TCP

Как всё работает

Эталонная сетевая модель OSI

OSI(Open System Interconnection Reference Model) - эталонная модель взаимодействия открытых систем ISO (открыми системами называются, системы, открытые для связи с другими системами).

Уровни модели OSI

7. Прикладной
6. Уровень представления
5. Сеансовый
4. Транспортный
3. Сетевой
2. Канальный
1. Физический

Физический уровень

Физическиӣ уровень занимается реальноиӣ передачеиӣ необработанных битов по каналу связи.

Канальный уровень (уровень передачи данных)

Канальный отвечает за передачу данных с физического уровня по надежной линии связи, свободной от необнаруженных ошибок, а так же сокрытие реальных ошибок, так чтобы они не были заметны на сетевом уровне.

Сетевой уровень

Сетевой уровень отвечает за управление операциями подсети. В частности он отвечает за определение маршрутов пересылки пакетов (данных) от источника к пункту назначения.

Транспортный уровень

Основной задачей транспортного уровня является принять данные с сеансовго уровня, если есть необходимость разбить их на небольшие части, и передать их сетевому уровню, и сделать это максимально эффективно.

Сеансовый уровень

Сеансовый уровень позволяет пользователям компьютеров устанавливать сеансы связи друг с другом. Передоставляет различные сервисы такие как:

- **управление диалогом** (отслеживание очередности передачи данных)
- **управление маркерами** (предотвращение одновременного выполнения критичной операции несколькими системами)
- **синхронизация** (установка служебных меток внутри длинных сообщений, позволяющих продолжить передачу с того места, на котором она оборвалась, даже после сбоя и восстановления).

Прикладной уровень

Прикладной уровень содержит набор протоколов, которые необходимы пользователям. Примером протокола прикладного уровня есть HTTP.

TCP

Протокол TCP (Transmission Control Protocol) - протокол управления передачей разработан для передачи сквозного байтового потока по ненадежной интерсети.

Модель сервиса TCP

- В основе сервиса лежат сокеты (конечные точки), которые создаются как отправителем, так и получателем.
- У каждого сокета есть адрес, состоящий из IP-адреса и порта.
- Один и тот же сокет может использоваться одновременно для нескольких соедений.
- Номера портов со значениями ниже 1024 зарезервированы стандартными сервисами и доступны только привилегированным пользователям (root)

Модель сервиса TCP

- Все TCP-соединения являются полнодуплексными и двухточечными. Полный дуплекс означает, что трафик может следовать одновременно в противоположные стороны.
- Двухточечное соединение подразумевает, что у него имеются ровно две конечные точки.
- TCP-соединение представляет собой байтовый поток, а не поток сообщений. Границы между сообщениями не сохраняются.

Модель сервиса TCP

- Получив данные от приложения, протокол TCP может послать их сразу или поместить в буфер, чтобы послать большую порцию данных, по своему усмотрению. Для мгновенной отправки данных нужно указать флаг PUSH.

- Отправляющая и принимающая TCP-подсистемы обмениваются данными в виде сегментов. Сегмент TCP состоит из фиксированного 20-байтового заголовка (плюс необязательная часть), за которым могут следовать байты данных.

Модель сервиса TCP

- Основным протоколом, используемым TCP-подсистемами, является протокол скользящего окна с динамическим размером окна. При передаче сегмента отправитель включает таймер. Когда сегмент прибывает в пункт назначения, принимающая TCP- подсистема посылает обратно сегмент (с данными, если есть, что посылать, иначе — без данных) с номером подтверждения, равным порядковому номеру следующего ожидаемого сегмента, и новым размером окна. Если время ожидания подтверждения истекает, отправитель посылает сегмент еще раз.

Заголовок TCP сегмента

Каждый сегмент начинается с 20-байтного заголовка фиксированного формата. И содержит в себе такие поля:

- Порт источника / порт назначения - конечные точки
- Порядковый номер - порядковый номер полученого сообщения
- Номер подтверждения - порядковый номер следующего бита
- Длина заголовка (смещение данных) - размер заголовка
- Флаги - URG, ACK, PSH, RST, SYN, FIN

Заголовок TCP сегмента

- Размер окна - в этом поле содержится число, определяющее в байтах размер данных, которые отправитель может отправить без получения подтверждения.
- Резерв
- Контрольная сумма
- Указатель важности
- Опции

Установка TCP соединения

В протоколе TCP соединения устанавливаются с помощью «тройного рукопожатия»:

- Чтобы установить соединение, одна сторона (например, сервер) пассивно ожидает входящего соединения, выполняя базовые операции LISTEN и ACCEPT, либо указывая конкретный источник, либо не указывая его.

Установка TCP соединения

- Другая сторона (например, клиент) выполняет операцию CONNECT, указывая IP- адрес и порт, с которым она хочет установить соединение, максимальный размер TCP-сегмента и, по желанию, некоторые данные пользователя (например, пароль). Примитив CONNECT посылает TCP-сегмент с установленным битом *SYN* и сброшенным битом *ACK* и ждет ответа.

Установка TCP соединения

- Когда этот сегмент прибывает в пункт назначения, TCP-подсистема проверяет, выполнил ли какой-нибудь процесс операцию LISTEN, указав в качестве параметра тот же порт, который содержится в поле *Порт получателя*. Если такого процесса нет, она отвечает отправкой сегмента с установленным битом *RST* для отказа от соединения. Если какой-либо процесс прослушивает указанный порт, то входящий TCP-сегмент передается этому процессу. Последний может принять соединение или отказаться от него. Если процесс принимает соединение, он отсылает в ответ подтверждение.

Разрыв TCP соединения

Чтобы разорвать соединение, любая из сторон может послать TCP-сегмент с установленным в единицу битом *FIN*, это означает, что у него больше нет данных для передачи. Когда этот TCP-сегмент получает подтверждение, это направление передачи закрывается. Тем не менее данные могут продолжать передаваться неопределенно долго в противоположном направлении. Соединение разрывается, когда оба направления закрываются. Обычно для разрыва соединения требуются четыре TCP-сегмента: по одному с битом *FIN* и по одному с битом *ACK* в каждом направлении.

Скользящее окно TCP

Управление окном в TCP решает проблемы подтверждения корректной доставки сегментов и выделения буферов на приемнике. Например: получателя есть 4096-байтовый буфер. Если отправитель передает 2048-байтовый сегмент, который успешно принимается получателем, то получатель подтверждает его получение. Однако при этом у получателя остается всего лишь 2048 байт свободного буферного пространства (пока приложение не заберет сколько-нибудь данных из буфера), о чем он и сообщает отправителю, указывая соответствующий размер окна (2048) и номер следующего ожидаемого байта.

Скользящее окно TCP

После этого отправитель посылает еще 2048 байт, получение которых подтверждается, но размер окна объявляется равным 0. Отправитель должен прекратить передачу до тех пор, пока получающий хост не освободит место в буфере и не увеличит размер окна.При нулевом размере окна отправитель не может посылать сегменты, за исключением двух случаев. Во-первых, разрешается посылать срочные данные, например, чтобы пользователь мог уничтожить процесс, выполняющийся на удаленной машине. Во-вторых, отправитель может послать 1-байтовый сегмент, прося получателя повторить информацию о размере окна и ожидаемом следующем байте. Такой пакет называется пробным сегментом (window probe).

Источники

Э. Таненбаум, Д. Уэзеролл. Компьютерные сети. 5-е издание. Издательство Питер 2012 год.