

Руководство разработчика "Рутокен VPN Клиент Community Edition"

ВЕРСИЯ 1.0

Оглавление

| | |
|---|----|
| Основные сведения | 2 |
| Термины, определения и сокращения | 3 |
| Ссылки на репозитории | 4 |
| Быстрый старт | 4 |
| Руководство пользователя..... | 4 |
| Интернационализация | 4 |
| Зависимости проекта и их взаимодействие..... | 5 |
| Описание компонентов системы | 5 |
| Описание взаимодействия компонентов системы | 5 |
| Стек технологий | 6 |
| Структура папок проекта..... | 7 |
| Описание классов..... | 9 |
| Main..... | 9 |
| Свойства..... | 9 |
| Методы | 9 |
| ConfigurationProvider | 9 |
| Свойства..... | 9 |
| Методы | 9 |
| ConnectionProvider..... | 10 |
| Свойства..... | 10 |
| Методы | 10 |
| LogProvider | 11 |
| Свойства..... | 11 |
| Методы | 11 |
| OpenVpnProvider..... | 11 |
| Свойства..... | 11 |
| Методы | 11 |
| PkcsIdsProvider | 11 |
| Свойства..... | 11 |
| Методы | 11 |
| Сигналы Main Process electron'а..... | 12 |

Основные сведения

Рутокен VPN Клиент Community Edition (далее "Рутокен VPN Клиент CE" или Клиент) представляет собой клиентское решение, предназначенное для подключения к серверу Рутокен VPN CE. Продукт базируется на программном продукте OpenVPN, который реализует технологию VPN для создания зашифрованных каналов.

Использование Рутокен VPN Клиент CE (совместно с Рутокен VPN CE) позволяет достичь следующих целей:

- обеспечить защищенное подключение к сети компании;
- внедрить двухфакторную аутентификацию, где в качестве фактора владения используются криптографические токены и смарт-карты Рутокен ЭЦП;

Помимо Рутокен VPN Клиент CE также существует версия продукта - Рутокен VPN Клиент, позволяющая взаимодействовать с коммерческой версией сервера Рутокен VPN. Рутокен VPN Клиент CE отличается от версии Рутокен VPN Клиент тем, что:

- не предоставляется возможность использования шифрования с использованием криптографических алгоритмов ГОСТ Р 34.10-2012 и ГОСТ Р 34.12-2015;
- нет возможности запускать клиент без наличия прав администратора системы;
- не предоставляется возможность создания установочного файла;

Подробное описание располагается в [README.md](#) файле.

Термины, определения и сокращения

CA – Certification Authority (Центр сертификации)

CRL – Certificate Revocation List (Список отозванных сертификатов)

IPC – Inter Process Communication

OpenVPN – свободная реализация технологии виртуальной частной сети (VPN) с открытым исходным кодом для создания зашифрованных каналов типа точка-точка или сервер-клиенты между компьютерами

OpenSSL – криптографическая библиотека с открытым исходным кодом, широко известна из-за расширения SSL/TLS, используемого в веб-протоколе HTTPS

SSL – криптографический протокол, который подразумевает более безопасную связь. Он использует асимметричную криптографию для аутентификации ключей обмена, симметричное шифрование для сохранения конфиденциальности, коды аутентификации сообщений для целостности сообщений

TLS – протокол защиты транспортного уровня, как и его предшественник SSL – криптографические протоколы, обеспечивающие защищенную передачу данных между узлами в сети Интернет

VPN – Virtual Private Network (виртуальная частная сеть)

ПК – персональный компьютер

ПО – программное обеспечение

Рутокен VPN Клиент CE – Рутокен VPN Клиент Community Edition

Ссылки на репозитории

Репозиторий проекта – <https://github.com/AktivCo/Rutoken-VPN-Community-Edition-Client>

Быстрый старт

Руководство для быстрого запуска Рутокен VPN Клиент CE на машине разработчика описано в файле [INSTALL.md](#)

Руководство пользователя

С руководством пользователя можно ознакомиться по [ссылке](#).

Интернационализация

Рутокен VPN Клиент CE не поддерживает интернационализацию интерфейса.

Зависимости проекта и их взаимодействие

Описание компонентов системы

Рутокен VPN Клиент CE использует следующие основные компоненты:

- [OpenVPN](#)
- [Electron](#)

Взаимодействие компонентов представлено на рис. 1 ниже.

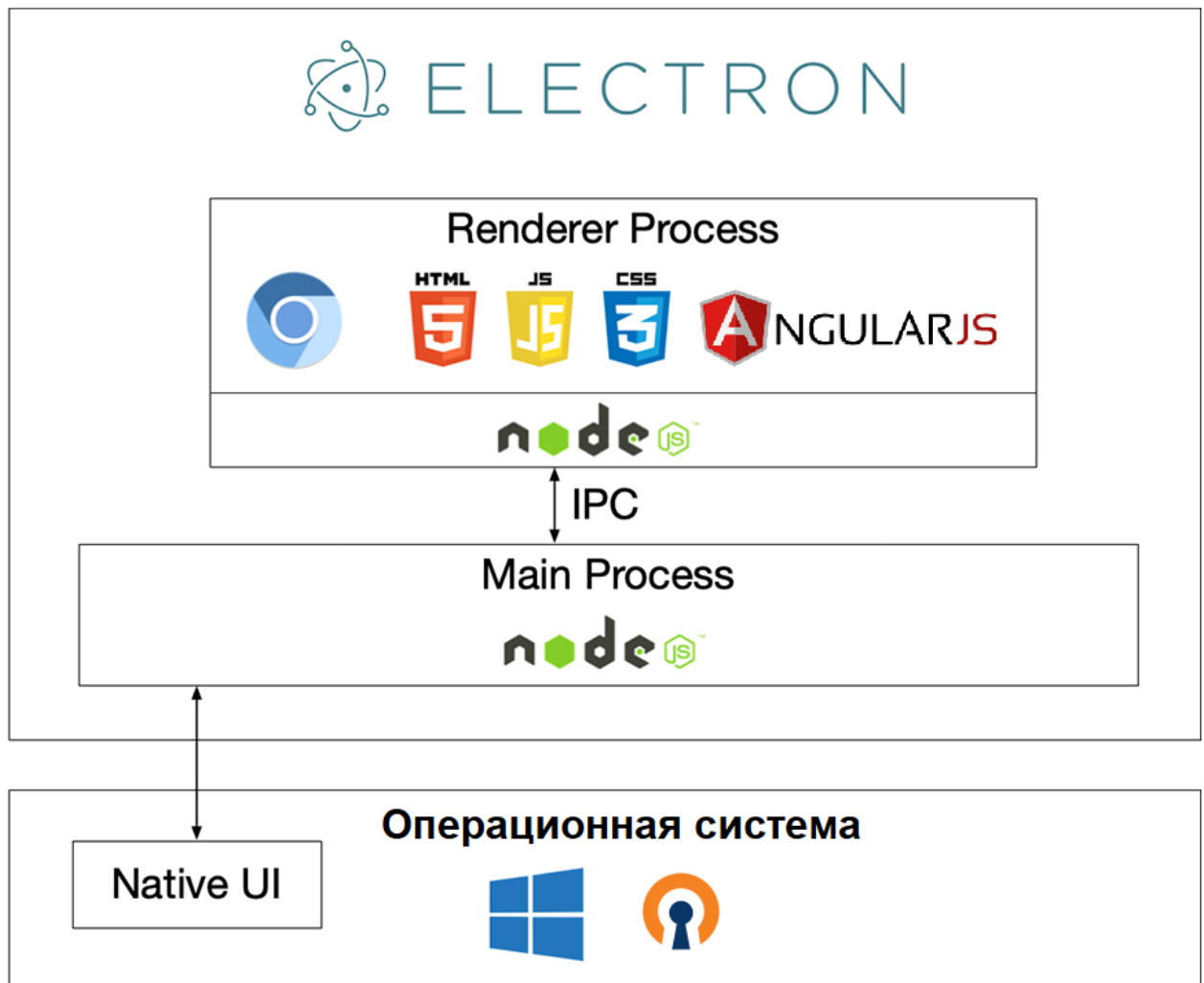


Рисунок 1 Взаимодействие компонентов системы

Описание взаимодействия компонентов системы

Пользователь взаимодействует с системой посредством UI. Обработка событий интерфейса осуществляется Renderer Process'ом. Renderer Process посылает сигнал в Main Process, в котором реализована логика приложения. Main Process в свою очередь работает с OpenVPN, расположенным в ОС Windows.

Стек технологий

Backend:

- Node.js
- Electron

Frontend:

- JavaScript
- AngularJs
- Webpack
- Node.js
- npm







Связанные продукты:

- [Рутокен VPN Сервер](#)
- [Рутокен VPN Сервер CE](#)
- [OpenVPN](#)
- [OpenSSL](#)
- [Токены Рутокен](#)
- [Рутокен плагин](#)

Структура папок проекта

Файловая структура проекта:

- 📁 app/
 - 📁 executables/ - директория с библиотеками, необходимыми для работы проекта
 - 📁 front/ - директория с frontend'ом проекта
 - 📁 css/ - директория со стилями для frontend'a
 - 📁 fonts/ - директория со шрифтами
 - 📁 img/ - директория с картинками
 - 📁 templates/ - директория с html-страницами
 - 📁 utils/ - директория с файлами, содержащими дополнительные функции
 - 📄 main.js – входной файл frontend'a с подключенным angularjs
 - 📁 node_modules/ - директория с модулями для сборки frontend'a проекта
 - 📁 providers/ - директория с сервисами для backend'a
 - 📁 tray/ – директория с картинками и иконками, используемых в собранном проекте
 - 📁 updateScripts/ – директория со скриптами для обновления проекта
 - 📄 app.bundle.js – файл с собранным проектом в .js-расширении
 - 📄 index.html – основной файл с html-разметкой проекта
 - 📄 log.txt – файл с логами проекта
 - 📄 main.js – основной и входной файл проекта
- 📁 assets/ - директория с ресурсами необходимыми для сборки проекта, сгруппированные по ОС
 - 📁 mac/ - для ОС macOS
 - 📁 win/ - для Windows
- 📁 node_modules/ - директория с модулями для сборки проекта.
- 📁 win/ - директория, в которой будет расположен собранный проект
- 📄 .eslintrc.js – конфигурационные файлы для eslint.
- 📄 config.json – конфигурационный файл для сборки electron.

-  gulpfile.js – файл со скриптами для автоматического выполнения.
-  INSTALL.md – инструкция разворачивания окружения разработчика и запуска проекта.
-  LICENSE – файл лицензии проекта.
-  package.json – конфигурационный файл менеджера пакетов .
-  README.md
-  webpack.config.js – конфигурационный файл для сборки проекта.

Описание классов

Main

Свойства

forceQuit: boolean – переменная отвечающая за принудительное закрытие приложения

mainWindow: object – объект главного окна приложения

application: object – объект всего приложения

BrowserWindow: object – объект браузерного окна главного окна приложения

_connectionProvider – ссылка на сервис с обработчиками сигналов

Методы

onWindowAllClosed() – обработчик события закрытия всех окон приложения

onClose(evt) – обработчик события evt закрытия главного окна приложения

onReady() – обработчик события готовности приложения к запуску и создания главного окна приложения

quit() – обработчик события завершения приложения

beforeQuit() – обработчик события происходящего в момент до завершения приложения

main(app, browserWindow) – функция работы с главным процессом приложения, которая присваивает всем событиям реализованные функции-обработчики

ConfigurationProvider

Свойства

_resources: object – объект со ссылками на все ресурсы, требующихся для работы приложения

Методы

getConfig(path) – метод получения конфигурации по пути path

saveConfig(path, data) – метод сохранения конфигурации data по пути path

deleteConfig(path) – метод удаления конфигурации по пути path

checkConfigValidity(config, properties) – метод валидации конфигурации config по свойствам properties

register(updateServer, productGuid, version) – метод регистрации версии version приложения с productGuid по адресу updateServer

checkUpdate(updateServer, productGuid, clientGuid) – метод проверки существования новой версии приложения

`downloadUpdate(downloadInfo)` – метод скачивания обновления
`getVpnServerConfig()` – метод получения конфигурации vpn сервера
`getSettingsConfig()` – метод получения конфигурации настроек приложения
`getVpnServerAddress()` – метод получения адреса vpn сервера
`saveVpnServerConfig(data)` – метод сохранения vpn конфигурации `data`
`saveSettingsConfig(data)` – метод сохранения конфигурации приложения `data`
`deleteVpnServerConfig()` – метод удаления конфигурации vpn сервера

ConnectionProvider

Свойства

`_configProvider`: object – ссылка на сервис для работы с конфигурацией
`_logProvider`: object – ссылка на сервис для работы с логированием
`_openVpnProvider`: object – ссылка на сервис для работы с OpenVPN
`_pkcsIdsProvider`: object – ссылка на сервис для работы с pkcs
`_resources`: object – ссылка на ресурсы
`_status`: `ConnectionStatus` – текущий статус приложения
`_tail`: object – объект для работы с Tail
`_pkcsIds`: Array – массив с полученными сертификатами
`_vpnProc`: object – объект процесса работы с OpenVPN
`_vpnProcService`: boolean – переменная показывающая, есть ли подключение по OpenVPN
`_updateModel`: object – объект с моделью для обновления приложения
`_settingsModel`: object – объект с моделью настроек приложения
`_contents`: object – объект с содержимым главного окна приложения
`_applcon`: object – объект с иконкой приложения
`_quit`: function – функция завершения приложения

Методы

`start()` – метод с инициализацией обработчиков событий electron'а
`getPkcsIds()` – метод с получением
`sendVpnStatus()` – метод отправки статуса приложения в UI
`getStatusInfo()` – метод получения текущего статуса
`startVpn(id)` – метод инициализации соединения по vpn

`processOutputDataHandler(output)` – метод обработки данных получаемых при работе через vpn

`killer()` – метод завершения процесса `openvpn`

LogProvider

Свойства

`_path: string` – путь к файлу с логами приложения

Методы

`init()` – метод инициализации соединения с файлом логирования

`log(text)` – метод записи текста `text` в файл логов по пути `_path`

OpenVpnProvider

Свойства

`_resources: object` – ссылка на ресурсы приложения

Методы

`CreateVpnConfig(vpnServerConfigurationModel, pkcsId)` – метод создания конфигурации `vpnServerConfigurationModel` для сертификата `pkcsId`

PkcsIdsProvider

Свойства

`_resources: object` – ссылка на ресурсы приложения

Методы

`formatDate(date)` – метод получения даты `date` в виде строки

`formatDateString(dateString)` – метод получения объекта с информацией о дате истечения сертификата по строке в виде `date` `dateString`

`getPkcsString()` – метод получения массива сертификатов

`getPkcsIdsModels(lines)` – метод получения объектов с информацией о сертификатах по данным, полученным из команды `OpenVPN`

`GetPkcsIdsModels()` – метод получения информации о сертификатах, возвращающих объект типа `Promise`

Сигналы Main Process electron'а

Коммуникация между frontend'ом и backend'ом в приложениях, созданных с помощью electron, происходит посредством отправления и принятия IPC-сигналов, описанных в файле `app/providers/connectionProvider.js`.

`connection` – событие инициализации работы и получения сертификатов

`startVpn` – событие соединения по vpn

`getCurrentConnectionStatus` – событие получения текущего статуса соединения

`stopVpn` – событие завершения соединения по vpn

`sendPin` – событие отправки PIN-кода

`startUpdate` – событие начала обновления

`setStatusToInit` – событие перехода приложения к начальному статусу

`saveConfig` – событие сохранения конфигурационного файла

`saveSettings` – событие сохранения настроек приложения

`deleteConfig` – событие удаления конфигурационного файла