**Лекция 2.3. Сетевое программирование**

В данной лекции будут рассмотрены следующие вопросы:

1. Общее понимание сетей на базе протокола TCP/IP.

2. TCP и UDP.

3. Адреса, порты, сокеты, взаимодействие с сервером.

**Глава 1. Сетевое программирование**

Стек протоколов TCP/IP (Transmission Control Protocol/Internet Protocol, протокол управления передачей/протокол интернета) — сетевая модель, описывающая процесс передачи цифровых данных. Она названа по двум главным протоколам, по этой модели построена глобальная сеть Интернет. Сейчас это кажется невероятным, но в 1970-х информация не могла быть передана из одной сети в другую. Чтобы обеспечить такую возможность, был разработан стек интернет-протоколов, известный как TCP/IP.

Разработка сетевой модели осуществлялась при содействии Министерства обороны США, поэтому иногда модель TCP/IP называют DoD (Department of Defence) модель. Если вы знакомы с моделью OSI, то вам будет проще понять построение модели TCP/IP, потому что обе модели имеют деление на уровни, внутри которых действуют определённые протоколы и выполняются собственные функции.



Выше упоминалось, что модель TCP/IP разделена на уровни, как и OSI, но отличие двух моделей в количестве уровней. Документами, определяющими сертификацию модели, являются RFC 1122 и RFC1123. Эти стандарты описывают четыре уровня абстракции модели TCP/IP: прикладной, транспортный, межсетевой и канальный.

Рассмотрим четыре уровня модели:

1. Канальный уровень (link layer).

Предназначение канального уровня — дать описание тому, как происходит обмен информацией на уровне сетевых устройств, определить, как информация будет передаваться от одного устройства к другому. Информация здесь кодируется, делится на пакеты и отправляется по нужному каналу, т. е. среде передачи.

Этот уровень также вычисляет максимальное расстояние, на которое пакеты возможно передать, частоту сигнала, задержку ответа и т. д. Всё это — физические свойства среды передачи информации. На канальном уровне самым распространённым протоколом является Ethernet.

2. Межсетевой уровень (internet layer).

Глобальная сеть интернет состоит из множества локальных сетей, взаимодействующих между собой. Межсетевой уровень используется, чтобы описать обеспечение такого взаимодействия.

Межсетевое взаимодействие — это основной принцип построения интернета. Локальные сети по всему миру объединены в глобальную, а передачу данных между этими сетями осуществляют магистральные и пограничные маршрутизаторы.

Именно на межсетевом уровне функционирует протокол IP, позволивший объединить разные сети в глобальную. Как и протокол TCP, он дал название модели.

**Глава 2. Сетевое программирование**

3. Транспортный уровень (transport layer).

Постоянные резиденты транспортного уровня — протоколы TCP и UDP, они занимаются доставкой информации.

TCP (протокол управления передачей) — надёжный, он обеспечивает передачу информации, проверяя дошла ли она, насколько полным является объем полученной информации и т. д. TCP даёт возможность двум конечным устройствам производить обмен пакетами через предварительно установленное соединение. Он предоставляет услугу для приложений, повторно запрашивает потерянную информацию, устраняет дублирующие пакеты, регулируя загруженность сети. TCP гарантирует получение и сборку информации у адресата в правильном порядке.

UDP (протокол пользовательских датаграмм) — ненадёжный, он занимается передачей автономных датаграмм. UDP не гарантирует, что всех датаграммы дойдут до получателя. Датаграммы уже содержат всю необходимую информацию, чтобы дойти до получателя, но они все равно могут быть потеряны или доставлены в порядке отличном от порядка при отправлении.

UDP обычно не используется, если требуется надёжная передача информации. Использовать UDP имеет смысл там, где потеря части информации не будет критичной для приложения, например, в видеоиграх или потоковой передаче видео. UDP необходим, когда делать повторный запрос сложно или неоправданно по каким-то причинам.

Протоколы транспортного уровня не интерпретируют информацию, полученную с верхнего или нижних уровней, они служат только как канал передачи, но есть исключения. RSVP (Resource Reservation Protocol, протокол резервирования сетевых ресурсов) может использоваться, например, роутерами или сетевыми экранами в целях анализа трафика и принятия решений о его передаче или отклонении в зависимости от содержимого.

4. Прикладной уровень (application layer).

В модели TCP/IP отсутствуют дополнительные промежуточные уровни (представления и сеансовый) в отличие от OSI. Функции форматирования и представления данных делегированы библиотекам и программным интерфейсам приложений (API) — своего рода базам знаний, содержащим сведения о том, как приложения взаимодействуют между собой. Когда службы или приложения обращаются к библиотеке или API, те в ответ предоставляют набор действий, необходимых для выполнения задачи и полную инструкцию, каким образом эти действия нужно выполнять.

Протоколы прикладного уровня действуют для большинства приложений, они предоставляют услуги пользователю или обмениваются данными с «коллегами» с нижних уровней по уже установленным соединениям. Здесь для большинства приложений созданы свои протоколы. Например, браузеры используют HTTP для передачи гипертекста по сети, почтовые клиенты — SMTP для передачи почты, FTP-клиенты — протокол FTP для передачи файлов, службы DHCP — протокол назначения IP-адресов DHCP и так далее.

**Глава 3. Сетевое программирование**

Приложения прикладного уровня, общаются также с предыдущим, транспортным, но они видят его протоколы как «черные ящики». Для приёма-передачи информации они могут работать, например, с TCP или UDP, но понимают только конечный адрес в виде IP и порта, а не принцип их работы.

Как говорилось ранее, IP-адрес присваивается каждому конечному устройству протоколом межсетевого уровня. Но обмен данными происходит не между конечными устройствами, а между приложениями, установленными на них. Чтобы получить доступ к тому или иному сетевому приложению недостаточно только IP-адреса, поэтому для идентификации приложений применяют также порты. Комбинация IP-адреса и порта называется сокетом, или гнездом (socket).

Кроме собственных протоколов, приложения на прикладном уровне зачастую имеют и фиксированный номер порта для обращения к сети. Администрация адресного пространства интернет (IANA), занимающаяся выделением диапазонов IP-адресов, отвечает ещё за назначение сетевым приложениям портов.

Так почтовые приложения, которые общаются по SMTP-протоколу, используют порт 25, по протоколу POP3 — порт 110, браузеры при работе по HTTP — порт 80, FTP-клиенты — порт 21. Порт всегда записывается после IP и отделяется от него двоеточием, выглядит это, например, так: 192.168.1.1:80.

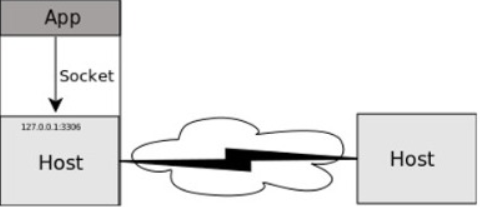
Чтобы не запоминать числовые адреса интернет-серверов была создана DNS — служба доменных имён. DNS всегда слушает на 53 порту и преобразует буквенные имена сетевых доменов в числовые IP-адреса и наоборот. Служба DNS позволяет не запоминать IP — компьютер самостоятельно посылает запрос «какой IP у selectel.ru?» на 53 порт DNS-сервера, полученного от поставщика услуг интернет.

**Глава 4. Сетевое программирование**

Сетевые сокеты — способ взаимодействия любого приложения с системой, на которой оно работает. Сокет — это интерфейс, который, по сути, представляет собой совокупность адреса в сети и используемого порта.

Приложение работает на сервере, подключённом к сети: глобальной или локальной.

Интерфейс между приложением и хост-машиной (Network-Application Interface) определяет, как приложение может использовать сеть.



Приложение клиента запрос (к MySQL, например) на определённый сетевой адрес и порт. В примере это localhost и порт 3306 — сервер в свою отвечает приложению.

К приложению при этом могут обращаться множество клиентов. Запросы и использованием сокета приходят на один и тот же адрес и обрабатываются одним пакетом. Так работает клиент-серверная модель взаимодействия.

В рамках одной системы часто используются Unix-сокеты. Такой способ взаимодействия быстрее.

Чтобы написать приложение, которое могло бы обслуживать множество клиентов (последовательно и параллельно) нужно сокет API.

Socket API — интерфейс, используемый всеми интернет-приложениями.

Socket API при соединении двух приложений может работать с потоками и с датаграммами:

потоки — отправка потока байтов с гарантированной доставкой;

датаграммы — отдельные сообщения без гарантии доставки.

Сетевой сокет — комбинация IP адреса и номера порта, которые представляют собой способ адресации. Адрес и порт обеспечивают нормальное взаимодействие большого количества приложений в рамках одной системы.

Один сокет не может использовать два приложения одновременно или два экземпляра одного приложения.

1. Как называется процесс, когда во время передачи каждый уровень добавляет новые заголовки к PDU, поступающему от вышестоящего уровня?
   1. Цифровая передача данных
   2. Инкапсуляция
   3. Декапсуляция
2. Даёт возможность узнать местоположение ваших устройств в более глобальной сети:
   1. IP-адрес
   2. Протокол
   3. MAC-адрес
3. Сетевой протокол, предназначенный для преобразования IP-адресов в MAC-адреса в компьютерных сетях:
   1. ARP (Address Resolution Protocol)
   2. IP (Internet Protocol)
   3. ISO/OSI (ISO Open System Interconnection)
4. Позволяет пользователю осуществлять подключения к Wi-Fi сетям:
   1. IP адрес
   2. MAC адрес
   3. Хост
   4. Протокол
5. Узел сети, любой компьютер, сервер (физический или виртуальный), подключённый к локальной или глобальной сети:
   1. Хостинг
   2. Сетевой протокол
   3. Сетевой интерфейс
   4. Хост
6. Физическое или виртуальное устройство, предназначенное для передачи данных между программами через компьютерную сеть:
   1. Сетевой интерфейс
   2. Хост
   3. Сетевой протокол
   4. Хостинг
7. Набор правил для регулирования обмена информации между устройствами:
   1. Протокол
   2. MAC-адрес
   3. IP—адрес
   4. Хост
8. Набор правил, определяющих способы управления соединениями на физическом и логическом уровнях:
   1. Хост
   2. Сетевой интерфейс
   3. Сетевой протокол
   4. Хостинги
9. Выберите все существующие уровни классической модели OSI:
   1. Сеансовый
   2. Физический
   3. Многоканальный
   4. Сетевой
   5. Прикладной
10. Как называется физический процесс, в котором данные переносятся в виде сигналов между точками (абонентами)?
    1. Инкапсуляция
    2. Декапсуляция
    3. Цифровая передача данных