Лабораторная работа № 8

Прием почтовых сообщений по протоколу IMAP

8.1. Цель работы:

Ознакомиться с протоколом управления почтовыми ящиками IMAP для клиента и его реализацией с базовым набором команд.

8.2. Задание на лабораторную работу:

Разработать приложение, позволяющее принимать электронные письма с вложенными файлами, используя протокол IMAP. Необходимо использовать защищённое(шифрованное) соединение. Приложение должно использовать простейший графический интерфейс.

8.3. Методические указания по протоколу IMAP:

IMAP (Internet Message Access Protocol) – протокол прикладного уровня, обеспечивающий возможность клиентам получить доступ и управлять почтовыми ящиками и сообщениями на сервере. Текущая версия протокола IMAP4rev1, описана в RFC 2060.

8.3.1. Транспортный уровень

IMAP использует протокол TCP в качестве транспортного уровня. В стандартной реализации прослушивается порт 143, который принимает незащищённые(нешифрованные) соединения. В связи с повышением уровня безопасности и доступа к личной информации многие почтовые сервера отказались использовать небезопасные(нешифрованные) соединения, поэтому для приёма безопасных(шифрованных) соединений сейчас используется порт 993. Таймаут, при котором сервер разорвёт соединение в связи с бездействием клиента, 30 минут.

8.3.2. Структура сообщений IMAP

Каждое сообщение протокола IMAP имеет стандартизованную структуру. Клиент начинает взаимодействие с сервером, отправляя команду с определённым запросом на сервер. Все команды должны начинаться с некого идентификатора(тэга), имеющего следующую структуру: заглавная буква латинского алфавита и 3 цифры, пример A001 или B992. Данный идентификатор генерируется для каждой команды со стороны клиента, он необходим так как использование IP сети не гарантирует постоянный уровень задержки, поэтому ответ на запрос, который был сделан позднее, может прийти раньше и для точной идентификации этих ответов необходим уникальный тэг. Сервер отвечает на запрос посылая этот же тэг. Кроме запросов и ответов используются отклики, которые несут в себе запрашиваемую информацию.

Пример сообщения:

// Запрос с уникальным идентификатором
A002 Select inBox
// Отклик с информацией о выбранном почтовом ящике
** FLAGS (\Answered \Seen \Draft \Deleted $Forwarded)*
** 25 EXISTS*
** 24 RECENT*
** OK [PERMANENTFLAGS (\Answered \Seen \Draft \Flagged \Deleted $Forwarded *)] Limited*
** OK [UIDNEXT 53] Ok*
** OK [UIDVALIDITY 1418574115] Ok*
// Ответ с этим же идентификатором
A002 OK [READ-WRITE] Select Completed.

8.3.3. Нумерация почтовых сообщений

Каждое сообщение в почтовом ящике имеет порядковый и уникальный номер(UID). Порядковый номер меняется в течении сеанса, например, при удалении первого сообщения в ящике номера всех последующих изменяется на 1 и второе сообщение становиться первым. Уникальный 32-битный номер сообщения используется вместе с уникальным 32-битный номером почтового

ящика *UIDVALIDITY,* вместе два этих идентификатора могут однозначно определить сообщение.

8.3.4. Флаги почтовых сообщений

Каждое сообщение имеет набор атрибутов, которые определяют параметры данного сообщения, эти атрибуты называются флагами. Описание всех флагов приведено в таблице 8.1.

Таблица 8.1

Флаги почтовых сообщений

Флаг	**Описание**
\Seen	Сообщение прочитано
\Answered	Был отправлен ответ на данное сообщение
\Flagged	Сообщение отмечено как важное
\Deleted	Сообщение удалено
\Draft	Сообщение помечено как черновик
\Recent	Сообщение получено

8.3.5. Состояния сервера и команды

Каждая команда требует определённого состояния сервера. Существуют 4 состояния сервера:

1) Аутентификация не произведена – состояние, в котором находиться сервер сразу после успешного соединения.
2) Аутентификация произведена – состояния, поле успешной аутентификации, в котором клиенту доступны манипуляции с почтовыми ящиками
3) Выбран почтовый ящик – состояние, в котором работа происходит только в выбранном почтовом ящике.
4) Завершение соединения – состояние, в котором сервер разрывает соединение.

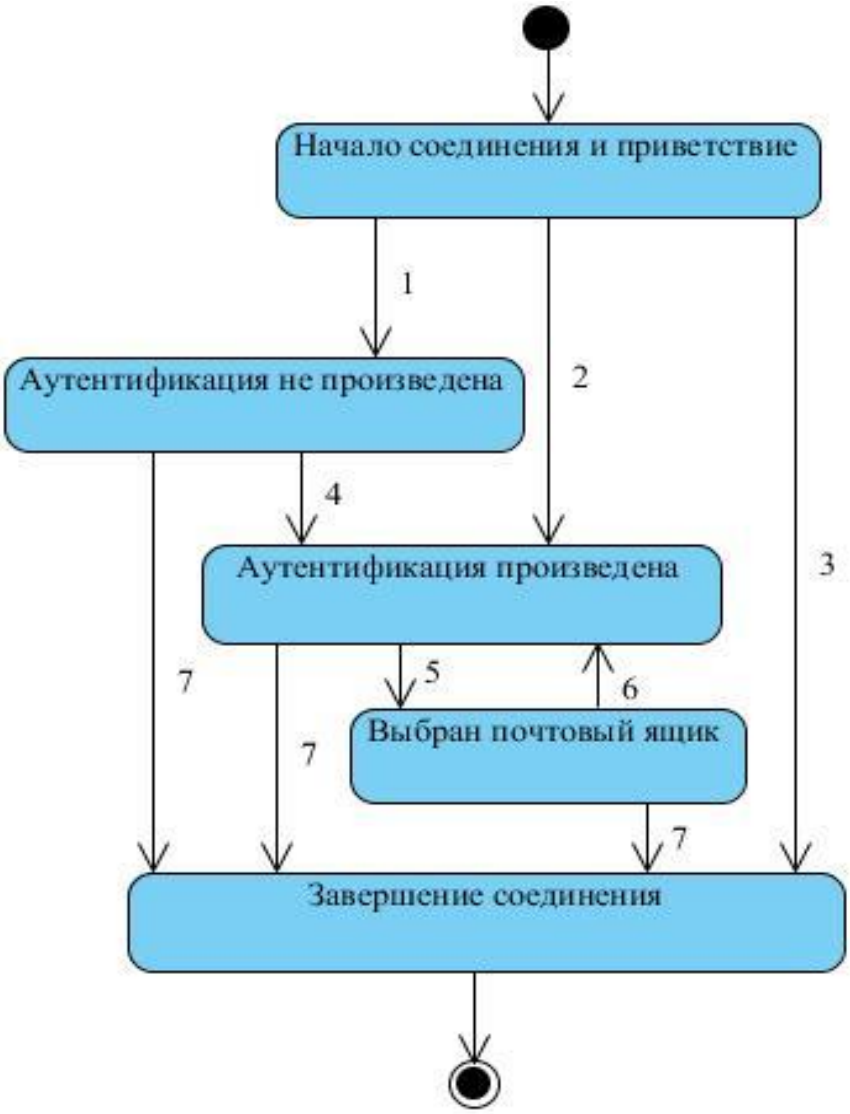

Рисунок 8.1 Диаграмма состояний сервера

На рисунке 8.1 изображена диаграмма состояний сервера, переход между состояниями:

1. Приветствие без предварительной аутентификации
2. Соединение с предварительной аутентификацией
3. Сервер сбросил соединение
4. Аутентификация (LOGIN или AUTHENTICATE)
5. Выбор почтового ящика (SELECT или EXAMINE)
6. Возврат к выбору почтового ящика (CLOSE)
7. Завершение соединения, инициированное клиентом (LOGOUT)

8.3.6. Именование почтовых ящиков и иерархия имён

Каждый почтовый ящик имеет уникальное имя, доступные для именования символы определяются конкретной реализацией сервера. Есть зарезервированное имя INBOX, которое используется в качестве основного почтового ящика. Так как почтовые ящики поддерживают иерархичную структуру, присутствует особый символ разделитель, который также определяется реализацией сервера.

Для лучшего понимания такой структуры приведём пример. Допустим, имеется почтовые ящики newMailBox и Group, при этом в почтовом ящике Group содержаться ещё 2 вложенных почтовых ящика Bonch и Work. Воспользуемся командой List для просмотра имеющихся почтовых ящиков.

A002 List "" ""*
** List (\Unmarked \HasNoChildren \Sent) "|" "&BB4EQgQ,BEAEMAQyBsENQQ9BD0ESwQl-"*
** List (\Unmarked \HasNoChildren) "|" "&BBgEQQRFBD4ENARPBEkEOAQ1-"*
** List (\Unmarked \HasNoChildren \Junk) "|" "&BCEEPwQwBDw-"*
** List (\Unmarked \HasNoChildren \Trash) "|" "&BCMENAQwBDsENQQ9BD0ESwQ1-"*
** List (\Unmarked \HasNoChildren \Drafts) "|" "&BCcENQRABD0EPgQyBDgEOgQ4-"*
** List (\Unmarked \HasChildren) "|" Group*
** List (\Unmarked \HasNoChildren) "|" "Group|Bonch"*
** List (\Unmarked \HasNoChildren) "|" "Group|Work"*
** List (\Marked \NoInferiors) "|" INBOX*
** List (\Unmarked \HasNoChildren) "|" newMailBox*
A002 OK List Completed.

List вернула нам список всех имеющихся почтовых ящиков, в список которых входят имеющиеся у нас почтовые ящики, а также зарезервированные почтовым сервером(стандартные. Символ "|" является разделителем уровня иерархии.

8.3.7. Команды протокола IMAP

В следующих таблицах (8.2 – 8.7) приведены базовые команды протокола IMAP, описанные в RFC 2060.

Таблица 8.2

Список базовых команд протокола IMAP доступных в любом состоянии

Команда	Аргументы	Отклик	Результат	Описание
CAPABILITY	Нет аргументов	CAPABILITY версия протокола, поддерживаемые механизмы аутентификации, дополнительные возможности	OK — успешно завершено	Запрашивает у сервера список поддерживаемых возможностей.
			BAD — сервер не поддерживает запрос или некорректны аргументы	
NOOP	Нет аргументов	Нет отклика	OK — успешно завершено	Не выполняет никаких действий, всегда возвращается успешно. Необходима для сброса таймеров бездействия сервера, т. е. для поддержки постоянной работы TCP соединения.
			BAD — сервер не поддерживает запрос или некорректны аргументы	
LOGOUT	Нет аргументов	BAY	OK — успешно завершено	Информирует сервер о намерении клиента разорвать соединение. Отправляет непомеченный отклик Bay перед отправкой OK, после чего разрывает соединение.
			BAD — сервер не поддерживает запрос или некорректны аргументы	
			NO — ошибка в аргументе или отказано в доступе	
			BAD — сервер не поддерживает запрос или некорректны аргументы	

Таблица 8.3

Список базовых команд протокола IMAP доступных в состоянии “Аутентификация не произведена”

Команда	Аргументы	Отклик	Результат	Описание
AUTHENTICATE	Имя механизма аутентификации	В зависимости от механизма аутентификации	OK — успешно завершено	Информирует сервер о способе аутентификации. Если сервер поддерживает требуемый механизм, то начинается обмен сообщениями по правилам данного механизма. Данный способ аутентификации необходим для незащищённых соединений.
			NO — ошибка при аутентификации: не поддерживаемый механизм или недостаток полномочий	
			BAD — сервер не поддерживает запрос, некорректны аргументы или аутентификация не удалась	
LOGIN	Имя пользователя, пароль	Нет отклика	OK — успешно завершено	Передаёт серверу имя пользователя и пароль в виде обычного текста. Данный способ аутентификации подходит только для защищённых соединений.
			NO — некорректна связка имени пользователя и пароля	
			BAD — сервер не поддерживает запрос или некорректны аргументы	
			NO — ошибка в аргументе или отказано в доступе	
			BAD — сервер не поддерживает запрос или некорректны аргументы	

Таблица 8.4

Список базовых команд протокола IMAP доступных в состоянии “Аутентификация произведена”

Команда	Аргументы	Отклик	Результат	Описание
SELECT	Имя почтового ящика	FLAGS – флаги, допустимые для данного почтового ящика <n> EXISTS, n – общее число сообщений <n> RECENT, n – число сообщений с флагом RECENT Дополнительные отклики: PERMANENTFLAGS - флаги, которые клиент не имеет права изменить UNSEEN - номер первого непрочитанного сообщения	OK — успешно завершено NO — отказано в доступе или недопустимое имя почтового ящика BAD — сервер не поддерживает запрос или некорректны аргументы	Запрашивает у сервера доступ к почтовому ящику для чтения и записи.
EXAMINE	Имя почтового ящика	FLAGS – флаги, допустимые для данного почтового ящика <n> EXISTS, n – общее число сообщений <n> RECENT, n – число сообщений с флагом RECENT Дополнительные отклики: PERMANENTFLAGS - флаги, которые клиент не имеет права изменить UNSEEN - номер первого непрочитанного сообщения	OK — успешно завершено NO — отказано в доступе или недопустимое имя почтового ящика BAD — сервер не поддерживает запрос или некорректны аргументы	Идентична команде SELECT.Запрашивает у сервера доступ к почтовому ящику только для чтения.

CREATE	Имя почтового ящика	Нет отклика	OK — успешно завершено	Создаёт новый почтовый ящик.
			NO —недопустимое имя почтового ящика	
			BAD — сервер не поддерживает запрос или некорректны аргументы	
DELETE	Имя почтового ящика	Нет отклика	OK — успешно завершено	Безвозвратно удаляет почтовый ящик.
			NO —недопустимое имя почтового ящика	
			BAD — сервер не поддерживает запрос или некорректны аргументы	
RENAME	Текущее имя почтового ящика, новое имя почтового ящика	Нет отклика	OK — успешно завершено	Изменяет имя почтового ящика.
			NO —недопустимое имя почтового ящика	
			BAD — сервер не поддерживает запрос или некорректны аргументы	
SUBSCRIBE	Имя почтового ящика	Нет отклика	OK — успешно завершено	Добавляет данный почтовый ящик в число подписанных (активных) почтовых ящиков.
			NO — не удалось подписать данный почтовый ящик	
			BAD — сервер не поддерживает запрос или некорректны аргументы	
UNSUBSCRIBE	Имя почтового ящика	Нет отклика	OK — успешно завершено	Убирает данный почтовый ящик из числа подписанных почтовых ящиков.
			NO — не удалось отписать данный почтовый ящик	
			BAD — сервер не	

			поддерживает запрос или некорректны аргументы	
LIST	Путь к почтовому ящику, имя почтового ящика или шаблон	LIST атрибуты почтового ящика, иерархический разделитель, имя почтового ящика	OK — успешно завершено NO — нет списка с заданным путём и именем BAD — сервер не поддерживает запрос или некорректны аргументы	Запрашивает у сервера все доступные имена почтовых ящиков.
LSUB	Путь к почтовому ящику, имя почтового ящика или шаблон	LIST атрибуты почтового ящика, иерархический разделитель, имя почтового ящика	OK — успешно завершено NO — нет списка с заданным путём и именем BAD — сервер не поддерживает запрос или некорректны аргументы	Запрашивает у сервера все доступные имена почтовых ящиков, которые подписаны (активны).
STATUS	Имя почтового ящика, в скобках необходимые элементы MESSAGES - общее число сообщений RECEN - число сообщений с флагом RECENT UIDNEXT – следующее UID, для нового сообщения. UIDVALIDITY – идентификатор корректности UNSEEN - число сообщений с флагом	STATUS имя почтового ящика, в скобках информация о запрашиваемых элементах	OK — успешно завершено NO — нет данных о состоянии для данного почтового ящика BAD — сервер не поддерживает запрос или некорректны аргументы	Запрашивает у сервера информацию о почтовом ящике.

	SEEN			
APPEND	Имя почтового ящика, в скобках список флагов, в скобках строка даты и времени, сообщение в формате MIME	Нет отклика	OK — успешно завершено NO — ошибка в аргументах или отказано в доступе BAD — сервер не поддерживает запрос или некорректны аргументы	Добавляет сообщение в конец указанного почтового ящика. Данная команда считается небезопасной, лучшее решение – использование специализированного протокола SMTP

Таблица 8.5

Список базовых команд протокола IMAP доступных в состоянии “Выбран почтовый ящик”

Команда	**Аргументы**	**Отклик**	**Результат**	**Описание**
CHECK	Нет аргументов	Нет отклика	OK — успешно завершено BAD — сервер не поддерживает запрос или некорректны аргументы	Организует проверку целостности выбранного почтового ящика. Особенности данной проверки определяет реализация программного обеспечения сервера.
CLOSE	Нет аргументов	Нет отклика	OK — успешно завершено NO — не выбран почтовый ящик BAD — сервер не поддерживает запрос или некорректны аргументы	Закрывает открытый почтовый ящик и удаляет все сообщения с флагом DELETED
EXPUNGE	Нет аргументов	EXPUNGE номер удалённого	OK — успешно завершено	Удаляет все сообщения с

		сообщения	NO — не выбран почтовый ящик или отказано в доступе	флагом DELETED
			BAD — сервер не поддерживает запрос или некорректны аргументы	
SEARCH	Критерии поиска (Описаны в таблице 8.6)	SEARCH номера сообщений, удовлетворяющим критериям поиска	OK — успешно завершено	Запрашивает у сервера список сообщений по критериям поиска.
			NO — ошибка в аргументе	
			BAD — сервер не поддерживает запрос или некорректны аргументы	
FETCH	Номер сообщения или диапазон, элементы данных в сообщении (Описаны в таблице 8.7)	FETCH запрошенные элементы сообщений	OK — успешно завершено	Запрашивает у сервера элементы данных сообщения или списка сообщений
			NO — ошибка в аргументе	
			BAD — сервер не поддерживает запрос или некорректны аргументы	
STORE	Номер сообщения или диапазон, +/-FLAGS добавление или удаление флагов, в скобках список флагов	FETCH обновлённая информация о сообщениях	OK — успешно завершено	Изменение списка флагов у сообщения или списка сообщений
			NO — ошибка в аргументе	
			BAD — сервер не поддерживает запрос или некорректны аргументы	
COPY	Номер сообщения или диапазон, имя почтового ящика	Нет отклика	OK — успешно завершено	Копирует сообщения или диапазон сообщений выбранного почтового ящика в конец указанного почтового ящика.
			NO — ошибка в аргументе или отказано в доступе	
			BAD — сервер не поддерживает запрос или некорректны аргументы	

Таблица 8.6

Основные критерии поиска команды SEARCH протокола IMAP

Критерий	**Описание**
ALL	Все сообщения в выбранном почтовом ящике
BEFORE <date>	Сообщения пришедшие раньше указанной даты
FROM <string>	Сообщения от указанного отправителя
KEYWORD <flag>	Сообщения с указанным флагом
ON <date>	Сообщения с указанной датой
SINCE <date>	Сообщения пришедшие позже указанной даты
SUBJECT <string>	Сообщения с указанной темой
TEXT <string>	Сообщения содержащие указанный текст
TO <string>	Сообщения отправленные указанному получателя
UID <message set>	Сообщения с указанными UID
UNKEYWORD <flag>	Сообщения без указанным флагом

Таблица 8.7

Основные элементы данных команды FETCH протокола IMAP

Элемент данных	**Описание**
ALL	Запрашивает всё сообщение вместе с заголовком
BODY	Запрашивает тело сообщения
BODY[<section>]	Запрашивает определённую часть сообщения. Подробности о структуре сообщения изложены в лабораторной работе посвященной протоколу SMTP
FLAGS	Запрашивает флаги, установленные для сообщения
INTERNALDATE	Запрашивает дату получения сообщения
UID	Запрашивает UID сообщения.

8.3.8. Пример IMAP транзакции

Приветствие сервера после успешного подключения, сервер переходит в состояние “Аутентификация не произведена”

** OK Yandex IMAP4rev1 at imap8j.mail.yandex.net:993 ready to talk with 178.66.216.140:12556, 2015-May-18 19:07:46, k7fPCe6eoa68*

Аутентификация с помощью команды LOGIN, сервер переходит в состояние “Аутентификация произведена”

A001 Login username password
A001 OK Login Completed.

Выбираем почтовый ящик с помощью команды SELECT, сервер переходит в состояние “Выбран почтовый ящик”

A004 Select inBox
** FLAGS (\Answered \Seen \Draft \Deleted $Forwarded)*
** 25 EXISTS*
** 24 RECENT*
** OK [PERMANENTFLAGS (\Answered \Seen \Draft \Flagged \Deleted $Forwarded *)] Limited*
** OK [UIDNEXT 53] Ok*
** OK [UIDVALIDITY 1418574115] Ok*
A004 OK [READ-WRITE] Select Completed.

Запрашиваем у сервера заголовок первого сообщения в почтовом ящике

A005 Fetch 1 body[header]
** 1 FETCH (BODY[HEADER] {329}*
Content-Type: multipart/related; boundary="==========1696383123=="
MIME-Version: 1.0
From: =?utf-8?b?0K/QvdC00LXQutGB?= <hello@yandex.ru>
Subject: =?utf-8?b?0KHQvtCx0LXRgNCIg0Y/RidC40Lo=?=
Message-Id: <20110815165837.A26162B2802A@yaback1.mail.yandex.net>
)
A005 OK Fetch Completed.

Закрываем почтовый ящик, сервер переходит в состояние “Аутентификация произведена”

A006 Close
** OK CLOSE expunged 3 of 3 messages so far*
A006 OK Close Completed.

Инициируем закрытие соединения

A007 Logout
** BYE IMAP4rev1 Server logging out*
A007 OK LOGOUT completed

8.4. Организация защищённого соединения

Для организации защищённых соединений необходимо подключить пакет OpenSSL, который предоставляет возможности шифрования. В файл qMake (name_project.pro) необходимо дописать следующие строки, исправив пути к библиотекам:

```
LIBS += -LC:\Qt\OpenSSL-Win32\ -llibeay32
LIBS += -LC:\Qt\OpenSSL-Win32\ -llibssl32
LIBS += -LC:\Qt\OpenSSL-Win32\ -lssleay32
INCLUDEPATH +=C:\Qt\OpenSSL-Win32\include\
DEPENDPATH +=C:\Qt\OpenSSL-Win32\include\
```

8.5. Организация защищённого соединения средствами библиотеки Qt

Чтобы организовать защищённое соединение средствами библиотеки Qt необходимо использовать класс QSslSocket. Данный класс обеспечивает шифрованное TCP соединение клиента и сервера. Класс унаследован от QTcpSocket.

QSslSocket поддерживает современные протоколы SSL, такие как SSLv3 и TLSv1_0. По умолчанию используется TLSv1_0. TLS и SSL используют ассиметричную криптографию для аутентификации и симметричное шифрование сообщений. Протокол TLS является приемником протокола SSL и является более безопасным, поэтому рекомендовано использовать именно протокол TLS. Шифрование осуществляется поверх существующего сокета, поэтому процедура соединения несколько отличается от QTcpSocket.

Таблица 8.8

Конструктор	QSslSocket:: QSslSocket (QObject * parent = 0)
Описание параметров	parent – объект владелец QSslSocket. При вызове деструктора объекта владельца так же будут уничтожены все дочерние объекты. В случае если владелец не указан, необходимо вручную вызвать деструктор.

Для того, начать процедуру соединения необходимо вызвать функцию connectToHostEncrypted(). При вызове данной функции начинается проверка адреса по всем сетевым интерфейсам, если адрес доступен испускается сигнал hostFound(). При получении сигнала hostFound() создаётся сокет на подходящем сетевом интерфейсе, использующий произвольный свободный порт, и происходит попытка соединения. Если соединение удалось, то посылается сигнал Connected(). После этого начинается процедура установления шифрованного SSL соединения. Если соединение успешно установлено, то испускается сигнал encrypted(), в противном случае испускается сигнал sslError() и соединение обрывается.

Таблица 8.9

Функция	void QSslSocket::connectToHostEncrypted(const QString & hostName, quint16 port, OpenMode mode = ReadWrite, NetworkLayerProtocol protocol = AnyIPProtocol)
Описание функции	Проверяет доступность адреса hostname и порта port, создаёт сокет на подходящем сетевом интерфейсе, используя произвольный незадействованный порт, пытается установить зашифрованное соединение.
Описание параметров	hostName – адрес сервера в сети port – порт сервера openMode – режим, в котором будет происходить работа с сокетом. По умолчанию - режим для чтения и записи (ReadWrite) protocol – протокол сетевого уровня модели OSI. По умолчанию - протокол IPv4 или IPv6(AnyIPProtocol)

После получения сигнала encrypted() работа с QSslSocket ничем не отличается от работы с QTcpSocket, все данные будут шифроваться автоматически.

Если необходимо точно задать версию протокола шифрования используется функция setProtocol().

Таблица 8.10

Функция	void QSslSocket :: setProtocol (QSsl :: SSLProtocol protocol)
Описание функции	Устанавливает протокол шифрования на QSslSocket. Если сокет уже зашифрован, то это не повлияет на текущее соединение.
Описание параметров	protocol – протокол шифрования, по умолчанию TVSv1_0

8.6. Контрольные вопросы:

1) Для чего используется протокол IMAP?
2) Назовите основные отличия протокола IMAP от протокола POP3.
3) Опишите процесс получения писем по протоколу IMAP.
4) Опишите процесс удаления писем.
5) Нарисуйте диаграмму состояния сервера IMAP и поясните переходы.
6) Объясните разницу в способах организации защищённого и незащищённого соединения средствами выбранной вами библиотеки.