# Техническое задание

**«Система передачи данных на удаленный сервер или в «облачное» хранилище данных».**

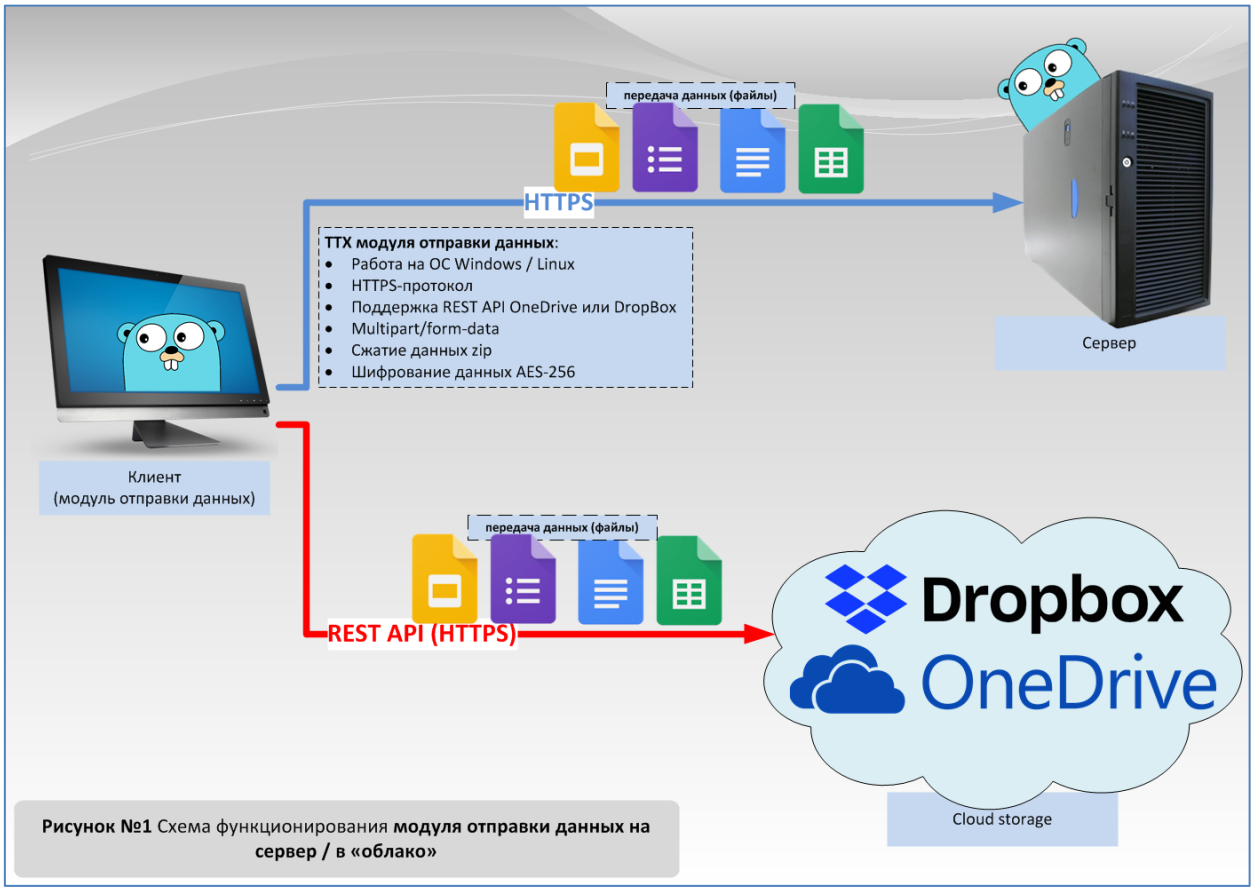


**Рисунок №1** Схема функционирования **модуля отправки данных на сервер / в «облако»**

## Описание системы

Сервис передачи данных предназначен для быстрой, эффективной и безопасной передачи файлов на удаленный сервер или «облачное» хранилище данных. Система позволяет осуществлять автоматическую передачу файлов, как на собственный сервер, так и в «облачное» хранилище данных «OneDrive» или «DropBox» через REST API.

Система состоит из двух модулей: клиентской программной части (далее клиент) и серверной части (далее сервер).

## Общие требования

Система представляет собой HTTP API со следующими требованиями к бизнес-логике:

**сервер:**

- прием данных (файлов) от клиента;
- учет и ведение списка переданных файлов на сервере;
- регистрация, аутентификация и авторизация пользователя на сервере приема данных (при использовании браузера);
- страница с формой отправки файлов через браузер;

- возможность шифрования и дешифрования данных.

**клиент:**

- отправка данных (файлов) на сервер;
- отправка нескольких файлов на сервер за один раз или каждого файла по отдельности;
- отправка файлов через REST API в «облачное» хранилище данных «OneDrive» или «DropBox»;
- получение списка ранее переданных файлов с сервера приема данных;
- получение списка ранее переданных файлов с «облачного» хранилища;
- удаление файла с сервера приема данных;
- удаление файла с «облачного» хранилища данных «OneDrive» или «DropBox»;
- возможность шифрования и дешифрования данных;
- возможность сжатия данных при отправке.

**Абстрактная схема взаимодействия с системой**

Ниже представлена абстрактная бизнес-логика взаимодействия пользователя с системой:

- пользователь запускает клиент, который выполняет сканирование директории, указанной в параметре, фиксирует находящиеся в директории файлы и выполняет их отправку на сервер. Клиент должен уметь работать в двух режимах: отправка данных на удаленный сервер и отправка данных в «облако» «DropBox» или «OneDrive» (в зависимости от параметров запуска);
- алгоритм отправки файлов на сервер должен предоставлять возможность выбора способа отправки: отправка нескольких файлов за раз или каждого файла по отдельности (в зависимости от параметров запуска);
- пользователь может осуществлять загрузку файлов на сервер через браузер (без использования клиентского модуля). Для этого пользователь регистрируется на сервере. На сервере реализован некий «frontend» для возможной загрузки файлов через html-страницу с формой в браузере;
- пользователь может осуществлять загрузку данных в «облачное» хранилище данных путем запуска клиента с параметром токена доступа;
- кроме отправки данных на сервер или в «облако» пользователь может получать листинг ранее загруженных файлов и выполнять запросы на удаление файлов (с сервера или из «облака»);
- сервер осуществляет прием данных от клиента, их регистрацию в базе данных и запись данных (файлы) в каталог, указанный в параметре. В случае применения «облачных» технологий в качестве сервера используется «облако». В этом случае, клиент осуществляет свое

взаимодействие с «облачным» хранилищем через REST API (по правилам облачной платформы «OneDrive» или «DropBox»).

**Общие ограничения и требования**

**клиент:**

- **a.** клиент должен функционировать на платформах ОС Windows / Linux;
- **b.** протокол клиент-серверного взаимодействия – https (443 порт);
- **c.** модуль отправки циклически осуществляет перебор файлов в указанном каталоге и готовит их к отправке на сервер;
- d. при обходе рабочего каталога подкаталоги, символические ссылки, а также файлы размером больше допустимого значения должны игнорироваться;
- **e.** имена файлов при отправке должны иметь следующий вид: «`unixtime_<filename>`»;
- **f.** размер передаваемых данных (размер файла) указывается в виде параметра и не может превышать 16 МБ;
- **g.** отправка нескольких файлов на сервер (одна сессия) должна осуществляться путем отправки POST-запроса с методом кодирования тела запроса – `multipart/form-data`;
- **h.** в случае работы с данными через «облако» (отправка, список, удаление файлов) взаимодействие осуществляется через REST API облачной платформы «DropBox или «OneDrive» (функции получения/обновления «токена доступа», если это необходимо, должны быть реализованы в клиенте);
- **i.** все найденные файлы перед отправкой на сервер/облако могут сжиматься алгоритмом сжатия данных (например zip) (в зависимости от параметров запуска);
- **j.** данные внутри запроса шифруются алгоритмом AES-256 (в зависимости от параметров запуска);
- k. ключ шифрования указывается в виде параметра при запуске клиента;
- **l.** при выходе на связь клиент получает шифрованные данные (AES-256) от сервера (responce), дешифрует их на том же ключе;
- **m.** при запуске клиента можно передавать конфигурационные данные через параметры запуска или через переменные окружения.
- **n.** состав и название параметров (флагов) на усмотрение студента.

**сервер:**

- **a.** сервер должен функционировать на платформе ОС Linux;
- **b.** настройка сервера может осуществляется через протокол SSH (в случае если запуск сервера осуществляется на удаленном хостинге);
- **c.** сервер должен уметь дешифровывать переданные клиентом шифрованные данные (алгоритм AES-256);
- **d.** ключ шифрования указывается в виде параметра при запуске сервера;

**e.** сервер осуществляет запись данных в файлы по указанному в параметре пути;
**f.** сервер регистрирует (заносятся в БД «PostgreSql») такие данные как: время загрузки файла; IP; User-Agent; имя полученного файла; путь к файлу; размер файла (состав полей, структура таблиц, типы и формат хранения на усмотрение студента);
**g.** формат и алгоритм проверки аутентификации и авторизации пользователя на сервере (при использовании браузера) на усмотрение студента;
**h.** если загрузка файлов осуществляется через браузер без использования клиента, то сервер предоставляет зарегистрированному пользователю html-страницу с формой загрузки файлов (подобие «frontend»);
**i.** при запуске сервера можно передавать конфигурационные данные через параметры запуска или через переменные окружения.
**j.** состав и название параметров (флагов) на усмотрение студента.

**Сводное HTTP API**

Сервис передачи данных должен предоставлять следующие HTTP-хендлеры:

- `POST …/multiupload` – отправка файлов (multipart/form-data);
- `POST …/upload` – отправка файла;
- `GET …/list` – получение списка ранее отправленных файлов;
- `GET …/index` – страница формы отправки файла через браузер;
- `DELETE …/{<filename>}` – удаление файла;
- `POST …/register` – регистрация пользователя;
- `POST …/login` – аутентификация пользователя.

**REST API которое предоставляет «облачное» хранилище данных «DropBox»**

- `POST https://content.dropboxapi.com/2/files/upload` - отправка файла в «облако»;
- `POST https://api.dropboxapi.com/2/files/list_folder` - получение списка файлов в директории;
- `POST https://api.dropboxapi.com/2/files/delete_v2` - удаление файла или директории в «облаке»;

Для успешной авторизации в «облаке» необходим параметр в заголовке запроса с актуальным значением токена доступа:
«`Authorization: Bearer <get access token>`».

**REST API которое предоставляет «облачное» хранилище данных «OneDrive»**

- `PUT https://graph.microsoft.com/v1.0/me/drive/root:/<foldername>/<filename>:/content` - отправка файла в «облако»;
- `GET https://graph.microsoft.com/v1.0/me/drive/root:/children` - получение списка файлов в директории;
- `DELETE https:// graph.microsoft.com /v1.0/me/drive/root:/<foldername>/<filename>` - удаление файла или директории в «облаке»;

Для успешной авторизации в «облаке» необходим параметр в заголовке запроса с актуальным значением токена доступа:
«`Authorization: bearer <get access token>`».

*Внимание! Токены доступа к «OneDrive» или «Dropbox» недолговечны, срок их действия истекает через короткий период времени. Точное время истечения срока действия токена возвращается конечной точкой токена. Поэтому, необходимо будет реализовать функцию обновления токена доступа (refreshAccessToken()).*

**Список возможных кодов ошибок (status codes)**

- **200** – OK;
- **201** – created;
- **202** – accepted;
- **204** – no content;
- **400** – bad input parameter, bad request;
- **401** – invalid access token («DropBox»), wrong authentication credentials;
- **404** – not found;
- **409** – endpoint-specific error, eg. login already busy;
- **500** – internal server error.