43 **Протокол передачи файлов FTP. Активный режим**

**FTP –**один из важнейших, базовый протокол передачи файлов

Первый стандарт RFC 114, 1971 г.

Действующий стандарт RFC 959, 1985 г.

Один из базовых протоколов TCP/IP

**Использует транспорт TCP,** использует передачу текстовых команд для организации управления, то есть в одну сторону передаются текстовые команды с параметрами в другую передаются текстовые ответы с статусами.

Поддерживает 2 режима передачи, это базовый или активный и пассивный.

Один из немногих 2-х канальных протоколов TCP/IP:

* Управляющий канал – по нему передаются управляющие сообщеня и получаются коды ответов и этот канал существует на протяжении всего обмена и обычно использует 21 порт (порт передачи команд)
* Канал данных – используется только тогда, когда требуется передать при помощи FTP большие данные (файлы либо каталоги). При этом это одноразовый временный канал, он организуется только на время передачи данных и потом закрывается. В зависимости от режима работы использует либо **20 порт в активном режиме** либо **непривилигорованный в пассивном**

**Активность** в режиме означает то, какая сторона является активной с точки зрения открытия канала данных.

В активном режиме активной стороной является сервер, он сам организует канал данных с клиентом, в пассивном режиме наоборот

**Активный режим**

* Режим «по умолчанию»
* Сервер инициирует соединение данных
* Клиент открывает слушающий порт . **На клиенте FTP организуется слушающий сокет, то есть серверный сокет**
* Номер TCP-порта сервера – 20. Из-за того, что в этом режиме сервер сам открывает соединение клиенту он должен знать IP адрес клиента и этот адрес должен быть доступен из вне и, как следствие, невозможно использовать с технологиями типа NAT, Proxy
* Обычно запрещён в межсетевых экранах



Клиент, когда хочет организовать канал обмена данными он организует канал команд и организует соединение с 21 портом сервера. В момент когда понадобиться передача данных сервер самостоятельно организует связь с клиентским сокетом по которому будут передаваться данные, при этом данные могут передаваться в обе стороны как на прием так и на передачу. Стрелка это только указатель того, кто был инициализатором установления соединения для передачи

44 **Протокол передачи файлов FTP. Пассивный режим**

* Клиент инициирует соединение данных
* Сервер информирует о параметрах канала данных
* Сервер открывает слушающий порт
* Изначально поддерживается не всеми реализациями



Канал связи открывает клиент. Номер порта выбирается непривилигированный, он каждый раз выбирается новый и информация об этом номере передается клиенту. Это может быть не очень удобно, что номер порта будет абстрактный, с точки зрения построения фаервола это тоже не очень просто, так как редко когда серверный порт заранее не определен и данном случае он как раз не определен

**Теперь на уровне описания самого протокола**

**Описание команд FTP**

* USER – имя. Команда аутентификации
* PASS – пароль. Команда аутентификации. Команды передаются в открытом виде и протокол уязъвим.

Команды передаются в открытом виде и протокол уязвим. По поводу USER и PASS, так как FTP используется не только для доступа к частным архивам но и для доступа к публичным архивам, для публичных архивов есть соглашение, что пользователи подключающиеся к публичному архиву в качестве имени пользователя указывают предопределенное слово **анонимус** а вкачестве пароля свою **электронную почту.** Но как таковое это не проверялось. И вроде получается, что в пароле просто достаточно указать строку с собачкой

* REIN – реинициализация
* ABOR – прервать обмены
* QUIT – завершение сесии

**Команды, оперирующие с файловой системой**

* DELE <имя> – удалить файла Если позволяют права. Анонимному пользователю обычно ничего не позволялось удалять
* RNFR <имя> - переименовать из
* RNTO <имя> - переименовать в

Перед RNTO выполняется RNFR

* CWD <путь> - сменить каталог
* CDUP – перейти в родительский каталог
* RMD <имя> - удалить каталог
* MKD <имя> - создать каталог
* PWD – показать текущий каталог

**Команды управления режимом работы сервера**

* PORT a1, a2, a3, a4, p1, p2 - перевод сервера в активный режим

Клиент выдает такую команду. Параметры это 6 байт или октетов, первые 4 байта это IP адрес а последние два это номер порта. Клиент говорит, что он ждет соединение по адресу a1.a2.a3.a4 и по номеру порта вычисляющему по формуле. Передав эти значения клиент инфмирт сервер куда над содиняться и сервер когда требуется создать канал данных соединяться с этим адресом и портом. Так команда переводит в активный режим

Address = ‘a1.a2.a3.a4’

Port = p1\*256+p2 . Клиент выдает такую команду.

* PASV <без параметров> – перевод сервера в пассивный режим. В качестве ответа сервер присылает те же 6 чисел, то есть он ждет от клиента соединение по адресу a1, a2, a3, a4 и номеру порта где p1 и p2 это старший и младший актет.  
  Режимы разделяются на сессии а не сам сервер, то есть сервер формально может работать с клиентами в разных режимах но по сути мы говорим, что ближайшая передача будет такая. Чаще всего все пользуются одним режимом и этот PASV (Этот момент уточнил Стас)

227 a1, a2, a3, a4, p1, p2 (что тут за 227 Ицыксон не говорит, не знаю удалять или нет)

* TYPE {A|E|I} – представление информации. Первая два режима это текстовый режим а последний бинарный режим. FTP реализует небольшую функцию представительского уровня, в TCP/IP нет явно представительского уровня и он реализуется протоколами (так предложение сформулировал Ицыксон). Одна из функций этого уровня это согласование форматов. Так сложилось, что в Windows и Unix используется разное кодирование элементов текстового файла, например в win перевод строки это 2 символа а в unix 1 символом. Соответственно FTP предлагает согласование форматов в случае передачи текстового файла и в режиме TYPE A (стандартный режим передачи текстовых файлов) можно попросить клиента с севрвером согласовали передачув файлов. Полученный с сервера файл в нем происходит перекодировка 2 символов перевода строки в одну сторону и в другую, то есть формально текстовый файл мы передаем как текстовый файл. Если выбрать бинарный режим, то в нем передача происходит без искажений. Актуалосчка 80-90.

◦ A - ASCII ◦ E - EDCDIC ◦ I – Image

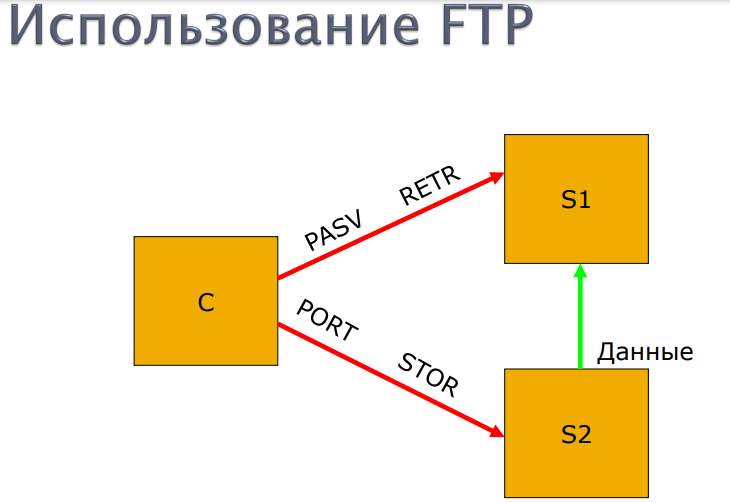
* MODE {S|B|C} – режим передачи данных. Можно попросить сервер передавать в разном режиме. В Compressed режиме файл перед тем как передаваться архивируется в алгоритм похожий на zip.

◦ S – Stream ◦ B – Block ◦ C – Compressed

**Команды, использующие канал данных**

* RETR < имя > - получить файл
* STOR < имя > - записать файл
* LIST [<путь>] – получить список файлов с атрибутами. По стандарту FTP присылается в формате команды ls системы unix
* NLST [<путь>] – получить список имен файлов

Как только клиент выбирает одну из этих команд сразу происходит соединение по второму сокету по кооторому происходит реальная передача данных.

****

Из-за того, что клиент и сервер присылают друг другу не только порт но и IP адресс, это позволяет сделать следующее. У нас есть 2 сервера и мы хотим передавать между ними данные, сам по себе протокол без всяких хитростей заставил бы сначала скачать с S1скачать на клиента а потом на S2, но можно обйтись без клиентского канала и передавать между серверами. Клиент соединяться с серверами, s1 сервер в пассивный, s1 передает клиенту данные где он ожидает соединение а s2 в активный режим а в качестве параметров передаются данные сервера и в итоге канал данных получается между серверами



**Недостатки**

* Не поддерживает шифрования
* Не поддерживает безопасной аутентификации
* Не поддерживает современных средств адресации. Не работает с URL, надо переходить по каталогам
* Сложность работы с защищенными сетями

**Достоинства**

* Эффективность. Использует 2 канала данных
* Гибкость. Для определенной степени, позволяет обеспечить навигацию потоково каталогов