Разработка SMTP-сервера. Вариант №12

(Студентка группы ИУ7-32М: Мамутова Влада.)

2 февраля 2022 г.

Оглавление

Введение

В данной курсовой работе рассматривается разработка SMTP-сервера. Simple Mail Transfer Protocol (SMTP) - это широко используемый сетевой протокол, предназначенный для передачи электронной почты в сетях TCP/IP. SMTP впервые был описан в RFC 821. Последнее обновление в RFC 5321 включает масштабируемое расширение - ESMTP. Протокол SMTP предназначен для передачи исходящей почты с использованием порта TCP 25.

Целью курсовой работы является реализация протокола прикладоного уровня для получения и доставки электронной почты – Simple Mail Transfer Protocol (SMTP). А именно, части, которая выполняет прием почты и выполняет ее передачу на следующий этап – отправку почты.

Вариант лабораторной работы 12: Используется вызов poll и единственный рабочий поток. Журналирование в отдельном процесс.

Глава 1.

Аналитический раздел

1.1. Описание протокола SMTP

Взаимодействие в рамках SMTP строится по принципу двусторонней связи, которая устанавливается между отправителем и получателем почтового сообщения. При этом отправитель инициирует соединение и посылает запросы на обслуживание, а получатель - отвечает на эти запросы. Фактически, отправитель выступает в роли клиента, а получатель - сервера. Протокол поддреживает маршрутизацию почты, то есть серверу может придти письмо, которое адресовано клиенту на другом сервере. В этом случае серверное программное обеспечение принимает роль клиента и отправлет почту другому серверу.

В спецификации SMTP протокола не определены способы настройки почтовых ящиков для отдельных пользователей, а также не упоминаются какие-либо иные задачи (такие как аутентификация), которые должны быть решены при приеме электронной почты. В этой спецификации просто указано, как должна осуществляться передача электронной почты от отправителя к получателю.

Протокол состоит из текстовых сообщений, которые передают друг другу клиент и сервер при взаимодействии. Каждое сообщение прдеставляет из себя команду с параметрами, которые выполняются сервером. На какждую команду сервер выдает отклик. При организации надежного соединения (например посредством протокола TCP) клиент инициирует почтовую транзакцию, которая состоит из последовательности команд, задающих отправителя и получателя сообщения, а так же передается содержательная часть письма. После чего клиент может завершить сеанс или начать новую почтовую транзакцию для передачи очередного письма.

Электронная почта представлена почтовым клиентом (MUA, mail user agent — пользовательский почтовый агент) для почтового сервера (MSA, mail submission agent — агент отправки электронной почты) с помощью SMTP. Оттуда MSA доставляет почту своим агентам передачи сообщений (MTA, mail transfer agent). Часто эти два агента являются просто различными образцами одного и того же программного обеспечения, запущенного с разными параметрами на одном устройстве. Локальная обработка может быть проведена как на отдельной машине, при этом вовлечённые процессы имеют общий доступ к файлам.

Граничный MTA должен найти целевой хост. Он использует систему доменных имен (DNS) для поиска записей почтового обменника (mail exchanger — MX) домена получателя (часть адреса, находящаяся справа от символа @). Возвращаемая запись почтового MX содержит имя целевого хоста. Затем MTA подключается к серверу обмена в качестве

SMTP-клиента.

Как только цель MX принимает входящее сообщение, она передаёт его агенту доставки почты (mail delivery agent — MDA) для локальной доставки сообщения.

SMTP определяет передачу сообщения, а не его содержание. Таким образом, он задаёт оболочку сообщения и её параметры (такие, как отправитель оболочки), но не заголовок либо тело самого сообщения. RFC 5321 определяет SMTP (оболочку), в то время как RFC 5322 — сообщение (заголовок и тело), официально называемый форматом почтового сообщения (Internet Message Format).

1.2. Объекты электронной почты

- Конверт
  - Адрес отправителя – определяется командой *MAIL FROM*, которая так же начинает почтовую транзакцию.
  - Адрес получателей - с помощью команды *RCPT TO* определяется один получатель и маршрут почты до этого получателя. Данная команда может быть передана несоклько раз для указания списка получателей одного письма.
  - Дополнительные заголовки. Протокол SMTP поддреживает расширения - добавление новых заголовков и параметров к стандартным заголовкам.
- Содержимое – передается после отправки команды *DATA*
  - Заголовок - список полей вида <ключ>:<значение>.
  - Тело сообщения - это непосредственное содержимое письма, которая представляет из себя текстовый набор данных

1.3. Получатель и отправитель

Протокол SMTP работает в 2 стороны. Получателем и отправителем может выступать как почтовая служба на сервере так и клиентское программное обеспечение. В протоколе выделяются следующие понятия:

- Клиент – Отправлющая сторона в текущей почтовой транзакции.
- Сервер – Принимающая сторона в текущей почтовой транзации.
- Агент доставки почты (Mail Transfer Agent, MTA) – Клиент и сервер SMTP обеспечивающее почтовый трансопртный сервис.
- Пользовательский почтовый агент (Mail User Agent, MUA) – Программное обеспечение выступающее в качетсве исходных отправителей и конечных получателей почтовых сообщений

1.4. Команды SMTP

Общение с SMTP сервером ведется при помощи команд. Команды SMTP указывают, какую операцию хочет произвести клиент. Команды состоят из ключевых слов, за которыми следует один или более параметров, Ключевое слово состоит из 4-х символов и разделено от аргумента одним или несколькими пробелами. Каждая команда заканчивается символами CRLF. Обычный ответ SMTP-сервера состоит из номера ответа, за которым через пробел следует дополнительный текст. Номер ответа служит индикатором состояния сервера.

Ниже приведен список основных команд:

- EHLO – данная команда используется для начала диалога клиента с сервером и получения расширений ESMTP, которые доступны для данного сервера (устаревшая - HELO).
- HELO – устаревшая стандартная команда SMTP для начала диалога клиента с сервером (не позволяет получать расширения ESMTP).
- MAIL – определяет отправителя сообщения, используется для ответных сообщений в случае невозможности доставки письма. Для каждого письма команда MAIL должна быть выполнена только один раз.
- RCPT – определяет получателей сообщения. Доставка сообщения возможна тогда, когда указан хотя бы один доступный адрес получателя. Команда RCPT принимает в качестве аргумента только один адрес. Если нужно послать письмо большему числу адресатов, то команду RCPT следует повторять для каждого.
- DATA – определяет начало сообщения. С помощью этой команды серверу передается текст сообщения, состоящий из заголовка и отделенного от него пустой строкой тела сообщения. Команда DATA может быть выполнена только после успешного выполнения хотя бы одной команды RCPT.
- QUIT – остановка сеанса SMTP. Клиент заканчивает диалог с сервером. Сервер посылает подтверждение и закрывает соединение. Получив это подтверждение, клиент тоже прекращает связь.
- HELP – запрашивает список команд. Если команда HELP вызывается без параметров, сервер посылает клиенту список доступных команд. Если в качестве параметра передано название команды, то клиенту посылается описание этой команды.
- VRFY – проверяет имя пользователя системы. Используется для проверки наличия указанного в качестве аргумента почтового ящика. В ответ сервер посылает информацию о владельце ящика или сообщение об ошибке, свидетельствующее о том, что указанный ящик не существует.
- RSET – сброс SMTP-соединения. Данная команда аннулирует все переданные до нее на сервер данные. Процесс передачи сообщения следует начать заново с выполнения команды EHLO (HELO).

1.5. SMTP-сессия

Любая SMTP-сессия состоит из двух ведующих компонентов: команд от клиента и соответствующих им ответов сервера. При открытой сессии обе этих составляющих обмениваются ее параметрами. Подобный обмен может включать как ноль, так и больше SMTP-операций (транзакций).

Классическая SMTP-сессия включает в себя следующие этапы:

1) Инициирование соединения. Клиент создает соединение с сервером. Сервер отвечает клиенту сообщением с кодом отклика 220 в случае готовности для продолжения работы или с кодом отклика 554 в случае отказа в открытии SMTP-сессии.

2) Инициирование работы с клиентом. Клиент передает команду EHLO (HELO). Сервер отправляет сообщение с кодом отклика 250.Если была отправлена команда EHLO, сервер также в сообщении возвращает список расширений, который он поддерживает. Сервер также может вернуть сообщение с кодом отклика 501, если не было передано в аргументах команды имя клиента.

3) SMTP-транзакция (в процессе SMTP-сессии их может быть несколько).

4) Завершение сессии. Если клиент желает завершить работу с сервером, то он посылает команду QUIT, на которую сервер отвечает сообщением с кодом отклика 221 и закрывает соединение. По данной команде можно определить, что клиент также должен освободить ресурсы под выделенное соединение.

Любая SMTP-транзакция представляет собой три последовательные этапа команда/ответ:

1) MAIL from. Определяет обратный адрес. Эта переменная необходима для возвращенных писем. Сервер в случае принятия данных отвечает сообщением с кодом отклика 250 в случае успеха. Но может также ответить с кодом отклика 501 (синтаксическая ошибка).

2) RCPT to. Определяет получателя текущего текстового сообщения. Команда может использоваться несколько раз, в зависимости от количества получателей. Сервер в случае принятия данных отвечает сообщением с кодом отклика 250 в случае успеха. Сообщение с кодом отклика 501 возвращается в случае синтаксической ошибки, а с кодом отклика 503 в случае невозможности принятия данных на данном этапе, с кодом отклика 550 - если пользователь не найден на сервере.

3) DATA. Определяется для последовательной отправки текстового сообщения. Включает в себя непосредственно содержимое письма, в отличие от оболочки. "DATA"несет в себе информацию о заголовке и теле сообщения (они разделяются пустой строкой). Ответ от сервера при передаче происходит в два этапа: на первом от отвечает конкретно на команду "DATA"(уведомление о готовности принять текстовое сообщения), а на втором - о принятии или отклонении всего письма в конце последовательности данных. После выполнения команды DATA сервер возвращает в случае успеха сообщение с кодом 354, что означает, что сервер готов принимать содержимое письма, на все последующие данные сервер не будет отвечать до того

момента, пока не встретит последовательность из <CRLF>.<CRLF>. В случае успеха получения сервером всего текста письма он отвечает сообщением с кодом отклика 250.

Стоит отметить, что SMTP-сервер может разоравать SMTP-сессию в случае истечения времени ожидания. Тогда он вернет сообщение с кодом отклика 421, что свидетельствует о том, что сервер разорвал соединение, и клиент должен освободить ресурсы, выделенные под SMTP-сессию.

Любой SMTP-сервер выполняет несколько функций. Одной из них является проверка правильности настроек и выдача разрешения компьютеру, который пытается отправить электронное письма. Другая функция состоит в отправке исходящих писем на указанный адрес с последующей проверкой доставки.

В качестве задачи, которую необходимо решить в рамках данной курсовой работы, является реализация подключения к серверу множества клиентов и сохранение писем, т.е. реализация SMTP-сервера как части MTA.

Глава 2.

Конструкторский раздел

2.1. Описание конечного автомата

Каждая SMTP-сессия представляет собой совокупность состояний и множество переходов между этими состояниями (в данном случае это определяется с помощью команд при работе с клиентом). Поэтому был построен конечный автомат для протокола SMTP.

Конечный автомат - это математическая абстракция, которая состоит из трех основных элементов: множества внутренних состояний, множества входных сигналов, которые определяют переход из текущего состояния в следующее, множество конечных состояний, при переходе в которые автомат завершает работу.

Конечный автомат, представленный на рисунке 2.1 , описывает состояния, изображенные в виде овалов, и переходы, изображенные на рисунке в виде дуг графа.

Состояния конечного автомата

1) STATE_START – Начальное состояние, в котором находится соединение, когда клиент только подлкючился к серверу

2) STATE_INIT – Состояние инициализированного smtp-сеанса. В него выполняется переход после отправки команды *HELO* или *EHLO*

3) STATE_MAIL_STARTED – инициализация почтовой транзакции, которая происходит, когда клиент отправлет команду *MAIL*

4) STATE_RCPT_RECEIVED – определение списка получателей, с помощью команды *RCPT*.

5) STATE_DATA_RECEIVING – получение сервером тела письма. Данная стадия запускается командой *DATA* и продолжается до тех пор, пока не будет получена последовательность конца данных <*CRLF*>.<*CRLF*>, после этого переходим в состояние инициализированного smtp-сеанса.

6) STATE_CLOSED – завершение сеанса клиента с сервером. Отправлется команда *QUIT* и сервер закрывает соединение.

Автомат описывает только корректную последовательность команд, но если в некотором состоянии будет передана команда, которая не определена конечным автоматом, то состояние не изменится, а сервер сформирует отклик с кодом 503, означающий что клиент ввел неподходяющую команду (некорректная последовательность команд).

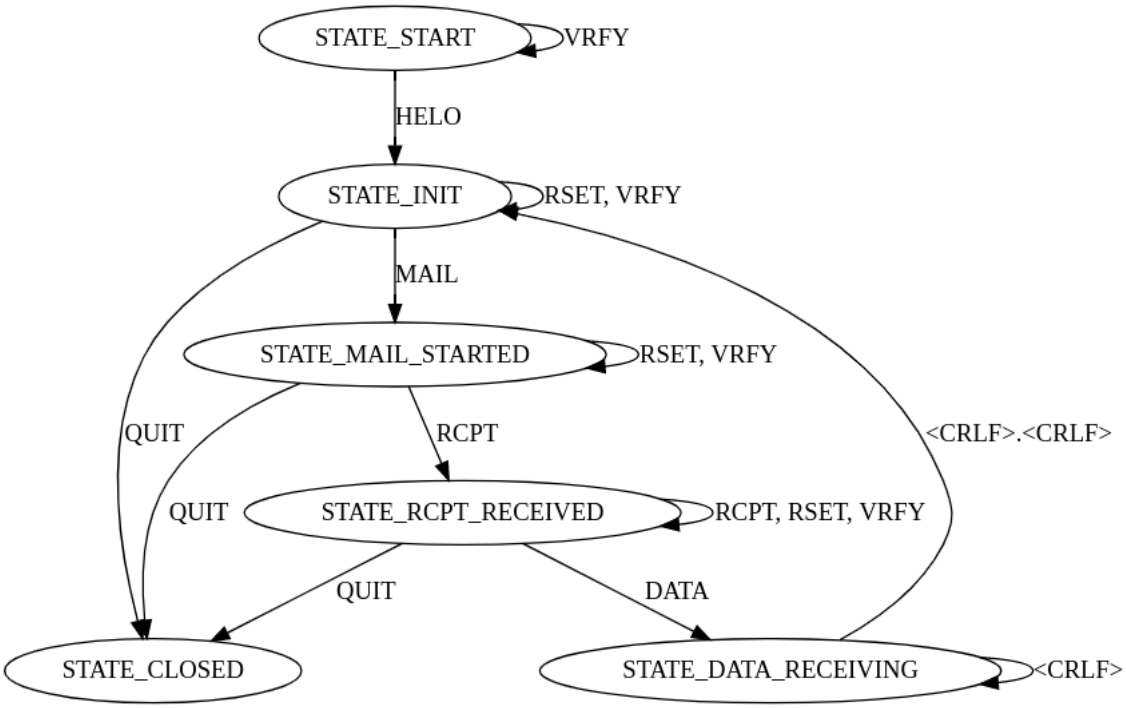

Рис. 2.1. Конечный автомат

2.2. Описание формата хранения писем в файловой системе

Для хранения текстов писем и дополнительной информации, которую принимает сервер во время SMTP-транзакции, используется файловая система.

Maildir - распространенный формат хранения электронной почты, нетребующий монопольного захвата файла для обеспечения целостности почтового ящика при чтении, добавлении или изменении сообщения. Каждое сообщение хранится в отдельном файле с уникальным именем, а каждая папка представляет собой каталог. Вопросами блокировки файлов при добавлении, перемещении и удалении файлов занимается локальная файловая система. Все изменения делаются при помощи атомарных файловых операций, таким образом, монопольный захват файла ни в каком случае не нужен.

В случае стандартного *Maildir* структура каталогов следующая:

```
-home
        -user1
                -Maildir
                        -new
                        -tmp
        -user2
                -Maildir
                        -cur
                        -new
                        -tmp
```

В связи с некоторыми требованиями по использованию стандартного *Maildir*, таким как

создание множества пользователей на устройстве, где запущен SMTP-сервер, было решено модифицировать структуру *Maildir* таким образом, чтобы он содержал письма всех пользователей в едином каталоге. Поскольку также *Maildir* предназначен для доставки только локальной почты, а по условию требуется пересылать также удаленную почту, был добавлена новая директория для удаленной почты, где содержатся папки с почтой, предназначенной для удаленных SMTP-сервером. Структура каталогов модифицированного *Maildir*:

```
- maildir
        -user1
                -cur
                -tmp
                -new
        -user2
                -cur
                -tmp
                -new
        .OTHER_SERVERS
                -server1
                        -error
                        -letter1
                        -letter2
                -server2
                        -error
                        -letter3
                        -letter4
```

- maildir - корневой каталог *Maildir*.
- user1, user2 - имена и каталоги пользователей SMTP-сервера, работающего на данном устройстве.
- cur - папка, содержащая прочитанную почту пользователем.
- tmp - папка, содержащая письма на стадии доставки, запись в файл не является атомарной операцией, поэтому пока производится запись в файл отправка этих данных недопустима.
- new - папка, содержащая файлы писем, которые готовы к отправке.
- .OTHER_SERVERS - каталог, содержащий папки с письмами для удаленных SMTP-серверов.
- server1, server2 - каталоги, содержащие письма для конкретных удаленных сервером с доменным именем по имени данного каталога.
- letter1, letter2, letter3, letter4 - письма удаленной почты готовые для отправки, содержатся сразу в каталогах, предназначенных для удаленных SMTP-серверов.

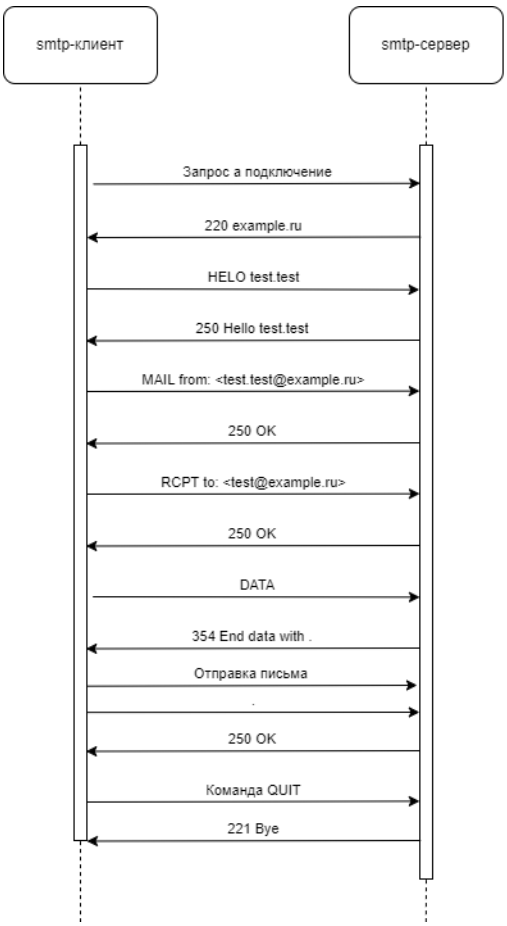

Рис. 2.2. Диграмма последовательности взаимодействия клиента с сервером

SMTP-сервер должен полученные от пользователя письма, если они предназначены удаленному серверу сохранять в директории *.OTHER_SERVERS*, Если письма предназначены для локальных пользователей, то их нужно сразу сохранять в соответствующих директориях.

2.3. Взаимодействие клиента и сервера

В процессе курсовой работы требовалось разработать SMTP-сервер как часть MTA для организации получения письма от клиента и записи в каталог MAILDIR.

На рисунке 2.2 изображена диаграмма последовательности взаимодействия клиента с сервером

Организация удаленной доставки может быть обеспечена следующим образом:

1) Клиент выполняет подключение к серверу.

2) Открытие необходимого соединения SMTP-клиента с SMTP-сервером для последующей сессии.

3) Начало сессии и инициирование общения с ним с помощью команды HELO.

4) Считывание данных письма и последующий анализ дополнительных заголовков необходимых для последующей отправки.

5) Передача адреса почтового ящика отправителя в MAIL. Данный адрес считывается из заголовков письма.

6) Передачи всех возможных адресов получателей, которые были считаны из текста файла, содержащего текст письма.

7) Передача команды DATA.

8) Передача текста письма.

9) Передача флага текста письма и получение сообщения.

10) В случае успеха полученное сообщение сохраняется.

11) Клиент инициирует закрытие сессии.

От SMTP-сервера требуется возможность поддерживания подключения с нескольким клиентами. Требовалось разработать SMTP-сервер таким образом, чтобы он обрабатывал несколько входящих соединений в одном процессе. Требуется воспользоваться вызовом poll.

Системный вызов poll предоставляет пользователю механизм одновременного управления вводом/выводом (мультиплексирования) для набора дескрипторов открытых потоков. Для использования poll нужно инициализировать члены структуры pollfd наблюдаемыми дескрипторами и событиями, а затем вызвать poll().

Особенностями poll является:

– Отсутствие лимита количества наблюдаемых дескрипторов, можно мониторить более 1024 штук.

– Не модифицируется структура pollfd, что даёт возможность её переиспользования между вызовами poll() — нужно лишь обнулить поле revents.

Глава 3.

Технологический раздел

3.1. Описание процесса работы SMTP-сервера

Работа программы начинается с того, что запускается подсистема логгирования, которая представляет собой отдельный процесс, который создаетя с помощью вызова *fork()* и очереди сообщений. В данном процессе выполняется получение сообщения из очереди и сохранение его в файл. Файл, в котором описаны все функции для работы с подсистемой логгирования, представлен в файле *common/logs.c*.

Далее производится загрузка конфигурации SMTP-сервера. Конфигурация представляет из себя параметры Полный путь до *Maildir* настраивается с помощью параметра *application.maildir.path*. Конфигурация позволяет включать режим отладки с помощью параметра *application.debug*, который активирует (или деактивирует) печать логов типа *DEBUG*. Также конфигурация позволяет настраивать доменное имя текущего сервера с помощью параметра *hostname*. Чтение конфигурационного файла реализовано посредством бибилотеки *libconfig*.

Если загрузка конфигурации произошла с ошибкой, то произойдет завершение работы программы путем освобождения ресурсов уже отданных под конфигурацию и остановка подсистемы логгирования.

Для компилирования сервера и отчета используется утилита make. Фаил сборки представлен в листинге ниже

```
CC = gcc
CFLAGS = -Wall -Werror -std=gnu99 -ggdb3
LFLAGS = -lrt -lconfig

INCLUDES = -I include -I ../common/include
SERVER_SRC = ../common/log.c config.c socket_utils.c status.c state.c letter.c client.c client_
OBJ_DIR = obj

OBJECTS = $(patsubst %.o,$(OBJ_DIR)/%.o, $(SERVER_SRC:.c=.o))

SERVER_EXE = server

all: server

server: $(SERVER_SRC)
```

```
	$(CC) $(CFLAGS) $(SERVER_SRC) $(INCLUDES) $(LFLAGS) -o $(SERVER_EXE)

clean:
	rm -rf *.o
```

Глава 4.

Заключение

В ходе выполнения курсовой работы был изучен протокол SMTP и реализовано серверное программное обеспечение выполняюще прием почты по данному протоколу. При реализации сервера использовался системный вызов poll(). Реализованное программное обеспечение было протестировано ручным способом.

Глава 5.

Список литературы

1) rfc2821. Simple Mail Transfer Protocol [Электронный ресурс]. URL: http://rfc.com.ru/rfc2821.htm .

2) Wikipedia.org. SMTP [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/SMTP .

3) Хабр. select / poll / epoll: практическая разница [Электронный ресурс]. URL: https://habr.com/ru/company/infopulse/blog/415259/.

4) Wikipedia.org. Maildir [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/Maildir