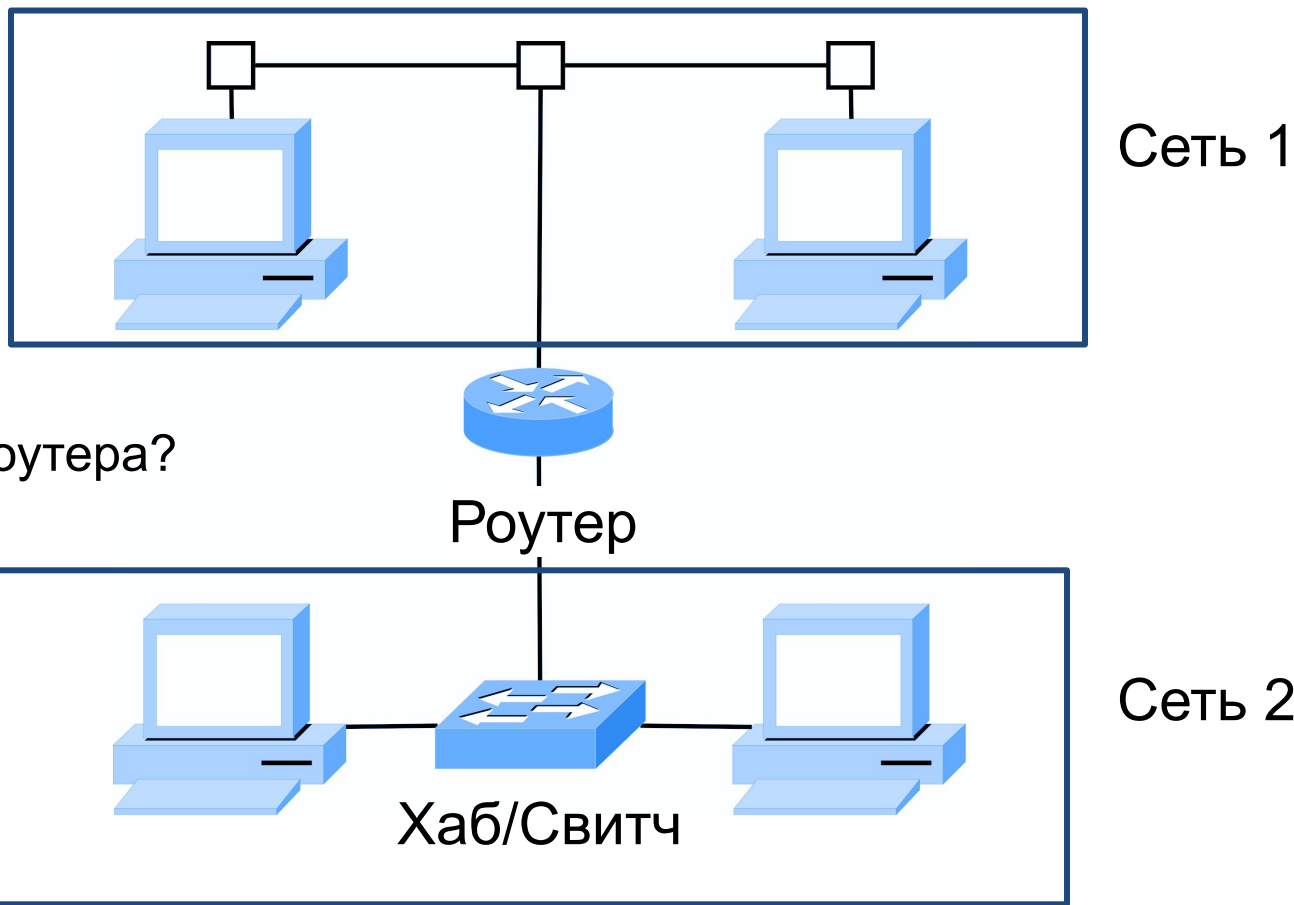
Послать пакет в другую подсеть

А как понять что получатель “за роутером”?

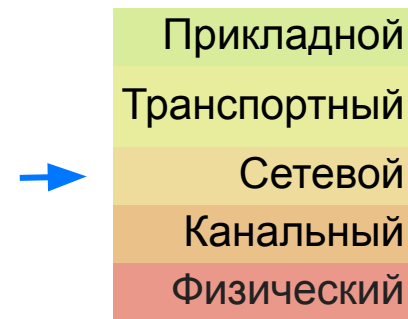


Послать пакет в другую подсеть

А как понять что получатель “за роутером”?



Какой IP адрес у роутера?



IPv4

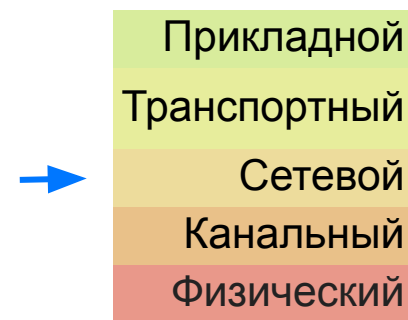
В пакете IPv4 отправляется IP адрес отправителя и получателя

На узле на каждом сетевом адаптере устанавливаются:

- Сетевой адрес
- Маска подсети (Какие адреса в его подсети)
- Адрес шлюза (Кому слать пакеты в другую подсеть)

СЕТЕВОЙ АДРЕС АССОЦИИРОВАН С КАКИМ-ЛИБО СЕТЕВЫМ АДАПТЕРОМ(КАБЕЛЬ, РОУТЕР)

В РАМКАХ ОДНОЙ ПОДСЕТИ РОУТЕР МОЖЕТ БЫТЬ НЕ ЗАДЕЙСТВОВАН, ДОСТАТОЧНО, НАПРИМЕР, СЕТЕВОГО КОНЦЕНТРАТОРА(ХАБА), ОБЩЕНИЕ В РАМКАХ 1-ОЙ ПОДСЕТИ ОСУЩЕСТВЛЯЕТСЯ ЧЕРЕЗ ARP-ЗАПРОСЫ С УЧЕТОМ ТОГО, ЧТО ОПРАВИТЕЛЬ ЗНАЕТ IP-АДРЕС ПОЛУЧАТЕЛЯ В ЛОКАЛЬНОЙ ПОДСЕТИ



Маска подсети

IP Отправителя	IP Получателя	Маска подсети	Вывод
192.168.1.2	192.168.1.3	255.255.255.0	Получатель в подсети
192.168.1.2	192.168.20.10	255.255.255.0	Получатель не в подсети
192.168.1.2	192.168.20.10	255.255.0.0	Получатель в подсети

ДЕЛАЕМ
ARP-ЗАПРОС,
ВЫЯСНЯЕМ
MAC-АДРЕС-ОТПРА
ВЛЯЕМ

запрос на vk.com --> выяснение ip-адреса получателя через dns --> определение подсети получателя с помощью собственного ip и маски подсети --> если своя подсеть: выясняем мас-адрес через arp, отправляем, если нет, выясняем мас адрес шлюза(домашнего роутера) также по ip получателя через arp --> ВСЕГДА НУЖЕН MAC АДРЕС ПОЛУЧАТЕЛЯ, ИНАЧЕ МАШИНА ФИЗИЧЕСКИ НЕ ПОНИМАЕТ, ЧТО ПАКЕТ ПРЕДНАЗНАЧЕН ЕЙ --> отправляем пакет

Как отправить пакет по IP адресу

дом. роутер видит, что у получателя другая подсеть(например, сравнивает ее на основе ip адреса от провайдера) --> выясняет мас-адрес следующего шлюза(например, на уровне провайдера(подъезд, еще что-то)) --> отправляет пакет туда --> и так пока пакет не дойдет до получателя (НА ПРОТЯЖЕНИИ ВСЕГО ПУТИ ПАКЕТА КОНЕЧНЫЙ IP НЕ МЕНЯЕТСЯ, MAC-АДРЕСА - МЕНЯЕТСЯ)

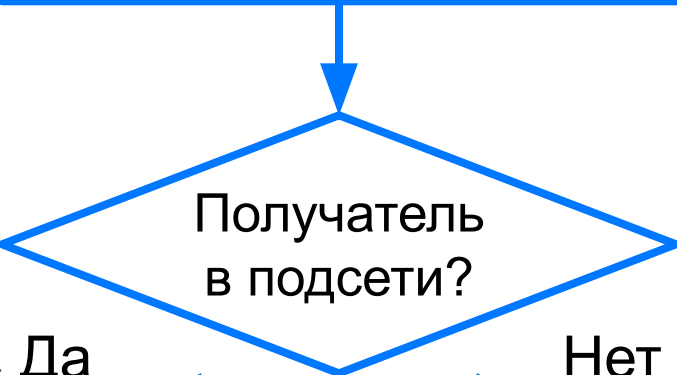
БЕЛЫЙ IP - УНИКАЛЬНЫЙ IP АДРЕС СЕТИ, ВИДЕН ВСЕМ, ДО НЕГО МОЖНО ДОСТУЧАТЬСЯ
ДИНАМИЧЕСКИЙ БЕЛЫЙ IP - АНАЛОГИЧНО, НО САМ АДРЕС МЕНЯЕТСЯ

СЕРЫЙ IP - НЕ УНИКАЛЬНЫЙ IP, КОТОРЫЙ ПОВТОРЯЕТСЯ ОТ ПОЛЬЗОВАТЕЛЯ К ПОЛЬЗОВАТЕЛЮ, ОТ НЕГО/К НЕМУ ИНФОРМАЦИЯ ДОХОДИТ БЛАГОДАРЯ NAT: ШЛЮЗ(УСТРОЙСТВО, ИМЕЮЩЕЕ БЕЛЫЙ IP, ПОДМЕНЯЕТ IP ИСХОДНОГО ОТПРАВИТЕЛЯ НА СВОЙ, ПОТОМ ПРОИСХОДИТ ОБРАТНАЯ ПОДМЕНА)

ПОДМЕНУ МОЖНО ДЕЛАТЬ, НАПРИМЕР, ПРИ УСЛОВИИ, ЧТО ПАКЕТ ПРИШЕЛ НА КОНКРЕТНЫЙ ПОРТ УСТРОЙСТВА С БЕЛЫМ IP, ТОГДА В ПАМЯТЬ УСТРОЙСТВА ЗАПИСЫВАЕТСЯ IP ИСХОДНОГО ПОЛУЧАТЕЛЯ, ДО ТОГО ДОХОДИТ ПАКЕТ, ДАЛЕЕ - БЕРЕМ ИНФУ ИЗ ПАМЯТИ И ДЕЛАЕМ ПОДМЕНУ

ДАЛЕЕ ВСЕ РОУТЕРЫ ДОМА - ТОЖЕ СЕРЫЕ IP, МОЖЕТ БЫТЬ 1 БЕЛЫЙ IP НА УРОВНЕ ДОМА, КОТОРЫЙ ТАКЖЕ ПОДМЕНЯЕТ ВСЕ АДРЕСА НА СВОЙ В ЗАВИСИМОСТИ ОТ ПОРТА

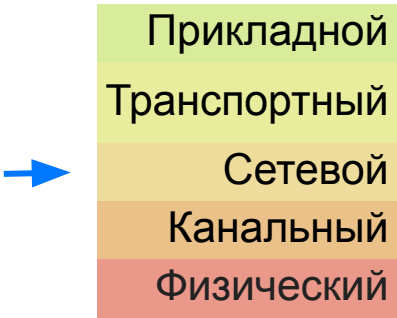
Сравнить адрес
получателя с маской
подсети



*ЕСЛИ IP УСТРОЙСТВА(РОУТЕРА) СОВПАДАЕТ С IP, КОТОРЫЙ ДЕТЕКТЯТ В ОБЩЕЙ СЕТИ ИНТЕРНЕТ, ТО У РОУТЕРА БЕЛЫЙ IP

Выяснить MAC
получателя,
отправить пакет

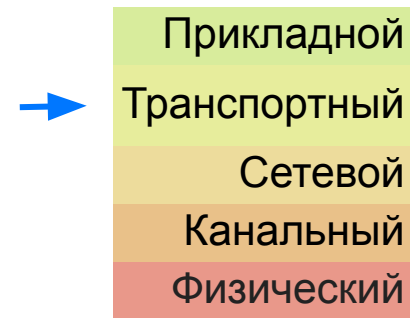
Выяснить MAC
шлюза,
отправить пакет



Транспортный уровень

Как доставить данные от точки к точке в пределах одного узла.
(между несколькими процессами/пользователями).

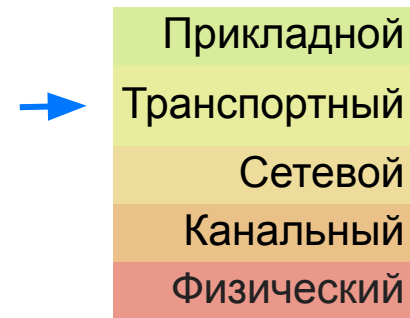
- Как сформировать логическое соединение?
- Как мультиплексировать данные?
- Как управлять соединением?
- Как гарантировать надежность соединения?



Логическое соединение, порты

Логическое соединение ассоциируется с числом, которое называется портом.

- Well Known Ports (Системные порты) 0-1023
- Registered Ports (Пользовательские) 1024-49151
- Dynamic Ports (Private Ports), from 49152-65535

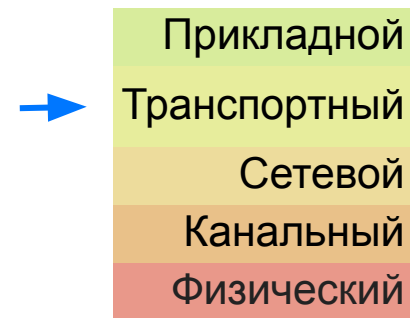


Логическое соединение, порты

Логическое соединение ассоциируется с числом, которое называется портом.

- Well Known Ports (Системные порты) 0-1023
- Registered Ports (Пользовательские) 1024-49151
- Dynamic Ports (Private Ports), from 49152-65535

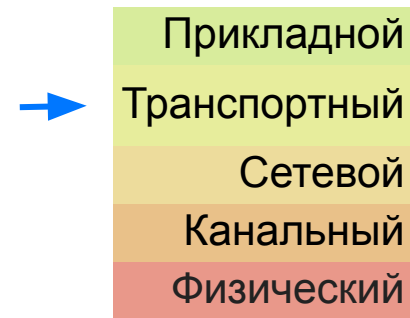
Системные	Пользовательские
20 FTP	37483 Google Drive Sync
22 SSH	2376 Docker REST API
80 HTTP / 443 HTTPS	
666 Doom	



TCP / UDP

User Datagram Protocol - простой ненадёжный протокол

Transmission Control Protocol - надёжный протокол

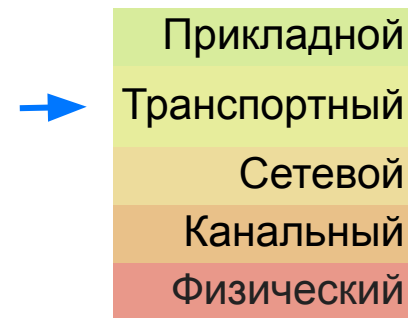


TCP / UDP

User Datagram Protocol - простой ненадёжный протокол

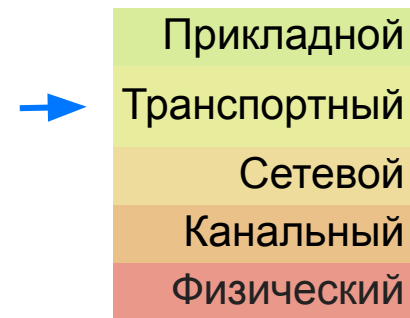
Transmission Control Protocol - надёжный протокол

TCP	UDP
Гарантирует доставку	Не гарантирует доставку
Гарантирует порядок	Не гарантирует порядок
Отслеживает состояние соединения	Нет соединения
Мультиплексирует пакеты	Мультиплексирует пакеты
Гарантирует целостность данных	Гарантирует целостность пакета
Медленный	Быстрый

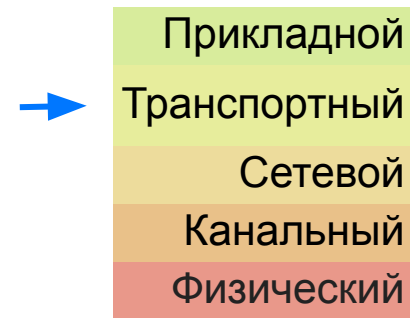
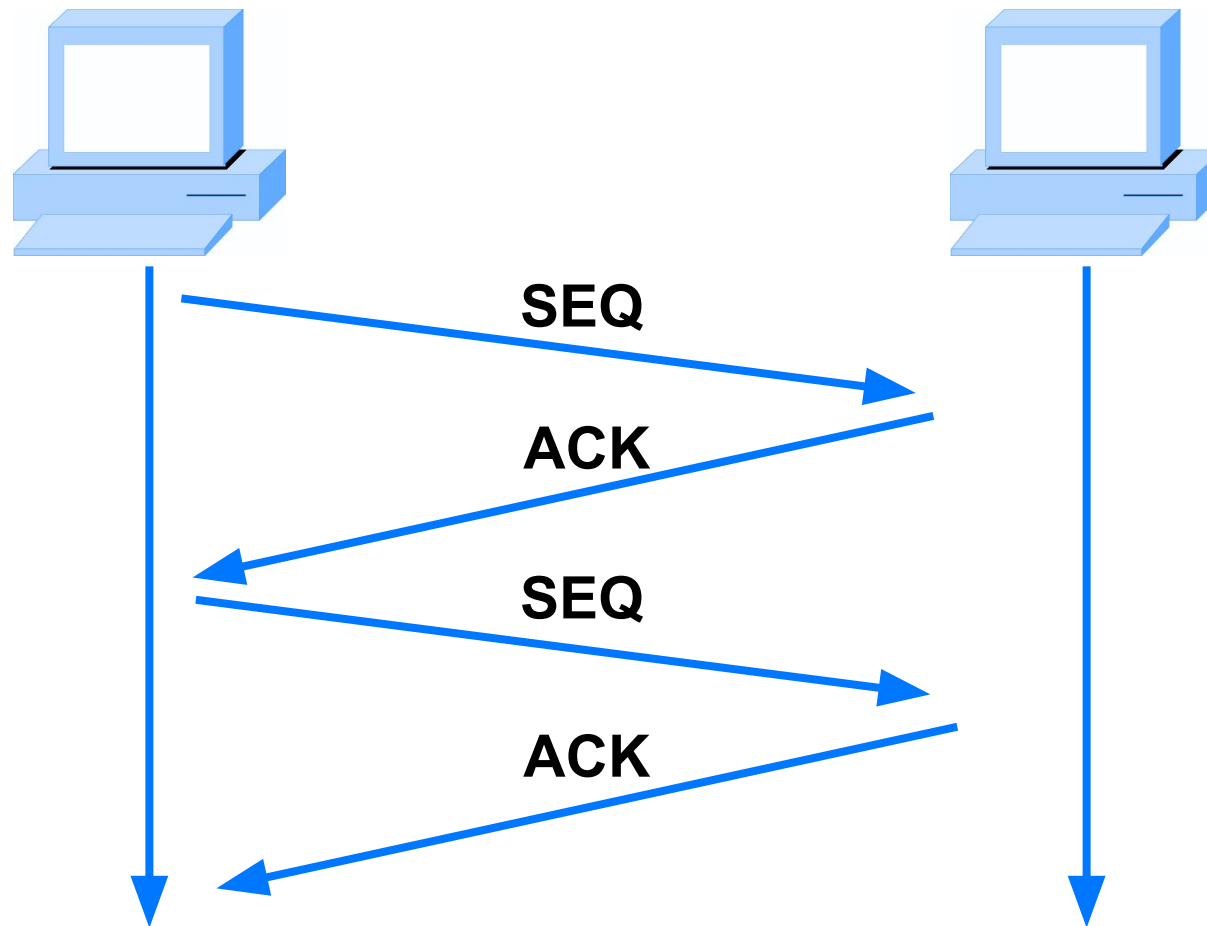


Структура UDP пакета

Биты	0 - 15	16 - 31
0-31	Порт отправителя (src)	Порт получателя (dst)
32-63	Длина датаграммы (Length)	Контрольная сумма
64-...	Данные (Data)	



TCP



Прикладной уровень

Прикладной уровень представлен протоколами, которые обеспечивают работу приложений и сервисов



Прикладной уровень

Прикладной уровень представлен протоколами, которые обеспечивают работу приложений и сервисов

Примеры:

- Telnet
- FTP
- SMTP
- RTP(UDP)
- HTTP



Перерыв

Сокеты Беркли

Сокеты Беркли - реализация (API) сетевого взаимодействия.(1982)

Разработан Калифорнийским университетом в Беркли.

socket

UDP:

```
int s = ::socket(PF_INET, SOCK_DGRAM, 0)
```

TCP:

```
int s = ::socket(PF_INET, SOCK_STREAM, 0)
```

sockaddr

sockaddr разный для разных протоколов

```
sockaddr_in addr{}; // no memset  
addr.sin_family = AF_INET;  
addr.sin_port = htons(8888);  
addr.sin_addr.s_addr = htonl(INADDR_LOOPBACK);
```

или

```
::inet_aton("192.168.1.37", &addr.sin_addr);
```

connect

sockaddr разный для разных протоколов

```
::connect(fd,  
           reinterpret_cast<sockaddr*>(&addr),  
           sizeof(addr)); // Блокируемся
```

connect

После удачного подключения, система присваивает нам случайный незанятый порт, через который с нами будет общаться сервер.

close инициирует завершение соединения клиентской стороной.

Сервер

```
int fd = socket(PF_INET, SOCK_STREAM, 0);
sockaddr_in addr;
// Заполняем addr

bind(fd,
      reinterpret_cast<sockaddr*>(&addr),
      sizeof(addr));

// Устанавливаем как слушающий сокет
listen(fd, MAX_QUEUE_SIZE);
```


Принимаем соединение

```
sockaddr_in client_addr;  
socklen_t addr_size = sizeof(client_addr);  
int client = accept(fd,  
                    reinterpret_cast<sockaddr*>(&client_addr),  
                    &addr_size);
```

Инструменты

- Wireshark
- `netstat -atnp` — Отобразить соединения в системе
 - `-a` — Отобразить и слушающие сокеты
 - `-t` — TCP сокеты
 - `-n` — числовые адреса
 - `-p` — отобразить программу и pid
- `strace -f -s 2048 ./prog_name args` - дамп системных вызовов

Инструменты

nc (netcat) — простой TCP/UDP сервер/клиент

- nc ip port — подключиться к адресу на порт
передает stdin, читает в stdout
- echo 123 | nc ip port

- nc -l port — слушать на порту
передает stdin, читает в stdout
- echo 123 | nc -l port — слушать на порту
передает stdin, читает в stdout

Спасибо за внимание!

