Технический Университет Молдовы

Факультет FCIM

# Отчёт

по предмету Сетевое программирование

по Лабораторной работе № 2

# Тема: “SMTP | POP3 || IMAP”

Выполнил: ст.гр. TI-174: Сидлецкий Андрей

Проверил: преп. Булдумак Олег

Кишинёв – 2020

**Назначение протокола SMTP**

SMTP (англ. Simple Mail Transfer Protocol — простой протокол передачи почты) — это широко используемый сетевой протокол, предназначенный для передачи электронной почты в сетях TCP/IP.

**Назначение протокола HTTP**

Hypertext Transfer Protocol – протокол, предназначенный для передач информации в интернете. Протокол HTTP основан на использовании технологии «клиент-сервер»: клиент, отправляющий запрос, является инициатором соединения; сервер, получающий запрос, выполняет его и отправляет клиенту результат.

**Что такое поток?**

Один поток – это одна единица исполнения кода. Каждый поток последовательно выполняет инструкции процесса, которому он принадлежит, параллельно с другими потоками этого процесса.

**Разница между TCP и UDP**

TCP гарантирует доставку пакетов данных в неизменных виде, последовательности и без потерь, UDP ничего не гарантирует.

TCP нумерует пакеты при передаче, а UDP нет

TCP работает в дуплексном режиме, в одном пакете можно отправлять информацию и подтверждать получение предыдущего пакета.

TCP требует заранее установленного соединения, UDP соединения не требует, у него это просто поток данных.

UDP обеспечивает более высокую скорость передачи данных.

TCP надежнее и осуществляет контроль над процессом обмена данными.

UDP предпочтительнее для программ, воспроизводящих потоковое видео, видеофонии и телефонии, сетевых игр.

UPD не содержит функций восстановления данных

**Порядок, в котором байты упорядочены в сети**

Порядок *от старшего к младшему* (Прямой порядок байтов) - способ хранения и обмена данными в сети, при котором сначала передается самый старший байт, а затем остальные в порядке их размещения в блоке данных. Этот порядок является стандартным для протоколов TCP/IP, он используется в заголовках пакетов данных и во многих протоколах более высокого уровня, разработанных для использования поверх TCP/IP. Поэтому порядок байтов от старшего к младшему часто называют «сетевым порядком байтов»

**Четыре уровня стека TCP / IP и семь уровней модели OSI**

Уровни Модели OSI:

* 1 уровень. Физический (physical).
* 2 уровень. Канальный (data link).
* 3 уровень. Сетевой (network).
* 4 уровень. Транспортный (transport).
* 5 уровень. Сеансовый (session).
* 6 уровень. Представительский (presentation).
* 7 уровень. Прикладной (application).

Стек протоколов TCP/IP включает в себя четыре уровня:

* Прикладной уровень (Application Layer),
* Транспортный уровень (Transport Layer),
* Межсетевой уровень (Сетевой уровень) (Internet Layer),
* Канальный уровень (Network Access Layer).

**Почему вы не можете отправлять HTTP-запросы на веб-сайт utm .md с использованием порта 80?**

Потому что utm.md использует протокол HTTPS, который прослушвает порт 443.

**Как создать соединение Socket?**

Socket socket = new Socket(ipAddress, serverPort);

**Как передавать данные в сокете?**

Необходимо получить выходной поток, который дает возможность писать в сокет:

OutputStream sout = socket.getOutputStream();

**Как читать данные из сокета?**

Необходимо получить входной поток, который позволит читать из сокета:

InputStream sin = socket.getInputStream();

**Какие методы у HTTP?**

#### GET

Используется для запроса содержимого указанного ресурса.

#### HEAD

Аналогичен методу GET, за исключением того, что в ответе сервера отсутствует тело.

#### OPTIONS

Используется для определения возможностей веб-сервера или параметров соединения для конкретного ресурса.

#### POST

Применяется для передачи пользовательских данных заданному ресурсу.

#### PUT

Применяется для загрузки содержимого запроса на указанный в запросе URI.

#### PATCH

Аналогично PUT, но применяется только к фрагменту ресурса.

#### DELETE

Удаляет указанный ресурс.

#### TRACE

Возвращает полученный запрос так, что клиент может увидеть, какую информацию промежуточные серверы добавляют или изменяют в запросе.

#### CONNECT

Преобразует соединение запроса в прозрачный TCP/IP-туннель, обычно чтобы содействовать установлению защищённого SSL-соединения через нешифрованный прокси.

**Коды состояния HTTP (200, 301, 302, 401, 404, 405, 500)**

200 - успешный запрос

301 - перемещено навсегда. Запрошенный документ был окончательно перенесен на новый адрес, указанный в поле Location

302 - перемещено временно. Запрошенный документ временно доступен по другому адресу, указанному в заголовке в поле Location

401 - не авторизован. Для доступа к запрашиваемому ресурсу требуется аутентификация.

404 - не найдено. Сервер понял запрос, но не нашёл соответствующего ресурса по указанному URL.

405 - метод не поддерживается.

500 - внутренняя ошибка сервера.

**Что такое безопасный HTTP?**

HyperText Transfer Protocol Secure - это расширение протокола HTTP, поддерживающее шифрование посредством криптографических протоколов SSL и TLS. HTTPS хранит данные в секрете, шифруя их при перемещении между браузером и сервером сайта. Это гарантирует, что любой, кто слушает разговор, не может ничего прочитать.

**Разница между HTTP и HTTPS**

HTTP небезопасен, потому что он транспортирует информацию в виде простого текста, HTTPS обеспечивает конфиденциальность информации путем ее шифрования;

HTTPS не является отдельным протоколом передачи данных, а представляет собой расширение протокола HTTP с шифрованием;

HTTP использует порт 80, HTTPS — порт 443.

**Разница между Socket и WebSocket**

Socket - это программный интерфейс. Это абстрактное понятие, которое, в большинстве случаев, используется для коммуникации программ в сети.

WebSocket - это протокол обмена данными (как, например, http, ftp, ssl и т.д.). Этот протокол идет поверх (передается посредством) протокола TCP.

В своей реализации Websockets используют сокеты.

WebSockets обычно запускаются из браузеров, подключающихся к Application Server по протоколу, похожем на **HTTP,** который работает с **TCP/IP**. Таким образом, они в основном предназначены для веб-приложений, для которых требуется постоянное подключение к серверу.

С другой стороны, простые сокеты более мощные и универсальные. Они превышают **TCP/IP,** но они не ограничены браузерами или протоколом **HTTP**. Они могут использоваться для осуществления любого вида связи.

**Разница между процессом и потоком**

Каждый процесс имеет свой собственный фрагмент памяти, назначенный операционной системой. Таким образом, два или более процессов не имеют возможности обмениваться данными.

В отличие от процессов, потоки совместно используют один и тот же кусок памяти, назначенный их родительскому процессу. Кроме того, потоки обычно легче, чем процесс: они требуют меньше ресурсов и быстрее создаются, поэтому их также называют легковесными процессами.

В одном процессе может быть несколько потоков.

Потоки - это удобный способ заставить программу выполнять несколько операций одновременно.

**Что такое тупик(Deadlock) и когда он появляется?**

Deadlock - ситуация, когда два или более потоков заблокированы навсегда, ожидая друг друга. Напрмер: Поток А использует ресурс А и ему необходим ресурс Б, в то время как Поток Б использует ресурс Б и ему необходим ресурс А. Таким образом 2 потока взаимно блокируют друг друга.

**Для чего нужен метод join () в многопоточном программировании?**

Join() - механизм, позволяющий одному потоку ждать завершения выполнения другого. Например: чтобы главный поток подождал завершения побочного потока myThread, необходимо выполнить инструкцию myThread.join() в главном потоке. Как только поток myThread завершится, метод join() вернет управление, и главный поток сможет продолжить выполнение.