ООО «ФИРМА «ИНФОКРИПТ»

IC_BicryptTools

Спецификация библиотеки

г. Москва 2022 год

Оглавление

Введение

Программный продукт IC_BicryptTools является средством создания, проверки электронной подписи и шифрования документов, предназначенным для встраивания в прикладные системы. Используемое криптографическое ядро — сертифицированное СКЗИ «Бикрипт 5.0».

Программный продукт IC_BicryptTools обеспечивает формирование и проверку ЭП в соответствии с ГОСТ Р34.10-2001 и ГОСТ Р34.10-2012 и хеширование данных в соответствии с ГОСТ Р34.11-94 и ГОСТ Р34.11-2012.

Программный продукт представляет собой набор java-библиотек и динамических библиотек, работающих в операционных системах семейства Windows, Linux, AIX и Solaris.

Работа в многопоточном режиме осуществляется при соблюдении требований, указанных в описании соответствующих методов.

Состав библиотеки

1. Основной модуль:
    - icbicrypttools
2. Модули СКЗИ «Бикрипт 5.0»:
    - AIX
        - libbicr5_64.so
        - libasn1pars_64.so
        - libcryptox509_64.so
        - libcms80_64.so
    - Linux
        - libbicr5_64.so
        - libasn1pars_64.so
        - libcryptox509_64.so
        - libcms80_64.so
    - Solaris
        - libbicr5_64.so
        - libasn1pars_64.so
        - libcryptox509_64.so
        - libcms80_64.so
    - genprnd
        - readme.txt
        - genprnd.exe
        - bicr_adm.dll
        - grn.dll
    - bicr5.dll
    - bicr5_64.dll
    - Grn.dll
    - Grn64.dll
    - asn1pars_64.dll
    - asn1pars.dll
    - cryptox509_64.dll
    - cryptox509.dll
    - cms80.dll
    - cms80_64.dll
3. Криптобиблиотеки Java:
    - cryptolib
    - x509
    - ocsp

Исключения

Все методы, в случае возникновения исключительных ситуаций, перенаправляют их в вызывающие библиотеку функции.

Класс, описывающий исключение: **ICException**

Возможные возвращаемые значения **ICError (таблица ICError)**:

Значение	Код	Текстовое описание
ERR_OK	0	Ошибок нет
ERR_MEM	1	Недостаточно памяти
ERR_BAD_SIGN	2	Подпись неверна
ERR_BAD_LEN	3	Длина буфера неверна
ERR_BAD_USER	4	Данные пользователя не соответствуют ожидаемым
ERR_CRYDRV	5	Ошибка внутреннего тестирования криптоопераций
ERR_INIT_CRYMASK	6	Ошибка декодирования мастер-ключа
ERR_NOT_SUPPORTED	8	Не поддерживаемая функция
ERR_CRC_SK_FILE	9	Ошибка контрольной суммы файла с закрытым ключом
ERR_NO_SIGN	11	Нет подписи
ERR_OPEN_FILE	12	Ошибка открытия файла
ERR_OPEN_PUB	14	Ошибка открытия файла БОК
ERR_TOO_MANY	16	Количество носителей ключа превысило максимально допустимый предел
ERR_READ_MK_FILE	18	Ошибка чтения файла с мастер-ключом
ERR_SIGN_NO_REG	19	Идентификатор подписи не зарегистрирован в БОК
ERR_BAD_SELFTEST	20	Внутренние тесты библиотеки проведены с ошибкой
ERR_GK_READ	21	Ошибка чтения главного ключа
ERR_UZ_READ	22	Ошибка чтения узла замены
ERR_CRC_GKUZ	23	Ошибка контрольной суммы главного ключа
ERR_GKUZ_PSW	24	Главный ключ требует ввода пароля
ERR_DSCH	25	Не найден ДСЧ
ERR_CRC_TM	26	Ошибка контрольной суммы при чтении ТМ
ERR_LOAD_GRN_DLL	27	Ошибка загрузки зависимых библиотеки
ERR_STOP	28	Процесс инициализации ПДСЧ прерван пользователем
ERR_TMDRV_NOT_FOUND	29	Ошибка использования ТМ-устройства - нет драйвера TMDRV
ERR_NO_TM_ATTACHED	30	Идентификатор TouchMemory не приложен к считывателю
ERR_READ_TM	31	Ошибка чтения ТМ
ERR_BAD_PARAM	32	Ошибка в параметрах функции
ERR_BAD_HANDLE	33	Ошибка дескриптора (например, происходит обращение к закрытому ранее дескриптору или вместо валидного дескриптора используется случайное значение)
ERR_HANDLE_TYPE	34	Неправильный тип дескриптора (например, вместо типа H_USER в соответствующий параметр функции передается дескриптор типа H_PKEY)
ERR_WRITE_TM	35	Ошибка работы с программным датчиком - требуется инициализация
ERR_READ_NET_FILE	37	Ошибка чтения файла сетевых ключей
ERR_INIT	39	Ошибка инициализации библиотеки, не был вызван cr_init
ERR_LOAD_KEY	40	Ошибка загрузки ключа
ERR_NET_KEY	42	Ошибка сетевого ключа
ERR_NO_CRYP	43	Буфер не был зашифрован
ERR_BAD_CRYP	44	Ошибка расшифрования буфера
ERR_FILE_KEY	45	Ошибка файлового ключа
ERR_READ_FILE	46	Ошибка чтения файла
ERR_WRITE_FILE	47	Ошибка записи файла
ERR_COMPRESS	48	Ошибка архивации данных
ERR_MORE_DATA	49	Длина выделенного буфера недостаточна
ERR_DEVICE_NOT_FOUND	101	Ошибка сервера ЭП. Устройство не найдено
ERR_NO_SOCKET	102	Ошибка сервера ЭП. Нет сокета
ERR_NO_RESOLVE	103	Ошибка сервера ЭП. ERR_NO_RESOLVE

ERR_NO_RESPONSE	104	Ошибка сервера ЭП. Нет отклика
ERR_BAD_PACKET	105	Ошибка сервера ЭП. Неверная структура пакета
ERR_NO_TCPIP	106	Ошибка сервера ЭП. Не поддерживается протокол TCPIP
ERR_NO_KEY	107	Ошибка сервера ЭП. Ключ не найден
ERR_FPSU_BAD_PARAM	108	Ошибка сервера ЭП. Некорректный параметр
ERR_DRIVER_INTERNAL	109	Ошибка сервера ЭП. Внутренняя ошибка драйвера
ERR_TIMEOUT	110	Ошибка сервера ЭП. Превышено время ожидания
ERR_BAD_VERSION	111	Ошибка сервера ЭП. Некорректная версия
ERR_NOT_USED_TMDRV	11110	Ошибка драйвера ТМ
ERR_NOT_USED_DSCH	11111	Ошибка ПДСЧ
ERR_NOT_USED_GKUZ	11112	Отсутствуют ГК и УЗ

Класс PrivateKey

Описание параметров класса PrivateKey приведено в таблице ниже.

Тип	Имя	Описание
String	confidentFileName	Путь к файлу с закрытым ключом
String	keyIdent	Идентификатор закрытого ключа
String	keyMediaId	Номер ключевого носителя

Ошибки, возникающие в ходе выполнения функций данного класса, соответствуют ошибкам **ICError**, описанным в соответствующей таблице в разделе **Исключения**.

1. Методы работы с ключом электронный подписи

1.1. Получить идентификатор ключа

String getKeyIdent()

Назначение:

Получение идентификатора закрытого ключа.

Параметры:

нет

Возвращаемое значение:

Идентификатор закрытого ключа

1.2. Получить серийный номер идентификатора TouchMemory

String getKeyMediaId()

Назначение:

Получение серийного номера идентификатора TouchMemory.

Параметры:

нет

Возвращаемое значение:

Серийный номер идентификатора TouchMemory.

1.3. Освободить ресурсы закрытого ключа

void close**()**

Назначение:

Освобождение ресурсов, занятых закрытым ключом.

Класс ICBicryptTools

1. Конструкторы

ICBicryptTools()

Назначение:

Конструктор создает экземпляр класса, предназначенный для работы с уже загруженными классами нативных библиотек (например, в среде WebSphere), либо для установки ПО ICBicryptTools на диск сервера/ПК с использованием метода installFiles.

При работе в среде WebSphere, перед запуском системы с поддержкой ПО ICBicryptTools необходимо скопировать ключ датчика случайных чисел (prnd.db3) в каталог ***<user.home>***\bicrypt (<user.home> -(System.getProperty("user.home")). Затем загрузить через интерфейс среды WebSphere библиотеки, необходимые для работы с ПО ICBicryptTools.

Пример использования:

```
new ICBicryptTools().installFiles("c:\iccryptotools");
```

ICBicryptTools(String targetPath)

Назначение:

Конструктор создает объект, предназначенный для инициализации ПО ICBicryptTools и загрузки зависимых библиотек. Является основным рабочим экземпляром класса для работы на произвольной платформе.

Параметры:

targetPath	Путь к **каталогу** с зависимыми нативными библиотеками, включая файл с ключом затравки для датчика случайных чисел - prnd.db3

Пример использования:

//Создание экземпляра bicryptTools класса ICBicryptTools при условии ранее установленных зависимых библиотек в каталоге "c:\iccryptotools" с использованием метода installFiles

```
ICBicryptTools bicryptTools = new ICBicryptTools("c:\iccryptotools")
```

2. Методы развертывания библиотеки и подготовки к работе

2.1. Инсталляция библиотеки

void installFiles (**String** path) throws IOException

Назначение:

Установка зависимых библиотек для работы ПО ICBicryptTools в указанный каталог.

Параметры:

path	Путь к каталогу для установки зависимых библиотек.

Возвращаемое значение:

Нет

Описание функции:

Функция копирует зависимые библиотеки в указанный каталог на целевой машине. Функция не проверяет наличие зависимых библиотек в каталоге перед началом копирования. Для оптимизации работы вызов функции должен быть осуществлен однократно. Если на целевой машине уже присутствует необходимый набор нативных библиотек, то вызывать функцию не надо. Метод предназначен для использования в рамках клиентского апплета автоматизированной системы. Метод не рекомендуется к использованию в серверной части автоматизированной системы.

2.2. Указание пути к файлу с затравкой ДСЧ

void setPrndPath(**String** prndPath) throws IOException

Назначение:

Установка затравки ДСЧ в криптоядро Бикрипт5.

Параметры:

Path	Путь к файлу prnd.db3 с затравкой.

Возвращаемое значение:

Нет

Описание метода:

Метод используется при работе в среде WebSphere.

Пример использования:

```
ICBicryptTools bicryptTools = new ICBicryptTools();
bicryptTools.setPrndPath("c:\iccryptotools\prnd.db3");
```

3. Методы загрузки закрытого ключа

3.1. Загрузить закрытый ключ с носителя TouchMemory

PrivateKey getPrivateKeyTM(**int** *timeout*)

Параметры:

timeout	Время ожидания приложения таблетки

Назначение:

Загружает закрытый ключ с носителя TouchMemory (таблетка).

Возвращаемое значение:

Метод возвращает объект PrivateKey.

Описание:

Метод загружает закрытый ключ с носителя TouchMemory и возвращает объект PrivateKey.

3.2. Загрузить ключ из файла с флеш-накопителя

PrivateKey getPrivateKeyFile(**String** confidentFileName, **String** password)

Параметры:

confidentFileName	полный путь к файлу с закрытым ключом
Password	пароль

Назначение:

Загружает закрытый ключ из файла.

Возвращаемое значение:

Метод возвращает объект PrivateKey.

Описание:

Метод загружает закрытый ключ из файла со съемного флеш-накопителя и возвращает объект PrivateKey.

3.3. Загрузить ключ ПАК «Сервер ЭП»

PrivateKey getPrivateKeyFPSU(**String** *ipaddress*, **int** *port*, **int** *timeout*, **int** *keyNumber*)

PrivateKey getPrivateKeyFPSU(**String** *ipaddress*, **int** *port*, **int** *timeout*,
int *keyNumber*, ***ProtocolType*** *protocolType*)

Параметры:

ipaddress	IP-адрес ПАК «Сервер ЭЦП» (на базе ФПСУ-IP)
port	Порт ПАК «Сервер ЭЦП»

timeout	Таймаут подключения к ПАК «Сервер ЭЦП»
keyNumber	Номер загружаемого ключа
protocolType	Тип протокола доступа к ПАК «Сервер ЭЦП»

Назначение:

Загружает закрытый ключ.

Возвращаемое значение:

Метод возвращает объект PrivateKey.

Описание:

Метод загружает закрытый ключ ПАК «Сервер ЭП» и возвращает объект PrivateKey.

3.4. Загрузить двухкомпонентный ключ с носителя TouchMemory

PrivateKey getPrivateKeyMK_TM(**String** *masterKeyPath*, **String** *gkPath*, **String** *uzPath*, **int** *timeout*)

Параметры:

masterKeyPath	Путь к файлу с мастер-ключом
gkPath	Путь к файлу с главным ключом
uzPath	Путь к файлу с узлами замены
timeout	Время ожидания контакта с ТМ

Назначение:

Загружает двухкомпонентный закрытый ключ с носителя ТМ.

Возвращаемое значение:

Метод возвращает объект PrivateKey.

Описание:

Метод загружает двухкомпонентный закрытый ключ с компонентами на ТМ и файловой системе и возвращает объект PrivateKey.

3.5. Загрузить двухкомпонентный ключ со съемного носителя

PrivateKey getPrivateKeyMK_File(**String** *masterKeyPath*, **String** *gkPath*, **String** *uzPath*, **String** *confidentFileName*)

Параметры:

masterKeyPath	Путь к файлу с мастер-ключом
gkPath	Путь к файлу с главным ключом
uzPath	Путь к файлу с узлами замены
confidentFileName	Путь к файлу с компонентой ключа

Назначение:

Загружает двухкомпонентный закрытый ключ со съемного носителя.

Возвращаемое значение:

Метод возвращает объект PrivateKey.

Описание:

Метод загружает двухкомпонентный закрытый ключ с компонентами на съемном носителе и файловой системе и возвращает объект PrivateKey.

4. Формирование ЭП

4.1 Формирование ЭП

BicryptSign sign(byte[] data**, PrivateKey** key**)** throws **ICException**

BicryptSign sign(File file**, PrivateKey** key**)** throws **ICException**

void sign(String filename, **PrivateKey** key**)** throws **ICException**

Назначение:

Формирование ЭП в формате Бикрипт. Метод, принимающий в качестве параметра путь к файлу, формирует ЭП сразу добавляя ее данные в конец файла. Применяется к файлам большого размера.

Параметры:

data	Блок данных
file	Файл
filename	Путь к файлу
key	закрытый ключ, которым осуществляется подпись данных

Возвращаемое значение:

ЭП в формате Бикрипт в виде объекта BicryptSign.

Описание:

Метод предназначен для формирования ЭП в формате Бикрипт для блока данных и файла. При формировании ЭП для файла в конец файла записывается ЭП в формате Бикрипт.

Метод, принимающий путь к файлу, может работать с файлами большого размера.

Пример использования:

```
ICBicryptTools bicryptTools = new ICBicryptTools(pathToNativeLibs);
byte[] data = …;
PrivateKey key = bicryptTools.getPrivateKeyTM(10000);
BicryptSign bSign = bicryptTools.sign(data, key);
```

4.2 Расчет хэш-значения

byte[] calcHash(byte[] data**, DigestParamSet** digestParamSet**)**

byte[] calcHash (InputStream in**, DigestParamSet** paramset**)**

byte[] calcHash (byte[] data**)**

Назначение:

Расчет хэш-значения.

Параметры:

data	Блок данных
in	Входной поток с данными
digestParamSet	Алгоритм расчета хеша

Возвращаемое значение:

Хэш-значение в виде массива байт.

Описание:

Метод предназначен для расчета хэш-значения для блока данных и файла.

Пример использования:

```
ICBicryptTools bicryptTools = new ICBicryptTools(pathToNativeLibs);
byte[] data = …;
byte[] hash= bicryptTools.calcHash(data, DigestParamSet.HASH_GOST3411_2012_256);
```

4.3 Формирование ЭП для хэш-значения

BicryptSign signHash(byte[] hash, PrivateKey key**)**

Назначение:

Формирование ЭП в формате Бикрипт.

Параметры:

data	данные
key	закрытый ключ, которым осуществляется подпись данных

Возвращаемое значение:

Объект SignContext, используемый в методах формирования ЭП.

Описание:

Метод предназначен для создания контекста подписи при выполнении задач, связанных с многократным формированием ЭП одним и тем же ключом.

Пример использования:

```
ICBicryptTools bicryptTools = new ICBicryptTools(pathToNativeLibs);
byte[] data = …;
byte[] hash= bicryptTools.calcHash(data, DigestParamSet.HASH_GOST3411_2012_256);
BicryptSign bSign = bicryptTools.signHash(hash, key);
```

5. Методы проверки ЭП

5.1 Проверка ЭП

boolean check(byte[] testData**, BicryptSign** signtestData**,**
PublicKeyBase pkb**, PublicKeyBase** admPkb**)** throws **ICException**

Параметры:

testData	Блок данных
signtestData	ЭП в формате Бикрипт
pkb	База открытых ключей пользователей
admPkb	База открытых ключей администраторов

Возвращаемое значение:

Метод возвращает результат проверки. true говорит об успешной проверке ЭП. Если ЭП не сошлась – false. Во всех остальных случаях ICException.

Пример использования:

```
ICBicryptTools bicryptTools = new ICBicryptTools(pathToNativeLibs);
byte[] data = …;
PublicKeyBase pkb = new PublicKeyBase("D:\sign00CA.xxx")
PublicKeyBase admPkb = new PublicKeyBase("D:\sign00CA.xxx")
PrivateKey key = bicryptTools.getPrivateKeyTM(10000);
BicryptSign bSign = bicryptTools.sign(data, key);
boolean res = bicryptTools.check(data, bSign, pkb, admPkb)
```

5.2 Проверка ЭП для файла

List**<CheckBicryptResult> check(String** fileName**, boolean** deleteSign**, PublicKeyBase** pkb**,**
PublicKeyBase admPkb**)** throws **ICException**

Параметры:

fileName	путь к проверяемому файлу
deleteSign	удалять или нет все ЭП (true - удалять)
pkb	База открытых ключей пользователей
admPkb	База открытых ключей администраторов

Возвращаемое значение:

Метод возвращает список результатов проверки.

Пример использования:

```
List<CheckBicryptResult> checks = bcryTools.check(_testPath + "sinedDoc2.sgn", true,
                                                    testBok, bok17);
for (CheckBicryptResult check : checks) {
      System.out.println("Sign Number '" + check.getSignNumber() + "' is " +
                        (check.isOk() ? "Good" : "BAD!!!"));
}
```

5.3 Проверка ЭП для хэш-значения

boolean checkHash(**byte[]** hash**, BicryptSign** signData**,**
PublicKeyBase pkb**, PublicKeyBase** admPkb**)** throws **ICException**

Параметры:

hash	хэш, под которым необходимо проверить ЭП
signData	ЭП в формате Бикрипт
pkb	база открытых ключей пользователей
admPkb	база открытых ключей администраторов

Возвращаемое значение:

Метод возвращает результат проверки. true говорит об успешной проверке ЭП. Если ЭП не сошлась – false. Во всех остальных случаях ICException.

Пример использования:

```
ICBicryptTools bicryptTools = new ICBicryptTools(pathToNativeLibs);
byte[] data = …;
PublicKeyBase pkb = new PublicKeyBase("D:\sign00CA.xxx")
PublicKeyBase admPkb = new PublicKeyBase("D:\sign00CA.xxx")
PrivateKey key = bicryptTools.getPrivateKeyTM(10000);
byte[] hash= bicryptTools.calcHash(data);
BicryptSign bSign = bicryptTools.signHash(hash, key);
boolean res = bicryptTools.check(data, bSign, pkb, admPkb)
```

5.4 Проверка ЭП для хэш-значения с использованием сервиса OCSP

boolean checkHash(byte[] hash**, BicryptSign** sign**,**
OCSP ocsp, **OcspFilter...** filters**)** throws **ICException**

boolean checkHash(byte[] hash**, byte[]** rawSign**, String** bicryptIdent**,**
OCSP ocsp, **OcspFilter...** filters**)** throws **ICException**

boolean checkHash(byte[] hash**, byte[]** rawSign**, String** bicryptIdent**,**
OCSP ocsp, **Calendar** checkTime, **OcspFilter...** filters**)** throws **ICException**

Параметры:

hash	хэш, под которым необходимо проверить ЭП
sign	ЭП в формате Бикрипт
rawSign	ЭП в формате RAW («чистая» подпись)
bicryptIdent	идентификатор Бикрипт подписанта
ocsp	экземпляр класса OCSP
checkTime	время, на которое проверяется актуальность сертификата

filters	[опционально] Ограничительный фильтр для поиска сертификата в OCSP

Возвращаемое значение:

Метод возвращает результат проверки. true говорит об успешной проверке ЭП. Если ЭП не сошлась – false. Во всех остальных случаях ICException.

Пример использования:

```
ICBicryptTools bicryptTools = new ICBicryptTools(pathToNativeLibs);
byte[] hash = …;
byte[] rawSign = …;
String ident = "…";
OCSP ocsp = new OCSP("http://1.1.1.1:9087/MegaCUKSOCSP/processOCSP", 40000);
boolean res = bicryptTools.check(hash, rawSign, ident, ocsp);
```

5.5 Проверка ЭП с использованием сервиса OCSP

boolean check (byte[] data**, BicryptSign** sign**, OCSP** ocsp**)**
List<CheckBicryptResult> check (byte[] dataNSign**, OCSP** ocsp**)**
List<CheckBicryptResult> check(byte[] data**, byte[]** detachBicrSign**, OCSP** ocsp**, OcspFilter...** filters**)**

Параметры:

dataNSign	данные с присоединенной одной или несколькими ЭП формата Бикрипт
ocsp	Экземпляр класса OCSP
sign	Отсоединенная ЭП в формате Бикрипт (объект **BicryptSign**)
detachBicrSign	Отсоединенная ЭП формата Бикрипт (в виде буфера данных)

Возвращаемое значение:

Метод возвращает список результатов проверки каждой подписи.

Пример использования:

```
OCSP ocsp = new OCSP("http://1.1.1.1:9087/MegaCUKSOCSP/processOCSP", 40000);
List<CheckBicryptResult> checks = bcryTools.check(dataNSign, ocsp);
for (CheckBicryptResult check : checks) {
      System.out.println("Sign Number '" + check.getSignNumber() + "' is " +
            (check.isOk() ? "Good" : "BAD!!!"));
}
```

5.6 Проверка ЭП с использованием сервиса OCSP с учетом ограничений

boolean check (byte[] data**, BicryptSign** sign**, OCSP** ocsp, **OcspFilter...** filters**)**
List<CheckBicryptResult> check (byte[] dataNSign**, OCSP** ocsp, **OcspFilter...** filters**)**
List<CheckBicryptResult> check (byte[] dataNSign**, OCSP** ocsp**, Calendar** checkTime**, OcspFilter...** filters**)**

Параметры:

dataNSign	данные с присоединенной одной или несколькими ЭП формата Бикрипт
ocsp	Экземпляр класса OCSP
checkTime	Время, на которое осуществляется проверка

sign	Отсоединенная ЭП в формате Бикрипт
filters	Ограничительный фильтр для поиска сертификата в OCSP

Возвращаемое значение:

Метод возвращает список результатов проверки каждой подписи.

Пример использования:

```
OCSP ocsp = new OCSP("http://1.1.1.1:9087/MegaCUKSOCSP/processOCSP", 40000);
OcspCryptoNet cnInfo = new OcspCryptoNet("00CA", "017");
OcspFilter cnFilter = new OcspFilter(OcspFilterType.CryptoNet, cnInfo);
List<CheckBicryptResult> checks = bcryTools.check(dataNSign, ocsp, cnFilter);
for (CheckBicryptResult check : checks) {
	System.out.println("Sign Number '" + check.getSignNumber() + "' is " +
		(check.isOk() ? "Good" : "BAD!!!"));
}
```

6. Разбор ЭП под данными

6.1 Получить список идентификаторов Бикрипт подписантов

List<String> getBicryptIdentifiers (byte[] signedData **)**

Параметры:

signedData данные с присоединенной одной или несколькими ЭП формата Бикрипт

Возвращаемое значение:

Метод возвращает список строк - идентификаторов подписантов.

Пример использования:

```
List<String> idents = bcryTools.getBicryptIdentifiers(signedData);
for(String ident : idents)
      System.out.println("Bicrypt identifier: " + ident + "\n");
```

6.2 Получить список объектов ЭП

public List<BicryptSign> getBicryptSigns(byte[] signedData**)** throws **ICException**

Параметры:

signedData данные с присоединенной одной или несколькими ЭП формата Бикрипт

Возвращаемое значение:

Метод возвращает список объектов **BicryptSign**.

Пример использования:

```
List<BicryptSign> bcrySigns = bcryTools.getBicryptSigns(signedDoc);
int i = 1;
for (BicryptSign bSign : bcrySigns) {
     System.out.println("Sign Number '" + i + "' ident: '" +
                                    bSign.getBicryptIdentifier() + "'");
     i++;
}
```

7. Шифрование и расшифрование данных

7.1 Зашифровать данные

byte[] encryptWithServerDS (ServerDS serverDS**, byte[]** dataToEncrypt**, String** pathFileKey**)**

Параметры:

serverDS	экземпляр класса ServerDS, интерпретирующий сервер ЭП
dataToEncrypt	данные, которые необходимо зашифровать
pathFileKey	полный путь к зашифрованному симметричному ключу

Возвращаемое значение:

Метод возвращает набор байт – зашифрованные данные

Пример использования:

```
String fileKey = "C:\\encrypted.key";
byte[] data = new byte[] {0x01, 0x02… 0xFF};
ServerDS dsServer = new ServerDS("192.168.1.0", 850, 2000);
byte[] encryptedData = bcryTools.encryptWithServerDS(dsServer, data,
fileKey);
```

7.2 Расшифровать данные

byte[] decryptWithServerDS (ServerDS serverDS**, byte[]** dataToDecrypt**,**
String pathFileKey**)** throws **ICException**

Параметры:

serverDS	экземпляр класса ServerDS, интерпретирующий сервер ЭП
dataToDecrypt	данные, которые необходимо расшифровать
pathFileKey	полный путь к зашифрованному симметричному ключу

Возвращаемое значение:

Метод возвращает набор байт – расшифрованные данные.

Пример использования:

String fileKey = "C:\\encrypted.key";
ServerDS dsServer = new ServerDS("192.168.1.0", 850, 2000);
byte[] decryptedData = bcryTools.decryptWithServerDS(dsServer, encryptedData, fileKey);

8. Получить данные об ЭП из потока

BigFileInfo parseBigData(InputStream in**)** throws **ICException**

Параметры:

in поток входных данных

Возвращаемое значение:

Структура ***BigFileInfo*** с данными об ЭП.

Пример использования:

```
BicryptContext bicrCtx = new BicryptContext(KeyStorageType.FileSystem);
BigFileInfo info = bicrCtx.getDigestInfo(new ByteArrayInputStream(signedData));
```

Класс BicryptSign

Класс предназначен для работы с структурой электронной подписи в формате Бикрипт.

Методы

byte[] **getSignOnly**

Назначение:

Получить «чистую» ЭП

Возвращаемое значение:

Массив с электронной подписью

String getBicryptIdentifier()

Назначение:

Получить идентификатор ключа электронной подписи в формате Бикрипт

Возвращаемое значение:

Идентификатор ключа в формате Бикрипт

byte[] **getBicryptSign()**

Назначение:

Получить ЭП в формате Бикрипт

Возвращаемое значение:

ЭП в формате Бикрипт

Класс PublicKeyBase

Класс предназначен для работы с базой открытых ключей (БОК).

1. Конструкторы

PublicKeyBase (String path)

Назначение:

База открытых ключей

Параметры:

path	Полный путь к файлу БОК

2. Методы

2.1. Освободить ресурсы БОК

void close()

Назначение:

Освобождение ресурсов, занятых базой открытых ключей.

Класс CheckBicryptResult

Описание:

Класс представляет результат проверки ЭП в формате Бикрипт.

Методы:

1. **Порядковый номер ЭП**
   int **getSignNumber**()
   Метод возвращает число – порядковый номер ЭП.
2. **Криптографический результат проверки**
   boolean **isOk**()
   Метод возвращает **true** в случае успешной проверки ЭП и **false** в обратном случае.
3. **Подписанные данные**
   byte[] **getSignedContent**()
   Метод возвращает подписанные данные (без самой ЭП) в виде массива байт.
4. **Ошибка СКЗИ Бикрипт**
   int **getError**()
   Метод возвращает код ошибки при проверке ЭП Бикрипт

Класс ServerDS

Описание:

Класс интерпретирует сервер ЭП.

Конструктор:

ServerDS (String addr, int port, int timeout)

Параметры:

addr	URL сервера ЭП
port	Порт для доступа к серверу ЭП
timeout	Максимальное время ожидания отклика от сервера

Класс OCSP

Описание:

Класс предназначен для работы с сервером OCSP, предназначенного для получения статуса сертификата. Канал взаимодействия с сервером OCSP должен быть защищен двусторонним SSL.

1. Конструктор

public OCSP (**String** url, **int** timeout**)**

Назначение:

Конструктор предназначен для создания объекта, предназначенного для взаимодействия с сервером OCSP на основе полученного массива байт.

Параметры:

url	Полный URL к серверу OCSP
Timeout	Время ожидания ответа от сервера

Пример использования:

ICBicryptTools bicryptTools = new ICBicryptTools(“c:\icbicrypttools”)

OCSP ocsp = new OCSP(“https://ocsp.sberbank.ru:9090/actual”, 2000);

1. Методы получения сертификата в формате Бикрипт

1.1. Получение сертификата с помощью сервера OCSP на основании идентификатора Бикрипт

OcspCertificate findBicryptCertificate(String bicryptIdentifier**)** throws **OcspException**

Назначение:

Метод осуществляет поиск сертификата в формате Бикрипт на сервере OCSP по идентификатору Бикрипт.

Параметры:

bicryptIdentifier	Идентификатор ключа Бикрипт

Возвращаемое значение:

Экземпляр класса **OcspCertificate**.

Пример использования:

ICBicryptTools bicryptTools = new ICBicryptTools(“c:\icbicrypttools”)

OCSP ocsp = new OCSP(“https://ocsp.sberbank.ru:9090/actual”, 2000);

BicryptCertificate bicrCert = ocsp.findCertificate(“00CA2129qТестовый сертификат УЦ”);

System.out.println(“Получен сертификат с идентификатором: ” + BicryptCertificate.getKeyIdent());

1.2. Получение сертификата с помощью сервера OCSP по ФИО и дополнительному параметру

OcspCertificate findBicryptCertificate(String fio**, String** email**, String** snils**, String** number**)** throws **OcspException**

Назначение:

Метод осуществляет поиск сертификата в формате Бикрипт на сервере OCSP. Поиск сертификата по фамилии, имени и отчеству необходимо осуществляется совместно с одним или несколькими параметрами в том числе - e-mail, СНИЛС, табельный номер.

Параметры:

fio	**Обязательный параметр!** ФИО владельца искомого сертификата
email	email владельца искомого сертификата
	Параметр может использоваться только при наличии значения в параметре fio
snils	СНИЛС владельца искомого сертификата
	Параметр может использоваться только при наличии значения в параметре fio
number	табельный номер владельца искомого сертификата
	Параметр может использоваться только при наличии значения в параметре fio

Возвращаемое значение:

Экземпляр класса **OcspCertificate**.

Пример использования:

```
ICBicryptTools bicryptTools = new ICBicryptTools(“c:\icbicrypttools”)

OCSP ocsp = new OCSP(“https://ocsp.sberbank.ru:9090/actual”, 2000);

BicryptCertificate bicrCert = ocsp.findCertificate(“Иванов Иван Иванович”, “”, “”, “1234567890”);

System.out.println(“Получен сертификат с идентификатором: ” + BicryptCertificate.getKeyIdent());
```

Класс BicryptCertificate

Описание:

Класс предназначен для работы с сертификатом в формате Бикрипт.

1. Конструктор

public BicryptCertificate ()

Назначение:

Конструктор предназначен для создания объекта типа BicryptCertificate с параметрами по умолчанию.

Параметры:

нет

Пример использования:

BicryptCertificate cer = new BicryptCertificate();

public BicryptCertificate (byte[] content)

Назначение:

Конструктор предназначен для создания объект типа BicryptCertificate на основе полученного массива байт.

Параметры:

content	Объект типа BicryptCertificate в виде массива байт

Пример использования:

byte[] cer_array = …;

BicryptCertificate cer = new BicryptCertificate(cer_array);

2. Методы

2.1. Получить открытый ключ

byte[] getPublicKey()

Назначение:

Метод возвращает открытый ключ в виде байтового массива.

Возвращаемое значение:

Открытый ключ в виде байтового массива.

2.2. Получить идентификатора ключа в формате Бикрипт

String getBicryptIdentifier()

Назначение:

Метод возвращает идентификатор ключа в формате СКЗИ «Бикрипт».

Возвращаемое значение:

Идентификатор ключа в формате Бикрипт в виде строки.

2.3. Получить фамилию, имя и отчество владельца сертификата

String getFullName()

Назначение:

Метод возвращает фамилию, имя и отчество владельца сертификата.

Возвращаемое значение:

Фамилия, имя, отчество в виде строки.

2.4. Получить должность владельца сертификата

String getTitle()

Назначение:

Метод возвращает должность владельца сертификата.

Возвращаемое значение:

Должность в виде строки.

2.5. Получить код организации (КУЦ) владельца сертификата

String getOrganizationCode()

Назначение:

Метод возвращает код организации (КУЦ) владельца сертификата.

Возвращаемое значение:

Код организации (КУЦ) в виде строки

2.6. Получить наименование подразделения владельца сертификата

String getOrganizationalUnit()

Назначение:

Метод возвращает наименование подразделения владельца сертификата.

Возвращаемое значение:

Наименование подразделения в виде строки.

2.7. Получить табельный номер владельца сертификата

String getPersonnelNumber()

Назначение:

Метод возвращает табельный номер владельца сертификата.

Возвращаемое значение:

Табельный номер в виде строки.

2.8. Получить СНИЛС владельца сертификата

String getSnils()

Назначение:

Метод возвращает СНИЛС владельца сертификата.

Возвращаемое значение:

СНИЛС в виде строки

2.9. Получить дату начала действия сертификата

Calendar getValidNotBefore()

Назначение:

Метод возвращает дату начала действия сертификата.

Возвращаемое значение:

Дата начала действия сертификата.

2.10. Получить дату окончания действия сертификата

Calendar getValidNotAfter()

Назначение:

Метод возвращает дату окончания действия сертификата.

Возвращаемое значение:

Дата окончания действия сертификата

2.11. Получить ИНН владельца сертификата

String getINN()

Назначение:

Метод возвращает ИНН владельца сертификата.

Возвращаемое значение:

ИНН в виде строки

2.12. Получить идентификатор ключа, которым подписан сертификат

String getIssuerBicryptIdentifier()

Назначение:

Метод возвращает идентификатор ключа издателя сертификата.

Возвращаемое значение:

Идентификатор ключа, которым подписан сертификат

2.13. Получить сертификат в виде байтового массива

byte[] getCertificate()

Назначение:

Метод возвращает сертификат в виде байтового массива.

Возвращаемое значение:

Сертификат в виде байтового массива

Класс BicryptPublicKey

Описание:

Класс предназначен для работы с открытым ключом в формате Бикрипт.

1. Конструктор

public BicryptPublicKey (byte[] bicryptPublicKeyBuffer**, String** keyIdent**)**

Назначение:

Конструктор предназначен для создания объекта типа BicryptPublicKey из буфера с сериализованным открытым ключом Бикрипт.

Параметры:

bicryptPublicKeyBuffer	Объект типа BicryptPublicKey в виде массива байт
keyIdent	Идентификатор ключа

Пример использования:

BicryptPublicKey pubKey = new BicryptPublicKey(bcryPubKeyBuffer, “TestKey”);

public BicryptPublicKey (byte[] x509PublicKeyBuffer**, String** keyIdent**, PublicKeyParamSet** paramSet**)**

Назначение:

Конструктор предназначен для создания объекта типа BicryptPublicKey на основе открытого ключа из сертификата X509.

Параметры:

X509PublicKeyBuffer	Открытый ключ из сертификата X.509
keyIdent	Идентификатор ключа
paramSet	Набор параметров открытого ключа (значение перечисления ***PublicKeyParamSet***)

Пример использования:

BicryptPublicKey pubKey = new BicryptPublicKey (x509PubKeyBuffer, “TestKey”, PublicKeyParamSet.B);

2. Методы

2.1 Проверить ЭП под блоком данных

boolean checkBuffer(**byte[]** dataBuffer, **byte[]** sign)

Назначение:

Метод проверяет ЭП ***sign*** под данными ***dataBuffer***.

Возвращаемое значение:

True, если ЭП криптографически верна, иначе – ***false***.

2.2 Проверить ЭП под блоком с хеш-данными

boolean checkDigest (**byte[]** digest, **byte[]** sign)

Назначение:

Метод проверяет ЭП ***sign*** под хеш-данными ***digest***.

Возвращаемое значение:

True, если ЭП криптографически верна, иначе – ***false***.

Класс BigFileInfo

Описание:

Класс, содержащий информацию об ЭП больших данных.

1. Методы

1.1 Получить длину подписанных данных (размер файла с данными)

long get_dataLength()

Назначение:

Получить длину подписанных данных (размер большого файла с данными)

Возвращаемое значение:

Размер файла.

1.2 Получить список данных по ЭП

List<SignInfo> get_signInfos()

Назначение:

Список данных, включающих хеши, ЭП и идентификаторы подписантов.

Возвращаемое значение:

Список объектов ***SignInfo***.

Класс SignInfo

Описание:

Вспомогательный класс, содержащий информацию об ЭП больших данных.

1. Методы

1.1 Получить хеш ГОСТ Р34-11-1994

byte[] get_digest_1994()

Назначение:

Получить хеш ГОСТ Р34-11-94

Возвращаемое значение:

32 байта значения хеша.

1.2 Получить хеш ГОСТ Р34-11-2012-256

byte[] get_digest_2012_SHORT()

Назначение:

Получить хеш ГОСТ Р34-11-2012-256

Возвращаемое значение:

32 байта значения хеша.

1.3 Получить хеш ГОСТ Р34-11-2012-512

byte[] get_digest_2012_LONG()

Назначение:

Получить хеш ГОСТ Р34-11-2012-512

Возвращаемое значение:

64 байта значения хеша.

1.4 Получить ЭП

byte[] get_sign()

Назначение:

Получить ЭП

Возвращаемое значение:

64/128 байт значения ЭП.

1.5 Получить идентификатор ключа

byte[] get_keyIdent()

Назначение:

Получить идентификатор ключа ЭП

Возвращаемое значение:

Строка с идентификатором ключа ЭП.