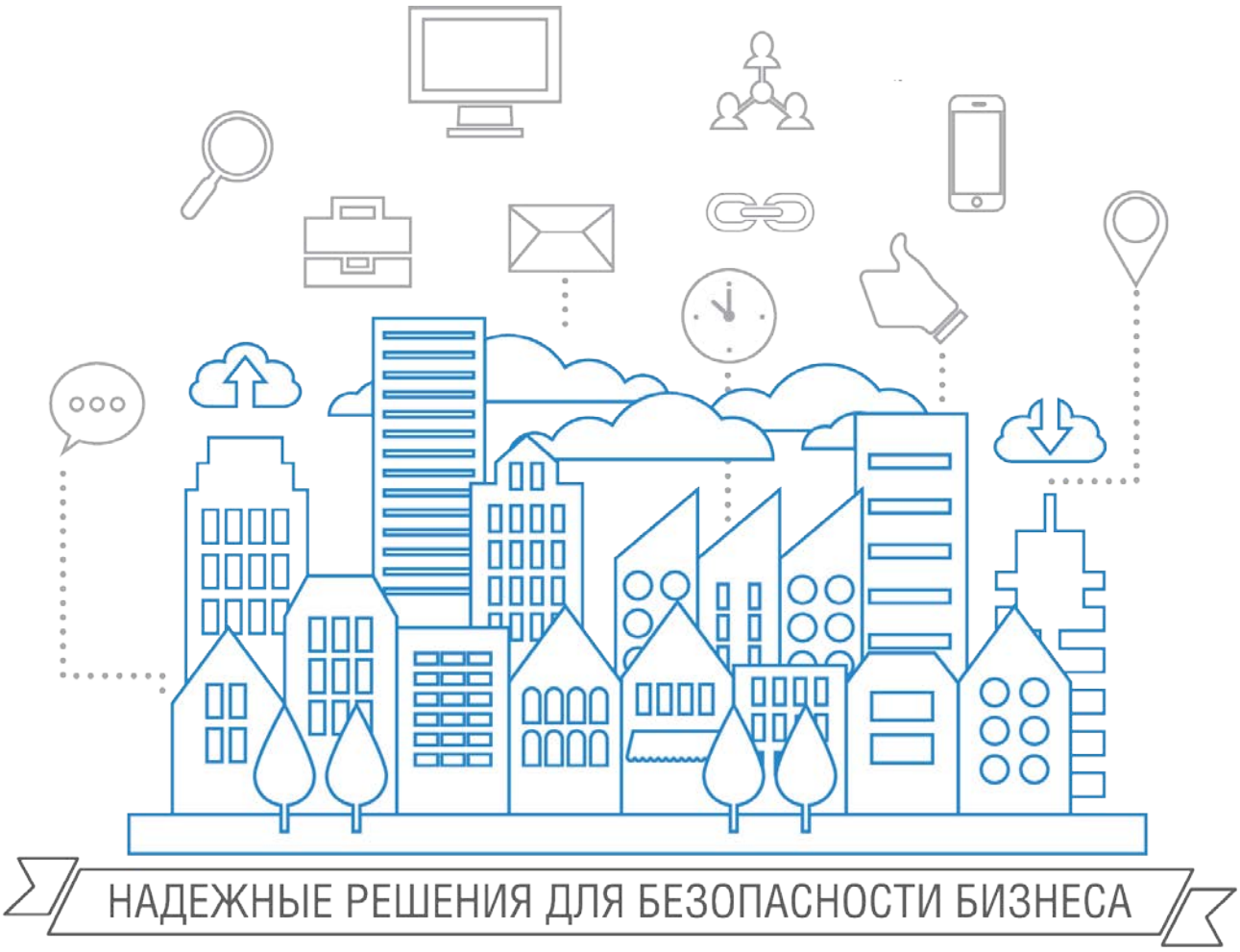
ГАЗИНФОРМСЕРВИС

198096, г. Санкт-Петербург, ул. Кронштадтская, д.10, лит. А, тел.: (812) 677-20-50, факс: (812) 677-20-51
Почтовый адрес: 198096, г. Санкт-Петербург, а/я 59, e-mail: resp@gaz-is.ru, www.gaz-is.ru
р/с 40702810800000001703 Ф-л Банка ГПБ (АО) в г. Санкт-Петербурге БИК 044030827,
к/с 30101810200000000827, ОКПО 72410666, ОГРН 1047833006099, ИНН/КПП 7838017968/783450001

Библиотека «GiSCryptoCpp» (Demo-версия)

Описание интерфейсов



Санкт-Петербург, 2017

Аннотация

В документе приводится описание программных интерфейсов библиотеки «GisCryptoCpp» (Demo-версия), реализованных на языке C++, с помощью которых вызываются основные функции криптографической платформы «Litoria Crypto Platform».

В разделе «Общие сведения» описано назначение библиотеки «GisCryptoCpp» и указаны функции, которые вошли в Demo-версию библиотеки.

В разделе «Описание программных интерфейсов» приведена классификация программных интерфейсов, а также указаны заголовки объявления, входные параметры, возвращаемые значения и примеры вызовов интерфейсов.

В разделе «Сообщения об ошибках» описан метод обработки исключительных ситуаций и указаны коды ошибок.

В конце документа приведен список использованных терминов и сокращений.

Содержание

1 Общие сведения

Библиотека «GisCryptoCpp» (Demo-версия) содержит программные интерфейсы, реализованные на языке C++, и предназначена для вызова основных функций криптографической платформы «Litoria Crypto Platform».

В библиотеку «GisCryptoCpp» (Demo-версия) включены интерфейсы для вызова следующих функций криптографической платформы «Litoria Crypto Platform»:

- получение коллекции сертификатов из хранилища;
- установка сертификата в хранилище (без привязки к закрытому ключу);
- создание усовершенствованной электронной подписи (УЭП);
- добавление УЭП;
- проверка УЭП;
- шифрование данных;
- расшифровывание данных.

2 Описание программных интерфейсов

При использовании программных интерфейсов библиотеки «GisCryptoCpp» (Demo-версия) и наличии на рабочей станции криптографической платформы «Litoria Crypto Platform» можно реализовать клиентское программное обеспечение, в котором пользователю предоставляются перечисленные выше функции криптографической платформы «Litoria Crypto Platform».

Классификация программных интерфейсов представлена в таблице 2.1.

Таблица 2.1. Классификация программных интерфейсов

Тип интерфейсов	Интерфейс	Назначение
Интерфейс для настройки доступа к прокси-серверу	setProxy	Настройка имени пользователя и пароля для доступа к прокси-серверу
Интерфейсы для работы с сертификатами	getCertFromStore	Получение сертификата из хранилища
	viewCertificateUi	Отображение сертификата
	getAllCertFromStore	Получение коллекции сертификатов из хранилища
	installCertificate	Установка сертификата в заданное хранилище (без привязки к закрытому ключу)
Интерфейсы для работы с ЭП	signData	Создание УЭП
	addSign	Добавление УЭП
	verifyEds	Проверка УЭП
Интерфейс для шифрования файла	encryptData	Шифрование данных
Интерфейс для расшифровывания файла	decryptData	Расшифровывание данных

2.1 Доступ к прокси-серверу. Интерфейс setProxy

Заголовок объявления интерфейса:

```
void setProxy (std::wstring name, std::wstring password);
```

Входные параметры:

name – имя пользователя

password – пароль

Пример вызова:

```
GisCryptoCpp::GisCrypto crypto;
crypto.setProxy(L"domen\\user", L"123456");
```

2.2 Получение сертификата из хранилища. Интерфейс getCertFromStore

Заголовок объявления интерфейса:

```
std::vector<unsigned char> getCertFromStore(const std::wstring& storeName) const;
```

Входные параметры:

StoreName – имя хранилища

Примеры имен хранилищ:

- «MY» – хранилище личных сертификатов
- «CA» – сертификаты удостоверяющего центра
- «ROOT» – корневые сертификаты
- «AddressBook» – сертификаты других пользователей

Возвращаемое значение:

Сертификат

Пример вызова:

```
GisCryptoCpp::GisCrypto crypto;
std::vector<unsigned char> cert = crypto.getCertFromStore(L"MY");
```

2.3 Отображение сертификата. Интерфейс viewCertificateUi

Заголовок объявления интерфейса:

```
void viewCertificateUi(const std::wstring& title, const std::vector<unsigned char> cert) const;
```

Входные параметры:

title – заголовок окна
cert – сертификат

Пример вызова:

```
GisCryptoCpp::GisCrypto crypto;
crypto.viewCertificateUi(L"Сертификат получателя", decodedCert);
```

2.4 Получение коллекции сертификатов из хранилища. Интерфейс getAllCertFromStore

Заголовок объявления интерфейса:

```
std::list<std::vector<unsigned char> > getAllCertFromStore(const std::wstring& storeName) const;
```

Входные параметры:

StoreName – имя хранилища

Примеры имен хранилищ:

- «MY» – хранилище личных сертификатов
- «CA» – сертификаты удостоверяющего центра
- «ROOT» – корневые сертификаты
- «AddressBook» – сертификаты других пользователей

Возвращаемое значение:

Коллекция сертификатов

Пример вызова:

```
GisCryptoCpp::GisCrypto crypto;
std::list<std::vector<unsigned char> > receiversCertList = crypto.getAllCertFromStore(L"AddressBook");
```

2.5 Установка сертификата в хранилище. Интерфейс installCertificate

Заголовок объявления интерфейса:

```
void installCertificate(const std::vector<unsigned char>& cert, const std::wstring& storeName);
```

Входные параметры:

cert – устанавливаемый сертификат
storeName – имя хранилища

Примеры имен хранилищ:

- «MY» – хранилище личных сертификатов
- «CA» – сертификаты удостоверяющего центра
- «ROOT» – корневые сертификаты
- «AddressBook» – сертификаты других пользователей

Пример вызова:

```
GisCryptoCpp::GisCrypto crypto;
crypto.installCertificate(certToInstall, L"AddressBook");
```

2.6 Создание УЭП. Интерфейс signData

Заголовок объявления интерфейса:

```
void signData(const std::vector<unsigned char>& data, const std::vector<unsigned char>& cert, const std::wstring& tspAddress, EdsFlag edsFlag, std::vector<unsigned char>& outData) const;
```

Входные параметры:

data – данные для подписи
cert – сертификат подписывающего лица
tspAddress – адрес используемой службы штампов времени
edsFlag – управляющий флаг
Возможные значения флага:
- DetachedEds – флаг отделенной ЭП
- IncludeTimeStamp – флаг включения штампа времени
- CreateCades – флаг создания УЭП

outData – выходные подписанные данные

Пример вызова:

```
std::vector<unsigned char> cert;
std::vector<unsigned char> data;
std::vector<unsigned char> outData;
std::wstring s = L"http://tsp.gaz-is.ru/tsp/tsp.srf";
GisCryptoCpp::EdsFlag edsFlag = GisCryptoCpp::CreateCades;
GisCryptoCpp::GisCrypto crypto;
crypto.signData(data, cert, s, edsFlag, outData);
```

2.7 Добавление УЭП. Интерфейс addSign

Заголовок объявления интерфейса:

```
void addSign(const std::vector<unsigned char>& data, const std::vector<unsigned char>& cert, const std::wstring& tspAddress, EdsFlag edsFlag, std::vector<unsigned char>& outData) const;
```

Входные параметры:

data – данные для добавления подписи
cert – сертификат подписывающего лица
tspAddress – адрес используемой службы штампов времени
edsFlag – управляющий флаг
Возможные значения флага:
- DetachedEds – флаг отделенной ЭП
- IncludeTimeStamp – флаг включения штампа времени
- CreateCades – флаг создания УЭП

outData – выходные подписанные данные

Пример вызова:

```
std::vector<unsigned char> cert;
std::vector<unsigned char> data;
std::vector<unsigned char> outData;
std::wstring s = L"http://tsp.gaz-is.ru/tsp/tsp.srf";
GisCryptoCpp::EdsFlag edsFlag = GisCryptoCpp::CreateCades;

GisCryptoCpp::GisCrypto crypto;

crypto.addSign(outData, cert, s, edsFlag, addedEdsData);
```

2.8 Проверка УЭП. Интерфейс verifyEds

Заголовок объявления интерфейса:

std::list<SignatureInfo> verifyEds(const std::vector<unsigned char>& signedData, const std::vector<unsigned char>& detachedData, EdsFlag edsflag) const;

Входные параметры:

signedData – подписанные данные
detachedData – отделенные данные
edsflag – управляющий флаг
Возможные значения флага:

- DetachedEds – флаг отделенной ЭП
- IncludeTimeStamp – флаг включения штампа времени
- CreateCades – флаг создания УЭП

Возвращаемое значение:

Информация о проверке УЭП (std::list<SignatureInfo>):
unsigned long getIndex() const; – метод получения индекса подписи
bool getVerifyResult() const; – метод получения результата проверки ЭП
bool getVerifyCertificateResult() const; – метод получения результата проверки сертификата
const std::vector< unsigned char>& getCertificate() const; – метод получения сертификата
time_t getSignatureTime() const; – метод получения время создания подписи
const std::vector< unsigned char>& getRawSiganture() const; – метод получения подписи
const std::wstring& getComment() const; – метод получения комментария к подписи
bool isAdvanced() const; – метод получения флага УЭП
bool isTimeStampIncluded() const; – метод получения флага наличия штампа времени
bool isCounterSignature() const; – метод получения флага заверяющей подписи
const GisCryptoError& getCryptoError() const; – метод получения ошибок подписи

Пример вызова:

```
GisCryptoCpp::GisCrypto crypto;

std::list<GisCryptoCpp::SignatureInfo> signInfos = crypto.verifyEds(signedData, std::vector<unsigned char>(), 0);
    if (signInfos.size() > 0) {
        std::list<GisCryptoCpp::SignatureInfo>::const_iterator it;
        for(it = signInfos.begin(); it != signInfos.end(); it++) {
                crypto.viewCertificateUi(L"Сертификат подписчика", it->getCertificate());
                }
        }
```

2.9 Шифрование данных. Интерфейс encryptData

Заголовок объявления интерфейса:

void encryptData(const std::list<std::vector<unsigned char> >& certs, const std::vector<unsigned char>& data, std::vector<unsigned char>& outData) const;

Входные параметры:

certs – сертификаты получателей
data – данные для шифрования
outData – выходные зашифрованные данные

Пример вызова:

```
GisCryptoCpp::GisCrypto crypto;

std::vector<unsigned char> data;
std::vector<unsigned char> encryptedData;
std::list<std::vector<unsigned char> > receiversCertList;

receiversCertList = crypto.getAllCertFromStore(L"AddressBook");

crypto.encryptData(receiversCertList, data, encryptedData);
```

2.10 Расшифровывание данных. Интерфейс decryptData

Заголовок объявления интерфейса:

```
std::vector<unsigned char> decryptData(const std::vector<unsigned char>& data, std::vector<unsigned char>& outData) const;
```

Входные параметры:

data – зашифрованные данные
outData – расшифрованные данные

Выходные параметры:

Сертификат получателя зашифрованного сообщения

Пример вызова:

```
GisCryptoCpp::GisCrypto crypto;

std::vector<unsigned char> encryptedData;
std::vector<unsigned char> decryprtedData;

std::vector< unsigned char> decodedCert = crypto.decryptData(encryptedData, decryprtedData);
```

3 Обработка исключительных ситуаций

Функции обработки исключений помогают обрабатывать любые непредвиденные или исключительные ситуации, происходящие при выполнении основных операций. Для обработки исключений в программных интерфейсах используются ключевые слова try и catch.

Пример обработки исключения при выполнении операции создания УЭП:

```
try
{
	crypto.signData(data, cert, s, edsFlag, outData);
}
catch(GisCryptoCpp::GisCryptoException excp)
{
	GisCryptoCpp::GisCryptoError error = excp.getError();
	std::list<int> errorCodeList = error.getBhGiSErrorCodes();
	std::list<int>::const_iterator codeIt;
	for (codeIt = errorCodeList.begin(); codeIt != errorCodeList.end(); codeIt++) {
	int code = *codeIt;
	}
	std::list<std::wstring> errorStringList = error.getBhGiSErrorStrings();
	std::list<std::wstring>::const_iterator stringIt;
	for (stringIt = errorStringList.begin(); stringIt != errorStringList.end(); stringIt++) {
	std::wstring string = *stringIt;
	}
	unsigned long systemErrorCode = error.getSystemErrorCode();
	std::wstring systemErrorString = error.getSystemErrorText();
	}
}
```

Описание методов получения информации об ошибке представлено в таблице 3.1.

Таблица 3.1. Описание методов получения информации об ошибке

Метод	Описание	Возвращаемое значение
const GisCryptoError& getError();	Метод получения ошибки	Объект, содержащий информацию об ошибках библиотеки
unsigned long getSystemErrorCode() const;	Метод получения системной ошибки	Код системной ошибки
const std::wstring& getSystemErrorText() const;	Метод получения описания системной ошибки	Описание системной ошибки
const std::list<int>& getBhGiSErrorCodes() const;	Метод получения списка ошибок модуля	Список ошибок модуля
const std::list<std::wstring>& getBhGiSErrorStrings() const;	Метод получения списка ошибок модуля	Список описаний ошибок модуля

Термины и сокращения

CA	–	Certifying Authority (Сертифицирующая организация)
TSP	–	Time-Stamp Protocol (Протокол штампов времени)
Сертификат	–	документ на бумажном носителе или электронный документ с ЭП уполномоченного лица удостоверяющего центра, которые включают в себя открытый ключ ЭП и которые выдаются удостоверяющим центром участнику информационной системы для подтверждения подлинности ЭП и идентификации владельца сертификата ключа подписи
Усовершенствованная электронная подпись (УЭП)	–	электронная подпись, усовершенствованная, в качестве меры борьбы с общепризнанными угрозами безопасности, добавлением (как необходимое требование) признаков ее (подписи) регламента и доказательств подлинности – таких, как штамп времени, данные об отзыве сертификата и др.
Электронная подпись (ЭП)	–	реквизит электронного документа, предназначенный для защиты данного электронного документа от подделки, полученный в результате криптографического преобразования информации с использованием закрытого ключа ЭП и позволяющий идентифицировать владельца сертификата ключа подписи, а также установить отсутствие искажения информации в электронном документе.