**Лекция за 14.03.2012, часть 1**

***Топологии вычислительных сетей (продолжение)***

***Общая шина*** – одна из базовых топологий сети *Ethernet,* которая представляет собой сеть со звездчатой топологией, где в качестве коммутационного устройства выступает сама физическая среда передач, т.е., в некотором смысле, это такой предельный вариант звезды, в которой нет центрального коммутационного оборудования.

Обозначение:  или 

***Особенности сети с общей шиной***

Поскольку используется единая среда передачи, то эта топология обладает низкой конфиденциальностью, более того, по-видимому, самой низкой из всех теоретически возможных. А именно, любая информация, передаваемая абоненту, может быть перехвачена любым другим абонентом сети. В сетях с кольцевой топологией это не так: при передаче информации предшествующему вам узлу, который расположен против направления движения маркера, информационный сигнал (только в данном конкретном случае) проходит через все возможные узлы сети кольца. В сетях же с общей шиной ситуация предельная: вся информация может быть перехвачена чужими абонентами.

Второй особенностью является *механизм разграничения полномочий,* обеспечивающий арбитраж доступа к единому пространству связи. Этот арбитраж может быть реализован различными способами. В *Ethernet*он носит название *механизма контроля коллизий.* ***Коллизия*** *(для сетей физического уровня)* – такое состояние, при котором два передатчика в сети начинают одновременную работу. В каких топологиях возможны *коллизии*? ***Точка-точка***: передатчики могут начать работать одновременно, но при полностью дуплексном соединении это роли не играет, если же одновременный прием и передача информации невозможны, то также требуется некоторый механизм контроля коллизий. ***Звезда***: коллизии эффективно разрешаются на уровне коммутатора; какое-то соединение коммутатор разрешает, а какое-то запрещает (как вариант, могут быть запрещены сразу оба). ***Кольцо***: коллизии принципиально невозможны, поскольку передача разрешена только после прохождения ненагруженного маркера, а через два узла он одновременно пройти не может. ***Общая шина***: когда возникает коллизия, она естественно детектируется каждым из участвующих в ней сетевых адаптеров (ситуация, в которой приемник воспринимает сигнал, исходящий еще откуда-то, помимо передатчика, есть коллизия), после чего оба (или больше) передатчиков, вызвавших ее, отключаются на некоторый случайный промежуток времени, по истечении которого передача возобновляется. Если после возобновления передачи снова наступает состояние коллизии, то передатчики отключаются, на этот раз, на больший момент времени. После трех неудачных попыток передачи данных, модуль физического уровня передает выше по стеку протоколов сообщение о том, что невозможно передать данные. В зависимости от степеней интеллектуальности сетевого адаптера и корректности написания сетевого стека удается, как правило, точно определить причину сбоя: что это именно коллизия, а не физическое повреждения сети (в отношении *Ethernet*:крайне редко встречаются такие способы повреждения сети, которые также выделяются как генерация коллизий).

Коллизии бывают трех *типов*:

* *«много во много»* - много передатчиков начинают одновременную передачу (о них было сказано выше)
* *«много в один»* - два передатчика ведут передачу на один приемник.

Важно уметь их различать.

***Пример***: пусть есть сеть с топологией звезда, и коммутатор имеет пропускную способность для того, чтобы обеспечить соединение всех хостов попарно. Такой коммутатор может эффективно подавить коллизию «*много во много»*, но коллизию *«много в один»* физически не может, т.к. в случае *«много во много»* можно физически развести каналы передачи данных внутри коммутатора, а в случае «*много в один*» один приемник не может работать с двумя передатчиками (физический сигнал на нем будет складываться и приводить непонятно к чему).

* *широковещательные* - сталкиваются два широковещательных запроса или широковещательный запрос с любым другим запросом (***широковещательный запрос*** – один хост передает информацию сразу для всех хостов локальной сети). Именно этот тип коллизий приносит больше всего проблем, т.к. в него вовлечены сразу все хосты сети.

***Другие варианты построения сетей с общей шиной***

* *сети со спектральным разделением* – сети, в которых каждому передатчику в общей полосе пропускания физической среды передачи данных отводится некоторая узкая поднесущая частота, на которую он способен передавать данные. При этом приемники имеют достаточно широкополосные фильтры и могут принимать сразу во всем диапазоне поднесущих. В такой системе не возникает коллизий *«много во много»,* но по-прежнему остаются *«много в один»* и *широковещательные.*

***Историческая справка***

По такой схеме работала локальная вычислительная сеть ***DECnet***, продвигаемая компанией *Digital Equipment Corporation.* В свое время *DEC* являлась конкурентом *IBM*, *Xerox*, но из-за неудачного маркетинга была впоследствии ликвидирована. Ее остатки купил *Compaq*, который затем был куплен *HP*. В сущности, все советские ***СМ ЭВМ*** являются клонами различных компьютеров *DEC*, также как и компьютеры ***ДВК*** (Диалоговый вычислительный комплекс, выпуск с *1982*). *DEC* долгое время являлась поставщиком Министерства обороны США и почему-то начала считать, что если не сможет продать свою продукцию на коммерческом рынке, то Пентагон у них обязательно что-нибудь купит. Но наступил момент, когда Пентагон не стал больше продлевать с ней контракт, и склады *DEC* стали ломиться от производимой продукции, при этом на офисе штаб-квартиры висел гордый лозунг «Хороший товар продает себя сам».

* *сети с разделением по времени –* сети, в которых каждому передатчику выделяется определенный интервал времени (***временной квант сети***), в течение которого он может вести передачу. При этом в продвинутых вариантах таких сетей, если остаются неиспользованные временные кванты, то они могут использоваться теми, кому есть, что передать. Одним из вариантов таких сетей являются современные сотовые сети (например, ***GSM***), некоторая модификация подобной технологии используется в сетях ***Wi-Fi***.

Таким образом, даже на базе сети с ***общей шиной*** можно построить достаточно высокопроизводительные сети, которые будут иметь не очень высокий уровень коллизий. В принципе, в построенной корректно сети по схеме *с разделением по времени* коллизии невозможны. Но основная проблема таких сетей – это необходимость очень точной синхронизации внутренних часов передач, для того, чтобы они могли точно определять момент возможности начала передачи. Первоначальное согласование часов происходит при вызове абонентом сотовой станции (вызов осуществляется на так называемом служебном канале), после чего в процессе продолжения передачи (пока не положена трубка) с интервалом *3-10 сек* проходят специальные ***gsm-фреймы***, которые обеспечивают согласование часов с сотовыми.

***Другие типы топологий***

* ***Облако*** – сети, в которых предоставляются те или иные услуги, при этом абонент не может точно сказать, какой из конкретных хостов сети (или их множество) эти услуги предоставляет.
* ***Плотная сеть*** – топология, в которой каждый хост связан с каждым. По ней строятся различные кластеры (вычислительные, БД). В многоядерных процессорах компании *Intel* ядра также связаны по схеме каждый с каждым, что объясняет отсутствие к настоящему моменту нормального шестиядерного процессора. Для соединения четырех ядер в данной топологии требуется *6 связей* (*4\*3/2*), соответственно для соединения шести ядер необходимо *15 связей* (*6\*5/2*), что означается возрастание *на пол порядка сложности,* что приводит к различным инженерным проблемам по его реализации.
* ***Коммутирующая решетка***: представим себе решетку из горизонтальных и вертикальных полос, одни из которых – входы, другие – выходы, при этом коммутирующее оборудование может соединять любой вход с любым выходом. Это топология также используется в кластерных системах.

***Локальные вычислительные сети. Ethernet.***

***Историческая справка***

***Ethernet*** был изобретен в начале *70*-х, когда еще никто не думал о локальных сетях. Все началось с того, что американскому военному ведомству потребовалась высокоскоростная телефонная связь с Гавайями. С этой целью был запущен первый коммерческий спутник, который должен был обеспечивать многоканальную цифровую телефонную связь. Для этого спутника компания *Xerox* изобрела специальный протокол передачи данных ***Aethernet*** (эфирная сеть). К современному ***Ethernet*** его архитектура не имеет непосредственного отношения, но некоторые его базовые элементы в этой сети присутствовали. На самом деле, ***Aethernet*** представляла собой сеть с ***общей шиной*** *с временным разделением каналов* (такие сети лучше всего подходят для передачи телефонных разговоров и голосовой информации). По разным причинам (в первую очередь, из-за неповоротливости *Xerox*) воплотить этот проект в жизнь в *72*-ом году не удалось: к моменту запуска спутника так и не было поставлено надежно работающее коммутирующее оборудование (как следствие, контракт с Пентагоном был разорван). Но важно то, что *Xerox* к этому моменту провел довольно большую работу и, чтобы не выкидывать ее результаты на ветер, необходимо было найти того, кому можно было бы продать результаты, а также надо было решать вопрос, на какие деньги заканчивать исследование. С этой целью *Xerox* обратился к ведущим американским университетам с предложением о совместном исследовательском проекте. В первую очередь откликнулись Калифорнийский технологический институт в Беркли и университет Пало-Альто. В результате, к концу 70-х годов у *Xerox* уже был прототип недорогой локальной вычислительной сети, основанной на топологии общей шины.

В *80*-м году Пентагон объявил новый конкурс о создании локальной вычислительной сети для Министерства обороны, в котором принял участие *Xerox*, и именно в это время началась работа по стандартизации сетевых протоколов (например, появилась *модель ISO-OSI*). Было заявлено о том, что с этого момента в любых государственных контрактах должны принимать участие те решения, которые соответствуют выработанным американским стандартам (хотя на тот момент никаких стандартов по уровню ***Ethernet*** не существовало). В силу принятого бюрократического решения этим стандартам надо было соответствовать, в результате чего началась болезненная история, продолжавшаяся до середины *90*-х годов. О ней стоит рассказать, поскольку эта драма очень хорошо обрисовывает ситуацию в отрасли на тот момент.

Сначала посмотрим на то, что к этому моменту уже было стандартизовано. Все заинтересованные стороны согласились на стандарте физического уровня – *MAC frame*.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DA** | **SA** | **Length** |  | **CRC32** |
| 6 | 6 | 2 |  | 4 |

Стандарт описывал три обязательных поля – ***Destination Address***, ***Source Address*** (по 6 октетов), ***Length*** (2 октета) и одно необязательное поле – ***CRC Checksum*** длиной 4 октета (обратная контрольная сумма, вычисленная по алгоритму ***CRC32***).

*Как вычисляется контрольная сумма по алгоритму* ***CRC32 (Cyclic Redundancy Code)****?*

Складываются побитно все 32-битные слова, из которых состоит фрейм (сложение по модулю 2^32), потом берется двоичное логическое дополнение (это и есть ***обратное CRC32***). Т.е. при сложении того, что написано в поле контрольной суммы, побитно со всеми 32-битными словами должен получиться 0. ***Прямая CRC32*** – побитовое сложение всех 32-битных слов.

Предполагалось, что сам модуль протокола физического уровня будет определять, нужна ли контрольная сумма или нет (например, для ***Ethernet*** контрольная сумма вычисляется, а для кого-нибудь другого протокола нет). Отсутствие данной суммы не так уж и глупо, потому что, к примеру, протокол может передавать потоковое аудио или видео. А поскольку вычисление контрольной суммы занимает довольно много времени, то для тех случаев, когда целостность данных не так важна, т.е. их можно пропустить с искажением, вычисление контрольной суммы становится избыточным (например, при передаче голоса зашифрованным голосовым кодеком в стандарте ***GSM*** контрольная сумма не вычисляется).

Таким образом, это было единственное, о чем договорились все заинтересованные стороны, и фрейм ***Ethernet*** должен был как-то вкладываться в *MAC* фрейм. Какие компании приняли участие в конкурсе? Кроме *Xerox*, еще *Intel* (известный на тот момент как производитель чипов для контроллеров), *DEC*, *IBM*; *Microsoft* еще не участвовал. Через некоторое время к ним присоединилась молодая, но очень агрессивная компания *Novell,* которая продвигала на рынок операционную систему ***Novell NetWare***.

Первая «схватка» случилась между компанией *IBM* и остальными. Поскольку *IBM* на тот момент очень активно продвигала технологию ***Token Ring***, а в подобного рода комитетах решения принимаются методом консенсуса, то *IBM* препятствовала принятию любых решений путем того, что ее представитель регулярно не являлся на заседание комитета.

Сначала комитет заседал под эгидой *ANSI* (*American National Standards Institute*) в комнате *802.3*, а потом перешел под эгиду *IEEE*  (*Institute of Electrical and Electronics Engineers*). Впоследствии за этим стандартом закрепилось название *802.3*.

После саботажа *IBM* все основные компании (*DEC*, *Intel*, *Xerox*) сделали вид, что комитета как такового не существует и попытались договориться с оставшимися о выработке промышленного стандарта. Но оставшиеся компании проявляли очень много амбиций и очень мало стремления к сотрудничеству. В результате, к концу *80*-х казалось, что работа по общей шине с контролем коллизий зайдет в полный тупик, и тут всех поразила компания *Novell.* Она, не договариваясь ни с кем, выпустила первую версию ***Novell NetWare*** на основе собственной версии ***Ethernet***.

***Фреймы Ethernet***

*Novell* выпустила свою версию ***Ethernet*** с фреймом ***802.3 Novell***.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DA** | **SA** | **Length** | **IPX** | **CRC32** |
| 6 | 6 | 2 | 42(6)-1500 | 4 |

Естественно, к деятельности комитета *802.3* этот тип фрейма не имел *никакого* отношения (название *802.3* использовалось *исключительно* в рекламных целях, поскольку скандалы вокруг этого комитета уже стали «притчей во языцех»). *Novell*’у за это очень сильно досталось: был подан иск в суд, в результате которого *Novell* официально отказался от именования своего фрейма названием ***802.3 Novell***, но в анналах IT-индустрии оно так и осталось.

На самом деле, внутрь novell’овского фрейма напрямую инкапсулировался протокол сетевого уровня ***Novell NetWare*** под названием ***IPX***, иными словами этот фрейм не мог транспортировать ничего, кроме ***IPX*** контента. Общая длина фрейма – ***60-1518*** октетов.

*Чем лимитируется минимальная длина фрейма?* Механизм контроля коллизий работает не мгновенно. Для его устойчивого срабатывания требуется время, эквивалентное времени передачи *40-50* октетов; для гарантии работы этого механизма взяли *60.* В современных версиях принято число *64* (круглое число в двоичной системе). Кроме того, для минимального типа фрейма длина заголовка *18* октетов, соответственно, длина поля полезной нагрузки *42* октета, т.е. примерно *треть* фрейма занимает заголовок. Еще одно ограничение на минимальную длину – размер заголовка ***IPX***.

*Максимальная длина* фрейма ограничивается средним количеством ошибок на переданный фрейм. Чем длиннее фрейм, тем больше вероятность попадания на него ошибки. Естественно, что в случае ошибки данные, которые в нем передаются, оказываются потерянными, и, если передается длинный фрейм, то теряется слишком много данных, что неприемлемо. С другой стороны, чем длиннее фрейм, тем меньше накладные расходы по передачи служебной информации. Т.о., *1500* октет – разумный компромисс между стремлением снизить накладные расходы на передачу служебных данных и оптимизировать значение количества ошибок на переданный фрейм.Непосредственно для ***Ethernet***’а ограничения на максимальную длину связаны еще с тем, что передача очень больших фреймов приводит к тому, что слишком много хостов будут находиться в состоянии ожидания до окончания передачи конкретного фрейма, и сразу после его окончания могут начать свою передачу, что вызовет состояние коллизии.