ЛАБОРАТОРНАЯ РАБОТА № 13

**«E-mail услуги и протоколы»**

**Цель работы.** Провести анализ работы E-mail протоколов и предоставляемые услуги с использованием Wireshark.

**Краткие теоретические сведения**

Главной целью протокола SMTP (Simple Mail Transfer Protocol, RFC-821, -822) служит надежная и эффективная доставка электронных почтовых сообщений. SMTP является довольно независимой субсистемой и требует только надежного канала связи. Средой для SMTP может служить отдельная локальная сеть, система сетей или весь Интернет.

SMTP базируется на следующей модели коммуникаций: в ответ на запрос пользователя почтовая программа-отправитель устанавливает двухстороннюю связь с программой-приемником (TCP, порт 25). Получателем может быть оконечный или промежуточный адресат. SMTP-команды генерируются отправителем и посылаются получателю. На каждую команду должен быть отправлен и получен отклик.

Когда канал организован, отправитель посылает команду MAIL, идентифицируя себя. Если получатель готов к приему сообщения, он посылает положительное подтверждение. Далее отправитель посылает команду RCPT, идентифицируя получателя почтового сообщения (таких команд можно выдать несколько, если число получателей более одного). Если получатель может принять сообщение для оконечного адресата, он выдает снова положительное подтверждение. В противном случае он отвергает получение сообщения для данного адресата, но не вообще почтовой посылки.

SMTP-отправитель и SMTP-получатель могут вести диалог с несколькими оконечными пользователями (рисунок 1). Любое почтовое сообщение завершается специальной последовательностью символов. Если получатель успешно завершил прием и обработку почтового сообщения, он посылает положительное подтверждение.

SMTP обеспечивает передачу почтового сообщения непосредственно конечному получателю, когда они соединены непосредственно. В противном случае пересылка может выполняться через одну или более промежуточных "почтовых станций".

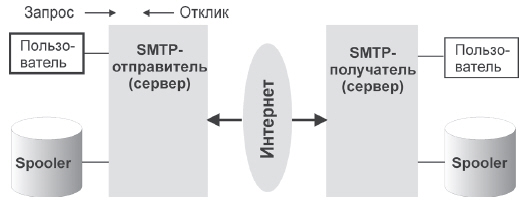


Рисунок 1 **–** Схема взаимодействия различных частей почтовой системы

Для решения поставленной задачи SMTP-сервер должен знать имя конечного получателя и название почтового ящика места назначения. Аргументом команды MAIL является адрес отправителя (обратный адрес). Аргументом команды RCPT служит адрес конечного получателя. Обратный адрес используется для посылки сообщения в случае ошибки.

Все отклики имеют цифровые коды. Команды, отклики и имена ЭВМ не чувствительны к тому, строчные или прописные символы использованы при их написании, но это не всегда справедливо при написании имен и адресов получателя.

Почтовый протокол SMTP работает только с ASCII-символами. Если транспортный канал работает с октетами, 7-битные коды будут дополнены нулевым восьмым битом. Именно здесь коренилась проблема пересылки почтовых сообщений на русском языке (русский алфавит требует 8-битового представления). Проблема усугубляется тем, что для русского алфавита принято 4 кодовых представления, здесь мы впереди планеты всей…

Как уже было сказано, процедура отправки почтового сообщения начинается с посылки команды MAIL, которая имеет формат:

**MAIL <SP> FROM:<reverse-path> <CRLF>**,

где <SP> — пробел, <CRLF> — комбинация кодов возврата каретки и перехода на новую строку, а <reverse-path> — обратный путь (имя почтового ящика отправителя). Именно этот адрес используется, если получатель сообщения воспользуется командой reply.

Эта команда сообщает SMTP-получателю, что стартует новая процедура и следует сбросить в исходное состояние все статусные таблицы, буферы и т.д. Если команда прошла, получатель реагирует откликом: 250 OK.

Аргумент может содержать не только адрес почтового ящика — в общем случае он является списком адресов ЭВМ-серверов, через которые пришло данное сообщение, включая, разумеется, и адрес почтового ящика отправителя. Первым в списке <reverse-path> стоит адрес ЭВМ-отправителя. После прохождения команды MAIL посылается команда RCPT:

RCPT <SP> TO:<forward-path> <CRLF>

Эта команда указывает адрес конечного получателя (<forward-path>). При благополучном прохождении команды получатель посылает кодотклик 250 OK, и запоминает полученный адрес. Если получатель неизвестен, SMTP-сервер пошлет отклик 550 Failure reply. Команда RCPT может повторяться сколько угодно раз, если адресат не один.

Аргумент может содержать не только адрес почтового ящика, но и маршрутный список ЭВМ по дороге к нему. Первым в этом списке должно стоять имя ЭВМ, получившей данную команду. По завершении этого этапа посылается собственно сообщение:

DATA <CRLF>

При правильном приеме этого сообщения SMTP-сервер реагирует посылкой отклика 354 Intermediate reply (промежуточный отклик), и рассматривает все последующие строки в качестве почтового текста. При получении кода конца текста отправляется отклик: 250 OK.

Признаком конца почтового сообщения является точка в самом начале строки, за которой следует <CRLF>.

В некоторых случаях адрес места назначения может содержать ошибку, но получатель знает правильный адрес. Тогда возможны два варианта отклика:

1. 251 User not local; will forward to <forward-path>

Это означает, что получатель берет на себя ответственность за доставку сообщения. Такое случается, когда адресат, например, мигрировал в другую субсеть в пределах зоны действия данного почтового сервера.

2. 551 User not local; please try <forward-path>

Получатель знает правильный адрес и предлагает отправителю переадресовать сообщение по адресу <forward-path>.

SMTP имеет команды для проверки корректности имени адресата (VRFY) и расширения списка адресов (EXPN). Обе команды в качестве аргументов используют строки символов (в некоторых реализациях эти две команды по своей функции идентичны). Для команды VRFY параметром является имя пользователя, а отклик может содержать его полное имя и адрес его почтового ящика.

Реакция на команду VRFY зависит от аргумента. Так если среди клиентов почтового сервера имеется два пользователя с именем Ivanov, откликом на команду "VRFY Ivanov" будет "553 User ambiguous". В общем случае команда VRFY Ivanov может получить в качестве откликов:

250 Vasja Ivanov Ivanov@cl.itep.ru

или:

251 User not local; will forward to Ivanov@cl.itep.ru

или:

550 String does not match anything (данная строка ничему не соответствует).

или:

551 User not local; please try Vasja@ns.itep.ru

или:

VRFY Chtozachertovchina

553 User ambiguous (несуществующее имя)

В случае распечатки списка адресов отклик занимает несколько строк, например:

EXPN Example-People

250-Juri Semenov Semenov@ns.itep.ru

250-Alexey Sher Sher@suncom.itep.ru

250-Andrey Bobyshev Bobyshev@ns.itep.ru

250-Igor Gursky Gursky@ns.itep.ru

В некоторых системах аргументом команды EXPN может быть имя файла, содержащего список почтовых адресов.

Команда Send And MaiL (SAML) предполагает доставку сообщение на экран терминала адресата и занесение в его почтовый ящик. Для открытия и закрытия коммуникационного канала используются команды:

HELO <SP> <domain> <CRLF>, где <domain> — имя запрашивающего домена.

QUIT <CRLF>

Выражение <forward-path> может быть маршрутом, имеющим вид "@ONE,@TWO:VANJA@THREE", где ONE, TWO и THREE — имена ЭВМ. Это подчеркивает различие между адресом и маршрутом. Концептуально элементы из <forward-path> переносятся в <reverse-path> при пересылке сообщений от одного SMTP-сервера к другому.

Если SMTP-сервер обнаружит, что доставка сообщения по адресу невозможна, тогда он формирует сообщение о "недоставленном письме", используя <reverse-path>. Следует также помнить, что и прямой, и обратный адреса-маршруты, вообще говоря, могут не иметь ничего общего с текстом заголовка почтового сообщения.

Если вы или ваша программа не указали обратного адреса, не следует думать, что это помешает работе почтовой программы и она не будет знать, куда посылать отклики. Практически все почтовые программы позволяют произвольно модифицировать поле <reverse-path>. Это может быть удобно, если вы собираетесь в командировку, но эта возможность широко используется и спамерами.

Поле <reverse-path> применяется почтовой программой, когда вы отвечаете на полученное сообщение с помощью утилиты Reply. Таким образом, ваш возмущенный ответ спамеру может прийти, например, к вам самому.

Следует помнить, что обратный IP-адрес (адрес отправителя) указан в каждом пакете, посылаемом адресату!

**Протокол POP3 (Post Office Protocol)**

Протокол обмена почтовой информацией **POP3 (RFC-1939)**предназначен для разбора почты из почтовых ящиков пользователей на их рабочие места при помощи программ-клиентов. Если по протоколу SMTP пользователи отправляют корреспонденцию через Интернет, то по протоколу POP3 пользователи получают корреспонденцию из своих почтовых ящиков на почтовом сервере в локальные файлы.

Итак, РОРЗ (Post Office Protocol version 3). Номер его TCP-порта - 110. Основное отличие РОРЗ от других Интернет-протоколов верхнего уровня заключается в том, что в нем отсутствует широкий спектр кодов ошибок: в ответ на любую команду он посылает строки, начинающиеся с "+ОК" или "-ERR", сигнализирующие соответственно об успешном или неудачном выполнении команды.

Набор основных команд протокола также достаточно прост:

**USER name**.

Это первое, что посылает клиент после того, как он считал строку-приветствие **+ОК РОРЗ served ready**. Аргумент name указывает имя пользователя на данном почтовом сервере, для которого требуется получить доступ к почтовому ящику. Пример:

USER paaa

**PASS password**.

После того как пользователь указал свое имя, он должен указать пароль к своему почтовому ящику. Пример:

PASS doom

**STAT**.

Если пользователь существует и правильно ввел свой пароль, он может посмотреть почту. Команда **STAT** сообщает текущее состояние ящика. Формат ответа - **"+ОК n m"**, где n - количество сообщений, m - количество байт. Пример (в почтовом ящике находятся 11 писем общим объемом 1594 байт):

С: STAT

S: +OK 11 1594

**LIST n.**

Для оценки размера конкретного письма существует команда **LIST.** Формат вывода такой же, как у **STAT**. Пример (третье письмо имеет размер 512 байт):

С: LIST 3

S: +ОК 3 512

**RETR n**.

Для получения письма с сервера используется команда **RETR n**, где n - номер письма. Пример (получаем четвертое письмо):

С: RETR 4

S: +ОК 124 octets

S: Здесь

S: идет

S: текст

S: письма

S:.

**DELE n.**

После того как письма успешно получены и сохранены локально, их можно удалить с почтового сервера. Для этого служит команда **DELE n.**. Пример (удаляем первое письмо):

С: DELE 1

S: +ОК message 1 deleted

**NOOP**

(не использует каких-либо аргументов). При реализации этой команды сервер не делает ничего, лишь посылает положительный отклик.

**RSET**

(не использует каких-либо аргументов)Если какие-либо сообщения помечены как удаленные, сервер POP3 удаляет эту пометку и возвращает положительный отклик. Например:

К: RSET

С: +OK maildrop has 2 messages (320 octets)

**TOP msg n**,

где msg — номер сообщения, а n — число строк (применяется только в режиме TRANSACTION).

При положительном отклике на команду TOP сервер посылает заголовки сообщений и вслед за ними n строк их текста. Если n больше числа строк в сообщении, посылается все сообщение.

**UIDL [msg]**,

где msg — номер сообщения является опционным (Unique-ID Listing).

**APOP name digest**,

где name — идентификатор почтового ящика, а digest — дайджест сообщения — MD5 (RFC-1828). Команда используется только на стадии авторизации.

Обычно любая сессия начинается с обмена USER/PASS. Но так как в некоторых случаях подключения к серверу POP3 может осуществляться достаточно часто, возрастает риск перехвата пароля. Альтернативным методом авторизации является использование команды APOP. Сервер, который поддерживает применение команды APOP, добавляет временную метку в свое стартовое уведомление. Синтаксис временной метки соответствует формату идентификаторов сообщений, описанному в [RFC822], и должен быть уникальным для всех заголовков уведомлений.

**QUIT.**

Для завершения сеанса используется команда **QUIT.**.

**Протокол IMAP (Internet Message Access Protocol)**

IMAP предоставляет пользователю обширные возможности для работы с почтовыми ящиками, находящимися на центральном [сервере](http://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80_%28%D0%B0%D0%BF%D0%BF%D0%B0%D1%80%D0%B0%D1%82%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5%29). Почтовая программа, использующая этот протокол, получает доступ к хранилищу корреспонденции на сервере так, как будто эта корреспонденция расположена на компьютере получателя. Электронными письмами можно манипулировать с компьютера пользователя ([клиента](http://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B8%D0%B5%D0%BD%D1%82_%28%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0%29)) без постоянной пересылки с сервера и обратно файлов с полным содержанием писем.

Любая процедура начинается с команды клиента. Любая команда клиента начинается с префикса-идентификатора (обычно короткая буквенно-цифровая строка, например A0001, A0002 и т.д.), называемого меткой (tag). Для каждой команды клиент генерирует свою метку. Имеется два случая, когда строка, посланная клиентом, не представляет собой законченную команду. В первом — аргумент команды снабжается кодом, определяющим число октетов в строке (см. описание литеральных строк в разделе "Форматы данных"). Во втором — аргументы команды требуют отклика со стороны сервера (см. описание команды authenticate). В обоих вариантах сервер посылает запрос продолжения команды, если он готов. Такой отклик сервера начинается с символа "+".

Данные, передаваемые сервером клиенту, а также статусные отклики, которые не указывают на завершение выполнения команды, имеют префикс "\*" и называются непомеченными откликами.

Имеется три вида отклика завершения сервера: ok (указывает на успешное выполнение), no (отмечает неуспех) или bad (указывает на протокольную ошибку, например, не узнана команда или зафиксирована синтаксическая ошибка).

**Команды клиента**

Ниже описаны команды IMAP 4.1. Команды рассматриваются с учетом состояния, в котором они допустимы.

Следующие команды могут использоваться в любом состоянии: CAPABILITY, NOOP и LOGOUT.

**Команда CAPABILITY**

|  |  |  |
| --- | --- | --- |
| Аргументы: | отсутствуют | |
| Отклики: | необходим немаркированный отклик: CAPABILITY. | |
| Результат: | OK | успешное завершение команды; |
| BAD | команда неизвестна или неверный аргумент. |

Команда CAPABILITY запрашивает перечень возможностей, поддерживаемых сервером.

**Команда NOOP**

|  |  |  |
| --- | --- | --- |
| Аргументы: | отсутствуют. | |
| Отклики: | никакого специального отклика на эту команду не требуется. | |
| Результат: | OK | команда успешно завершена; |
| BAD | команда неизвестна или неверен аргумент. |

Команда NOOP ничего не делает и всегда успешно завершается.

**Команда LOGOUT**

|  |  |  |
| --- | --- | --- |
| Аргументы: | отсутствуют. | |
| Отклики: | необходим немаркированный отклик BYE. | |
| Результат: | OK | прерывание сессии завершено; |
| BAD | неизвестная команда или неверный аргумент. |

Команда LOGOUT информирует сервер о том, что клиент прерывает соединение. Сервер должен послать немаркированный отклик BYE, прежде чем отсылать маркированный отклик OK, после чего завершить разрыв соединения.

**Команды клиента в состоянии без аутентификации**

В состоянии без аутентификации команды AUTHENTICATE или LOGIN организуют аутентификацию и переводят систему в состояние с аутентификацией. Об аутентификации в IMAP можно прочесть в документе RFC-1731. Команда AUTHENTICATE предоставляет общий механизм для целого ряда методов аутентификации, среди которых команда LOGIN используется для традиционного ввода имени и пароля в текстовом виде.

**Команда AUTHENTICATE**

|  |  |  |
| --- | --- | --- |
| Аргументы: | имя механизма аутентификации. | |
| Отклики: | может быть запрошена дополнительная информация. | |
| Результат: | OK | Аутентификация завершена, осуществлен переход в состояние аутентификация выполнена; |
| NO | Ошибка аутентификации: неподдерживаемый механизм аутентификации, параметры аутентификации отвергнуты; |
| BAD | Неизвестная команда или неверный аргумент, механизм аутентификации прерван. |

Команда AUTHENTICATE указывает серверу на механизм аутентификации, как это описано в [IMAP-AUTH]. Если сервер поддерживает запрошенный механизм аутентификации, он выполняет обмен согласно аутентификационному протоколу и идентифицирует клиента. Он может также согласовать опционный механизм защиты для последующих протоколов взаимодействия. Если запрошенный механизм аутентификации не поддерживается, сервер должен отвергнуть команду AUTHENTICATE путем посылки маркированного отклика NO.

**Команда LOGIN**

|  |  |  |
| --- | --- | --- |
| Аргументы: | имя пользователя, пароль. | |
| Отклики: | команда не требует какого-либо специального отклика. | |
| Результат: | OK | login завершено, система в состоянии с аутентификацией; |
| NO | login не прошла: имя пользователя или пароль отвергнуты; |
| BAD | команда неизвестна или неверный аргумент. |

Команда LOGIN идентифицирует клиента серверу и передает пароль пользователя открытым текстом.

**Команды клиента в состоянии "аутентификация осуществлена"**

В состоянии "аутентификация осуществлена" разрешены команды манипуляции почтовыми ящиками как объектами-атомами. Команды SELECT и EXAMINE реализуют выбор почтового ящика и переход в состояние "выбрано".

В добавление к стандартным командам (CAPABILITY, NOOP и LOGOUT), в состоянии "аутентификация осуществлена" допустимы следующие команды: SELECT, EXAMINE, CREATE, DELETE, RENAME, SUBSCRIBE, UNSUBSCRIBE, LIST, LSUB, STATUS и APPEND.

**Команда SELECT**

|  |  |  |
| --- | --- | --- |
| Аргументы: | имя почтового ящика. | |
| Отклики: | Необходимы немаркированные отклики: FLAGS, EXISTS, RECENT; | |
| опционны немаркированные отклики OK: UNSEEN, PERMANENTFLAGS. | |
| Результат: | OK | процедура выбора закончена, система находится в состоянии выбрано; |
| NO | выбор неудачен: нет такого ящика, доступ к почтовому ящику невозможен; |
| BAD | команда неизвестна или неверен аргумент. |

Команда SELECT осуществляет выбор почтового ящика, так, чтобы обеспечить доступ к сообщениям, находящимся там. Прежде чем присылать клиенту OK, сервер должен послать клиенту следующие немаркированные данные:

* FLAGS — флаги, определенные для почтового ящика
* <n> EXISTS — число сообщений в почтовом ящике
* <n> RECENT — число сообщений с набором флагов \Recent
* OK [UIDVALIDITY <n>] — уникальный идентификатор корректности

**Команда EXAMINE**

|  |  |  |
| --- | --- | --- |
| Аргументы: | имя почтового ящика. | |
| Отклики: | Необходимы немаркированные отклики: FLAGS, EXISTS, RECENT; | |
| опционны немаркированные отклики OK: UNSEEN, PERMANENTFLAGS. | |
| Результат: | OK | Просмотр закончен, система в состоянии выбор сделан; |
| NO | Просмотр не прошел, система в состоянии аутентификация выполнена; нет такого почтового ящика; доступ к почтовому ящику невозможен; |
| BAD | Команда неизвестна или неверен аргумент. |

Команда EXAMINE идентична команде SELECT и дает тот же результат, однако, выбранный почтовый ящик идентифицируется "только для чтения". Никакие изменения постоянного состояния почтового ящика в этом случае не разрешены. Текст маркированного отклика OK на команду EXAMINE должен начинаться с кода отклика [READONLY].

**Команда CREATE**

|  |  |  |
| --- | --- | --- |
| Аргументы: | имя почтового ящика. | |
| Отклики: | На эту команду не посылается каких-либо специфических откликов. | |
| Результат: | OK | Команда выполнена; |
| NO | команда не выполнена: почтовый ящик с таким именем не может быть создан; |
| BAD | команда неизвестна или неверен аргумент. |

Команда CREATE создает почтовый ящик с заданным именем. Отклик OK присылается в случае, когда новый почтовый ящик с указанным именем создан. Попытка создания INBOX или почтового ящика с именем существующего почтового ящика является ошибкой. Любая ошибка при попытке создания почтового ящика вызовет маркированный отклик NO.

**Команда DELETE**

|  |  |  |
| --- | --- | --- |
| Аргументы: | имя почтового ящика. | |
| Отклики: | Команда не требует каких-либо откликов. | |
| Результат: | OK | команда завершена; |
| NO | ошибка при выполнении команды: не удается стереть ящик с этим именем; |
| BAD | команда неизвестна или неверен аргумент. |

Команда DELETE навечно удаляет почтовый ящик с указанным именем. При этом присылается маркированный отклик OK только в том случае, когда ящик уничтожен. Ошибкой считается попытка стереть INBOX или ящик с несуществующим именем.

**Команда UNSUBSCRIBE**

|  |  |  |
| --- | --- | --- |
| Аргументы: | имя почтового ящика. | |
| Отклики: | Эта команда не требует каких-либо специфических откликов. | |
| Результат: | OK | ликвидация подписки прошла успешно; |
| NO | ликвидация подписки не прошла: это невозможно для данного имени; |
| BAD | команда неизвестна или неверен аргумент. |

Команда UNSUBSCRIBE удаляет специфицированный почтовый ящик из списка активных или подписных почтовых ящиков данного сервера, как это определяется командой LSUB. Эта команда возвращает маркированный отклик OK только в случае, если ликвидация подписки прошла успешно.

**Команда LIST**

|  |  |  |
| --- | --- | --- |
| Аргументы: | имя, | |
| имя почтового ящика может содержать символы подмены (wildcard). | |
| Отклики: | немаркированные отклики LIST. | |
| Результат: | OK | команда LIST выполнена; |
| NO | команда не прошла: не возможно выполнение LIST для данного образца или имени; |
| BAD | команда неизвестна или неверен аргумент. |

Команда LIST возвращает субнабор имен из полного набора, доступного клиенту. Присылается нуль или более немаркированных откликов LIST, содержащих атрибуты имен, иерархические разделители и имена.

**ВЫПОЛНЕНИЕ**

Доменное имя сервера: mail.ru

Ответ сервера на запрос HELO:

250-smtp8.mail.ru Hello sisko24e37e878 [83.69.130.8]

|  |  |
| --- | --- |
| **E-mail client** | **E-mail server** |
| Mail from: [ramil\_kh@mail.ru](mailto:ramil_kh@mail.ru) | 250 Ok |
| Rcpt to: [mescase@rambler.ru](mailto:mescase@rambler.ru) | 250 Accepted |
| Data | 354 Enter message, ending with “.” On a line by itself |

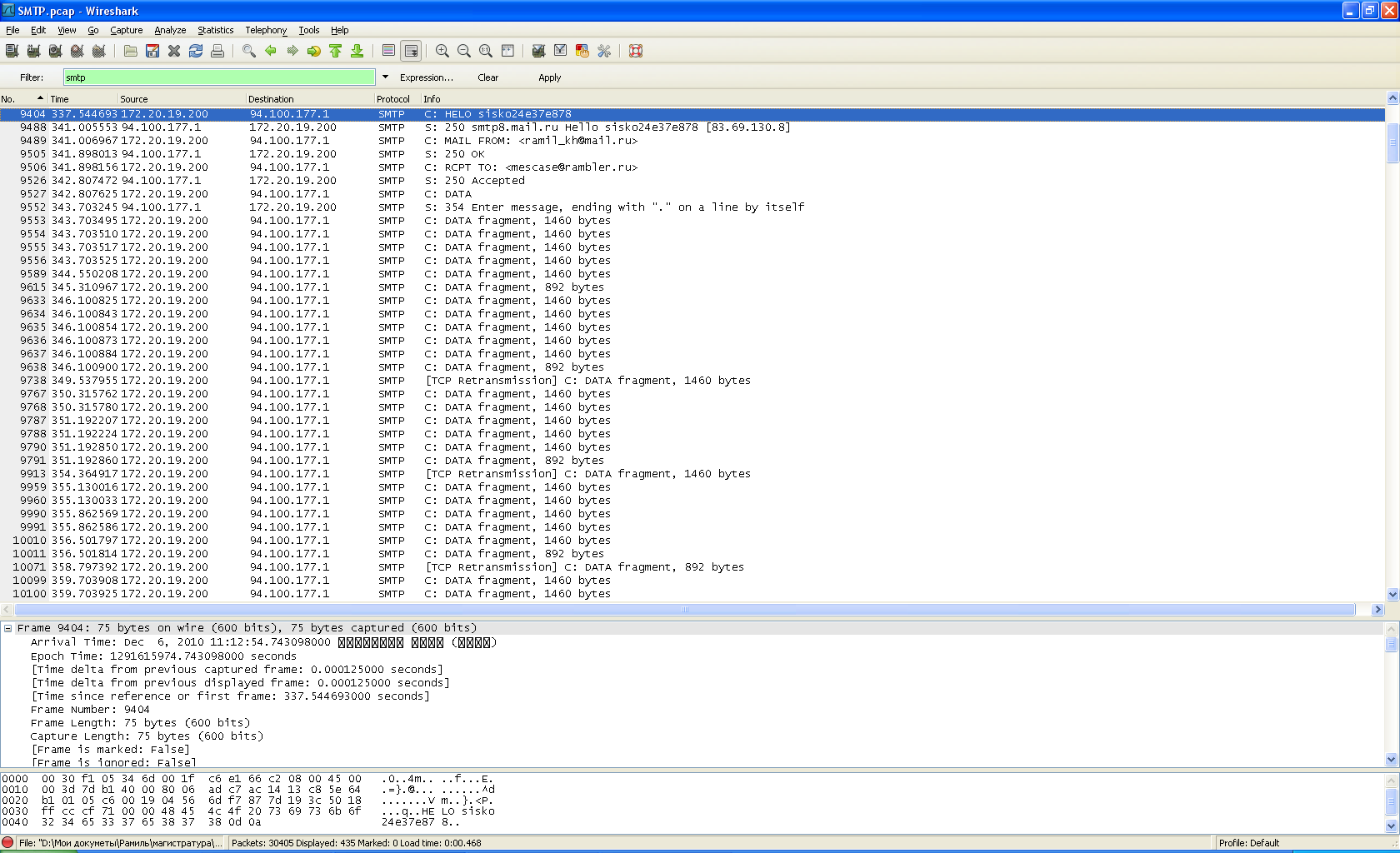


Рисунок 2 – Начало отправки

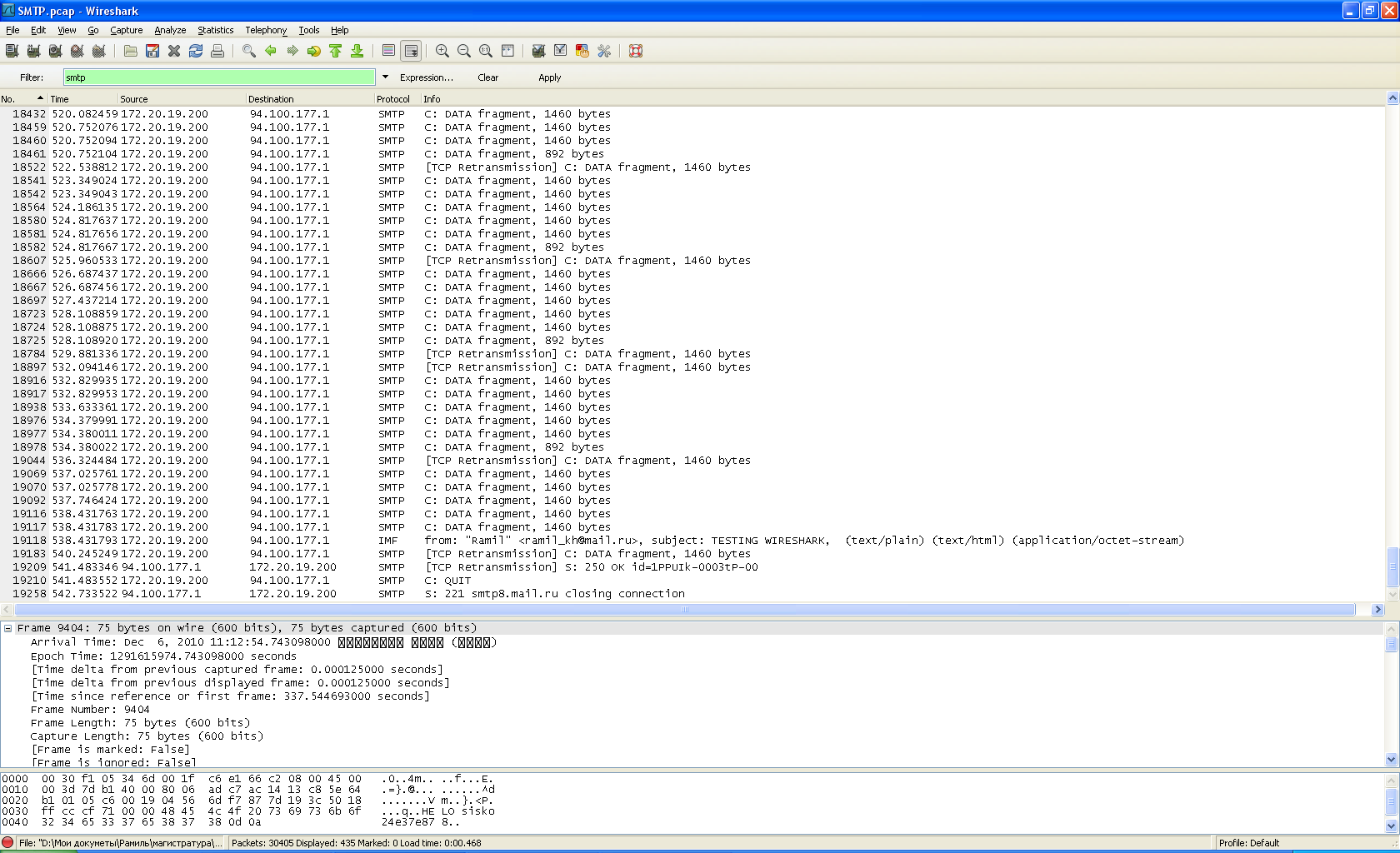


Рисунок 3 – Окончание отправки

В результате отправки сообщения реальный размер файла составил 354 КВ. Размер файла, полученный адресатом сообщения, оказался равным 488 КВ. Таким образом, реальный размер файла составил 73% от размера доставленного сообщения, оставшаяся информация представляет собой дополнительные данные, полученные вследствие выполнения кодирования для надежной доставки данных. На сегодняшний день, большинство из почтовых служб позволяют передавать файлы размером до 20-25 МБ.

**Содержание отчета**

1. Краткие теоретические сведения.
2. Анализ протоколов e-mail и услуг.
3. Протоколы SMTP, POP3, IMAP 4.1.
4. Результаты отправки SMTP – пакетов.
5. Анализ полученных результатов.

**Контрольные вопросы**

1. Назначение и функции программы Wireshark.
2. Назначение и функции протокола SMTP.
3. Назначение и функции протокола POP3.
4. Назначение и функции протокола IMAP.