**СКЗИ «KALKAN-CRYPT»**

**Общее описание СКЗИ и архитектуры построения**

**Описание Функций криптографической библиотеки KalkanCrypt**

|  |  |  |  |
| --- | --- | --- | --- |
| **Функция** | **Обзор** | **Описание** | **Возвращаемые значения** |
| **KC\_Init ().**  Инициализация библиотеки. | unsigned long KC\_Init(void) | – | При успешном завершении возвращает 0. |
| **KC\_GetTokens().**  Обеспечивает получение указателя на строку подключенных устройств типа storage и их количество. | unsigned long (\*KC\_GetTokens)(unsigned long storage, char \*tokens, unsigned long \*tk\_count); | Параметры:  [in] INT storage — тип хранилища;  [out] CHAR \* tokens — указатель на строку подключенных устройств;  [out] INT\* tCount — количество подключенных устройств. | При успешном завершении возвращает 0, в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **KC\_GetCertificatesList ().**  Обеспечивает получение списка сертификатов в виде строки и их количество. | unsigned long (\*KC\_GetCertificatesList)(char \*certificates, unsigned long \*cert\_count); | Параметры:  [out] CHAR \* certAlias — указатель на строку сертификатов;  [out] INT\* count — количество сертификатов. | При успешном завершении возвращает 0,в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **KC\_LoadKeyStore ().**  Загрузка ключей/сертификата их хранилища. | unsigned long (\*KC\_LoadKeyStore)(int storage, char \*password, int passLen, char \*container, int containerLen, char \*alias); | Параметры:  [in] INT storage — тип хранилища;  [in] CHAR \* password — пароль к хранилищу;  [in] CHAR \* container — название хранилища;  [in] CHAR \* alias — label (alias) сертификата. | При успешном завершении возвращает 0,в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **X509LoadCertificateFromFile().**  Загрузка сертификата из файла для дальнейшей работы с ним. | unsigned long (\*X509LoadCertificateFromFile)(char \*certPath, int certType); | Параметры:  [in] CHAR \* certPath — путь до файла сертификата;  [in] INT certType) — тип сертификата (см. KalkanCrypt Cert Types). | При успешном завершении возвращает 0,в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **X509LoadCertificateFromBuffer().**  Загрузка сертификата из памяти. | unsigned long (\*X509LoadCertificateFromBuffer)(unsigned char \*inCert, int certLength, int flag); | Параметры:  [in] CHAR \* inCert — сертификат в виде строки;  [in] INT flag — флаги. Указывается кодировка сертификата (см. Кодировка сертификатов). | При успешном завершении возвращает 0,в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **X509ExportCertificateFromStore().**  Экспорт сертификата из хранилища. | unsigned long (\*X509ExportCertificateFromStore)(char \*alias, int flag, char \*outCert, int \*outCertLength); | Параметры:  [in] CHAR \* alias — label (alias) сертификата;  [in] INT flag — флаги;  [out] CHAR \* outCert — указатель на начало сертификата. | При успешном завершении возвращает 0,в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **X509CertificateGetInfo().**  Обеспечивает получение значений полей/расширений из сертификата. Сертификат должен быть предварительно загружен с помощью одной из функций: LoadKeyStore(), X509LoadCertificateFromFile(),X509LoadCertificateFromBuffer(). | unsigned long (\*X509CertificateGetInfo)(char \*inCert, int inCertLength, int propId, unsigned char \*outData, int \*outDataLength); | Параметры:  [in] CHAR \* inCert — сертификат в виде строки;  [in] INT propId — идентификатор полей/расширений сертификата (см. KalkanCrypt CertPropID);  [out] CHAR \* outData — указатель на значение указанного поля/расширения. | При успешном завершении возвращает 0,в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **X509ValidateCertificate().**  Осуществляет проверку сертификата: проверка срока действия, построение цепочки сертификатов, проверка отозванности по OCSP или CRL. | unsigned long(\*X509ValidateCertificate)(char \*inCert, int inCertLength, int validType, char \*validPath, long long checkTime, char \*outInfo, int \*outInfoLength, [in] INT flags, [out] CHAR \* getResp, [in] INT getRespLenght); | Параметры:  [in] CHAR \* inCert — сертификат в виде строки;  [in] INT validType — тип проверки (OCSP/CRL) (см. KalkanCrypt ValidType);  [in] CHAR \* validPath — путь до CRL или URL OCSP;  [in] \_\_int64 checkTime — дата и время, на момент которого необходимо провести проверку (зарезервировано, в текущей версии не используется);  [out] CHAR \* outInfo — выходная подробная информация о результате проверки;  [in] INT flags — флаги, устанавливающие необходимость вывода OCSP-response и проверки просроченного сертификата.  [out] CHAR \* getResp – вывод ответа от сервиса OCSP. | При успешном завершении возвращает 0,в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **SignData().**  Подписывает данные. | unsigned long(\*SignData)(char \*alias, int flags, char \*inData, int inDataLength, unsigned char \*inSign, int inSignLen, unsigned char \*outSign, int \*outSignLength); | Параметры:  [in] CHAR \* alias — label (alias) сертификата;  [in] INT flags — флаги. Устанавливают формат входных/выходных данных, тип подписи (см. KalkanCrypt flags);  [in] CHAR \* inData — входные данные;  [ref] CHAR \* outSign — выходные данные (в случае первой подписи) и входные/выходные данные (в случае мультиподписи) | При успешном завершении возвращает 0,в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **SignXML().**  Подписывает данные в формате XML. | unsigned long(\*SignXML)(char \*alias, int flags, char \*inData, int inDataLength, unsigned char \*outSign, int \*outSignLength, char \*signNodeId, char \*parentSignNode, char \*parentNameSpace); | Параметры:  [in] CHAR \* alias — label (alias) сертификата;  [in] INT flags — флаги;  [in] CHAR \* signNodeId — идентификатор тэга, который необходимо подписать. Не указывается, если необходимо подписать все содержимое документа;  [in] CHAR \* parentSignNode — идентификатор тэга, в который необходимо поместить значение подписи;  [in] CHAR \* parentNameSpace — пространство имен тэга, в который необходимо поместить значение подписи. Если пространство имен есть, но не будет указано - то тег не найдется;  [in] CHAR \* inData — входные данные;  [out] CHAR \* outSign — выходные данные (подписи). | При успешном завершении возвращает 0,в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **VerifyData().**  Обеспечивает проверку подписи. | unsigned long(\*VerifyData)(char \*alias, int flags, char \*inData, int inDataLength, unsigned char \*inoutSign, int inoutSignLength,  char \*outData, int \*outDataLen, char \*outVerifyInfo, int \*outVerifyInfoLen, int inCertID, char \*outCert, int \*outCertLength); | Параметры:  [in] CHAR \* alias — label (alias) сертификата;  [in] INT flags — флаги;  [in] INT inCertID — идентификатор (порядковый номер) сертификата (начинается с 0);  [in] CHAR \* inData — входные данные;  [in] CHAR \* inSign — подписанные входные данные;  [out] CHAR \* outData — выходные данные;  [out] CHAR \* outVerifyInfo — выходная подробная информация о результате проверки подписи;  [out] CHAR \* outCert — указатель на начало сертификата с подробной информацией. | При успешном завершении возвращает 0,в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **UVerifyData ().**  Обеспечивает проверку подписи. | unsigned long(\*VerifyData)(char \*alias, int flags, char \*inData, int inDataLength, unsigned char \*inoutSign, int inoutSignLength,  char \*outData, int \*outDataLen, char \*outVerifyInfo, int \*outVerifyInfoLen, int inCertID, char \*outCert, int \*outCertLength); | Параметры:  [in] CHAR \* alias — label (alias) сертификата;  [in] INT flags — флаги;  [in] INT inCertID — идентификатор (порядковый номер) сертификата (начинается с 0);  [in] CHAR \* inData — входные данные;  [in] CHAR \* inSign — путь до подписанных входных данных;  [out] CHAR \* outData — выходные данные;  [out] CHAR \* outVerifyInfo — выходная подробная информация о результате проверки подписи;  [out] CHAR \* outCert — указатель на начало сертификата с подробной информацией. | При успешном завершении возвращает 0,в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **VerifyXML().**  Обеспечивает проверку подписи данных в формате XML. | unsigned long (\*VerifyXML)(char \*alias, int flags, char \*inData, int inDataLength, char \*outVerifyInfo, int \*outVerifyInfoLen); | Параметры:  [in] CHAR \* alias — label (alias) сертификата;  [in] INT flags — флаги;  [in] CHAR \* inData — входные данные;  [out] CHAR \* outVerifyInfo — выходная подробная информация о результате проверки подписи. | При успешном завершении возвращает 0,в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **KC\_GetLastError ().**  Обеспечивает получение подробного кода и описание ошибки, возникшей в процессе выполнения функций криптопровайдера KalkanCryptCOM. | unsigned long (\*KC\_GetLastError)(void); | Параметры: | Возвращает 0, если не найдена ошибка, иначе — значение > 0 (см. KalkanCryptCOM Errors).  При вызове очищается протокол работы криптопровайдера и значение кода последней ошибки. |
| **KC\_GetLastErrorString ().**  Обеспечивает получение подробного протокола работы функций криптопровайдера KalkanCryptCOM. | unsigned long (\*KC\_GetLastErrorString)(char \*errorString, int \*bufSize); | Параметры:  [out] CHAR \* errorString — подробный протокол работы функций криптопровайдера; | При успешном завершении возвращает 0, в противном случае – 1.  При вызове очищается протокол работы криптопровайдера и значение кода последней ошибки. |
| **KC\_getSigAlgFromXML ().**  Обеспечивает получение алгоритма подписи из XML. | unsigned long (\*KC\_getSigAlgFromXML)(const char\* xml\_in, int xml\_in\_size, char \*retSigAlg, int \*retLen); | Параметры:  [in] CHAR \* inXML — входные данные в формате XML;  [out] CHAR \* outSigAlg — алгоритм подписи. | При успешном завершении возвращает 0,в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **KC\_getCertFromXML ().**  Обеспечивает получение сертификата из XML. | unsigned long(\*KC\_getCertFromXML)(const char\* inXML, int inXMLLength, int inSignID, char \*outCert, int \*outCertLength); | Параметры:  [in] CHAR \* inXML — входные данные в формате XML;  [in] INT inSignID — идентификатор сертификата;  [out] CHAR \* outCert — указатель на начало сертификата. | При успешном завершении возвращает 0,в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **KC\_XMLFinalize ().**  Освобождает память и завершает работу библиотеки с модулями, отвечающие за парсинг, подпись и проверку данных в формате XML. | void (\*KC\_XMLFinalize)(void); | – | Не надо вызывать каждый раз при подписи. Можно только один раз после цикла подписания xml файлов. |
| **KC\_Finalize ().**  Освобождает ресурсы криптопровайдера KalkanCryptCOM и завершает работу библиотеки. | void (\*KC\_Finalize)(void); | – | — |
| **HashData().**  Хэширует данные. | unsigned long (\*HashData)(char \*algorithm, int flags, char \*inData, int inDataLength, unsigned char \*outData, int \*outDataLength); | [in] CHAR \* algorithm –алгоритм хэширования (строка “sha256” или “Gost34311\_95”);  [in] INT flags – флаги;  [in] CHAR \* inData – входные данные;  [out] CHAR \* outData – выходные хэшированные данные. | При успешном завершении возвращает 0,в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **SignHash().**  Подписывает входные хэшированные данные. | unsigned long(\*SignHash)(char \*alias, int flags, char \*inHash, int inHashLength, unsigned char \*outSign, int \*outSignLength); | [in] CHAR \* alias - label (alias) сертификата;  [in] INT flags – флаги;  [in] CHAR \* inData – входные хэшированные данные;  [out] CHAR \* outData – выходные данные. | При успешном завершении возвращает 0,в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **TSASetUrl().**  Установка адреса сервиса TSA. | void(\*KC\_TSASetUrl)(char \*tsaurl); | [in] CHAR \* tsaurl – адрес сервиса TSA | При успешном завершении возвращает 0,в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError.  Значение по умолчанию http://tsp.pki.gov.kz:80 |
| **KC\_GetTimeFromSig ()**  Получить время подписи. | unsigned long(\*KC\_GetTimeFromSig)(char \*inData, int inDataLength, int flags, int inSigId, time\_t \*outDateTime); | [in] CHAR \* inData – входные данные (подпись, в текущей версии только формата CAdES);  [in] INT flags – флаги  [in] INT inSigId – идентификатор подписи (в текущей версии не используется)  [out] time\_t \* outDateTime – выходные данные, время подписи UTC в Unix формате (отсчет от 01/01/1970) | При успешном завершении возвращает 0,в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **KC\_SetProxy ()**  Установить настройки прокси-сервера. | unsigned long (\*KC\_SetProxy)(int flags, char \*inProxyAddr, char \*inProxyPort, char \*inUser, char \*inPass); | [in] INT flags – флаги;  [in] CHAR \* inProxyAddr – адрес прокси-сервера;  [in] CHAR \* inPort – порт;  [in] CHAR \* inUserName – имя пользователя;  [in] CHAR \* inUserPass – пароль. | При успешном завершении возвращает 0,в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **ZipConSign()**  Подпись файлов с последующим размещением их в zip-контейнер | unsigned long (\*ZipConSign)(char \*alias, const char \*filePath, const char \*name, const char \*outDir, int flags); | [in] CHAR \* alias - label (alias) сертификата;  [in] CHAR \* inFiles – файлы, которые необходимо записать.(В конце каждого пути к файлу необходимо вставить вертикальную линию - «|», например: ***inFiles=*****“C:\\test\\1.pdf|C:\\test\\2.txt|”);**  [in] CHAR \* zipName – имя создаваемого архива;  [in] CHAR \* outDir – расположение создаваемого архива:  [in]INT flags – флаги. | При успешном завершении возвращает 0, в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **ZipConVerify()**  Проверка подписи электронных документов (zip-контейнер) | unsigned long (\*ZipConVerify)(char \*inZipFile, int flags, char \*outVerifyInfo, int \*outVerifyInfoLen); | [in] CHAR \* inZipFile – имя ZIP-файла  [in] INT flags – флаги;  [out] CHAR \* outVerifyInfo - выходная подробная информация о результате проверки подписи. | При успешном завершении возвращает 0, в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **SignWSSE()**  Подпись SOAP-сообщения по спецификации WS-Security | unsigned long(\*SignWSSE)(char \*alias, unsigned long flags, char \*inData, int inDataLength, unsigned char \*outSign, int \*outSignLength, char \*signNodeId); | [in] CHAR \* alias — label (alias) сертификата;  [in] INT flags — флаги;  [in] BYTE\* inData - входные данные;  [in] CHAR \* signNodeId — идентификатор тэга, который необходимо подписать;  [out] CHAR \* outSign — выходные данные (подписи). | При успешном завершении возвращает 0, в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **KC\_GetCertFromCMS ().**  Обеспечивает получение сертификата из XML. | unsigned long(\*KC\_GetTimeFromSig)(char \*inData, int inDataLength, int flags, int inSigId, time\_t \*outDateTime); | Параметры:  [in] CHAR \* inCMS — входные данные в формате CMS;  [in] INT inSignID — идентификатор сертификата;  [out] CHAR \* outCert — указатель на начало сертификата. | При успешном завершении возвращает 0,в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **KC\_getCertFromZipFile ().**  Обеспечивает получение сертификата из XML. | unsigned long(\*KC\_getCertFromZipFile)(char\* inZipFile, int flags, int inSignID, char \*outCert, int \*outCertLength); | Параметры:  [in] CHAR \* inZipFile — входной файл в формате ZIP;  [in] INT inSignID — идентификатор сертификата;  [out] CHAR \* outCert — указатель на начало сертификата. | При успешном завершении возвращает 0,в противном случае – 1. Код ошибки и подробный текст можно получить с помощью функции GetLastError. |
| **KC\_InitDebug ().**  Включает запись лога работы библиотеки на файл | void(\*KC\_InitDebug)(void); | – | — |