ООО «ВАЛИДАТА»

УТВЕРЖДЕН
ВАМБ.00060-06–ЛУ

СРЕДСТВО КРИПТОГРАФИЧЕСКОЙ ЗАЩИТЫ ИНФОРМАЦИИ «ВАЛИДАТА CSP» ВЕРСИЯ 6

Руководство программиста

ВАМБ.00060-06 33 01

2020

Аннотация

Данный документ содержит описание библиотеки функций электронной подписи и шифрования, библиотеки поддержки считывателей ключей электронной подписи и/или закрытых ключей шифрования и датчиков случайных чисел из состава программного комплекса ВАМБ.00060-06 «СКЗИ «Валидата CSP» версия 6» (далее — СКЗИ «Валидата CSP»).

Документ предназначен для разработчиков прикладных программ (внешних по отношению к СКЗИ «Валидата CSP»).

Содержание

1 НАЗНАЧЕНИЕ И УСЛОВИЯ ПРИМЕНЕНИЯ

Библиотека функций электронной подписи (ЭП) и шифрования (далее - библиотека криптографического провайдера), вызываемая через **Microsoft CryptoAPI**, является составной частью СКЗИ «Валидата CSP» и предназначена для:

– вычисления и проверки ЭП в соответствии с ГОСТ Р 34.10-2012 (для ключей ЭП длиной 256 и 512 бит);

– проверки ЭП в соответствии с ГОСТ Р 34.10-2001;

– выполнения зашифрования и расшифрования данных в соответствии с ГОСТ Р 34.12-2015/ГОСТ Р 34.13-2015 (Магма, Кузнечик) и ГОСТ 28147-89;

– вычисления имитовставки данных в соответствии с ГОСТ Р 34.12-2015/ГОСТ Р 34.13-2015 (Магма, Кузнечик) и ГОСТ 28147-89;

– вычисления ключа парной связи Диффи-Хелмана с использованием пар закрытых и открытых ключей шифрования, созданных в соответствии с ГОСТ Р 34.10-2012 (для закрытых ключей шифрования длиной 256 и 512 бит);

– вычисления хэш-значений данных в соответствии с ГОСТ Р 34.11-2012 (для хэш-значений длиной 256 и 512 бит);

– вычисления хэш-значений данных в соответствии с ГОСТ Р 34.11-94;

– формирования (генерации) ключей ЭП и закрытых ключей шифрования длиной 256 и 512 бит в соответствии с ГОСТ Р 34.10-2012;

– вычисления ключей проверки ЭП и открытых ключей шифрования в соответствии с ГОСТ Р 34.10-2012 (для ключей ЭП и закрытых ключей шифрования длиной 256 и 512 бит);

– выработки случайного числа заданной длины.

*Примечание - В данном документе термин **закрытый ключ** обозначает ключ ЭП и/или закрытый ключ шифрования, а термин **открытый ключ** обозначает ключ проверки ЭП и/или открытый ключ шифрования.*

2 ИСПОЛЬЗОВАНИЕ БИБЛИОТЕКИ

2.1 Общее описание

Использование библиотеки криптографического провайдера производится посредством выполнения вызовов низкоуровневых функций прикладного программного интерфейса **Microsoft CryptoAPI** из системной разделяемой библиотеки **cryptsp.dll** (или **advapi32.dll**). При этом отпадает необходимость вызова функций библиотеки криптографического провайдера напрямую.

В состав инструментария разработчика криптографического провайдера входят следующие файлы заголовков:

– **vdcsp.h** - файл заголовков, содержащий описание всех необходимых констант и структур криптопровайдера;

– **vderror.h** - файл заголовков, содержащий список кодов ошибок.

Данные файлы заголовков необходимо включать после файла заголовков **wincrypt.h**, входящего в состав **Microsoft Windows Development Kit**. Также необходимо включить в список библиотек для выполнения компоновки системную библиотеку **cryptsp.lib** (или **advapi32.lib**), входящую в состав **Microsoft Windows Development Kit**.

2.2 Описание констант

Посредством использования библиотеки доступны два криптографических провайдера со следующими параметрами (Таблица 1, Таблица 2).

Таблица 1 – Параметры криптографического провайдера по ГОСТ Р 34.10-2012

Параметр	Переменная	Значение
Название криптографического провайдера	**GR3410_2012_PROV_A**, **GR3410_2012_PROV_W**	Validata GOST R 34.10-2012 CSP
Тип криптографического провайдера	**PROV_GOST_2012_DH**	80

Таблица 2 – Параметры криптографического провайдера по ГОСТ Р 34.10-2001

Параметр	Переменная	Значение
Название криптографического провайдера	**GR3410_2001_PROV_A**, **GR3410_2001_PROV_W**	Validata GOST R 34.10-2001 CSP
Тип криптографического провайдера	**PROV_GOST_2001_DH**	75

Константы, описывающие идентификаторы криптографических алгоритмов, поддерживаемых библиотекой, приведены в таблице 3.

Таблица 3 – Идентификаторы криптографических алгоритмов

Алгоритм	**Описание**
CALG_GR3411_12_256	Вычисление хэш-значений данных в соответствии с ГОСТ Р 34.11-2012 (для хэш-значений длиной 256 бит)
CALG_GR3411_12_512	Вычисление хэш-значений данных в соответствии с ГОСТ Р 34.11-2012 (для хэш-значений длиной 512 бит)
CALG_GR3411	Вычисление хэш-значений данных в соответствии с ГОСТ Р 34.11-94. Поддерживаются наборы параметров из ГОСТ и A из RFC 4357
CALG_GR3412_15MG	Зашифрование и расшифрование данных в соответствии с ГОСТ Р 34.12-2015/ГОСТ Р 34.13-2015 (Магма). В режиме гаммирования CTR-ACPKM соответствует разделу 4.1 документа **Р 1323565.1.017—2018**
CALG_GR3412_15MG_-MAC	Вычисление имитовставки данных в соответствии с ГОСТ Р 34.12-2015/ГОСТ Р 34.13-2015 (Магма)
CALG_GR3412_15GH	Зашифрование и расшифрование данных в соответствии с ГОСТ Р 34.12-2015/ГОСТ Р 34.13-2015 (Кузнечик). В режиме гаммирования CTR-ACPKM соответствует разделу 4.1 документа **Р 1323565.1.017—2018**
CALG_GR3412_15GH_-MAC	Вычисление имитовставки данных в соответствии с ГОСТ Р 34.12-2015/ГОСТ Р 34.13-2015 (Кузнечик)
CALG_G28147, **CALG_TLS1_ENC_KEY**	Зашифрование и расшифрование данных в соответствии с ГОСТ 28147-89. Поддерживаются наборы параметров из ГОСТ, A, B, C, D из RFC 4357 и Z (ISO/IEC 18033-3) из рекомендаций Технического комитета №26
CALG_G28147_IMIT, **CALG_TLS1_MAC_KEY**	Вычисление имитовставки данных в соответствии с ГОСТ 28147-89. Поддерживаются наборы параметров из ГОСТ, A, B, C, D из RFC 4357 и Z (ISO/IEC 18033-3) из рекомендаций Технического комитета №26

Алгоритм	Описание
CALG_GR3410EL_12_256	Вычисление и проверка ЭП в соответствии с ГОСТ Р 34.10-2012 (для ключей ЭП длиной 256 бит). Поддерживаются наборы параметров из ГОСТ, A, B, C из RFC 4357 и A, B, C, D из документа **МР 26.2.002-2018**
CALG_GR3410EL_12_512	Вычисление и проверка ЭП в соответствии с ГОСТ Р 34.10-2012 (для ключей ЭП длиной 512 бит). Поддерживаются наборы параметров из ГОСТ, и A, B, C из из документа **МР 26.2.002-2018**
CALG_DH_EL_12_256_SF	Вычисление ключа парной связи Диффи-Хелмана с использованием пар закрытых и открытых ключей шифрования, созданных в соответствии с ГОСТ Р 34.10-2012 (для закрытых ключей шифрования длиной 256 бит). Поддерживаются наборы параметров из ГОСТ, A, B, C из RFC 4357 и A, B, C, D из документа **МР 26.2.002-2018**
CALG_DH_EL_12_512_SF	Вычисление ключа парной связи Диффи-Хелмана с использованием пар закрытых и открытых ключей шифрования, созданных в соответствии с ГОСТ Р 34.10-2012 (для закрытых ключей шифрования длиной 512 бит). Поддерживаются наборы параметров из ГОСТ, и A, B, C из документа **МР 26.2.002-2018**
CALG_DH_EL_12_256_-EPHEM	Вычисление эфемерного ключа парной связи Диффи-Хелмана с использованием пар закрытых и открытых ключей шифрования, созданных в соответствии с ГОСТ Р 34.10-2012 (для закрытых ключей шифрования длиной 256 бит). Поддерживаются наборы параметров из ГОСТ, A, B, C из RFC 4357 и A, B, C, D из документа **МР 26.2.002-2018**
CALG_DH_EL_12_512_-EPHEM	Вычисление эфемерного ключа парной связи Диффи-Хелмана с использованием пар закрытых и открытых ключей шифрования, созданных в соответствии с ГОСТ Р 34.10-2012 (для закрытых ключей шифрования длиной 512 бит). Поддерживаются наборы параметров из ГОСТ, и A, B, C из документа **МР 26.2.002-2018**

Алгоритм	Описание
CALG_GR3410EL, CALG_DH_EL_SF	Проверка ЭП в соответствии с ГОСТ Р 34.10-2001. Поддерживаются наборы параметров из ГОСТ и A, B, C из RFC 4357
CALG_15MG_EXPORT	Выполнение экспорта и импорта ключа шифрования или сеансового ключа посредством вычисления имитовставки и шифрования по ГОСТ Р 34.12-2015/ГОСТ Р 34.13-2015 (Магма). Соответствует разделу 5 документа **Р 1323565.1.017—2018**
CALG_15GH_EXPORT	Выполнение экспорта и импорта ключа посредством вычисления имитовставки и шифрования по ГОСТ Р 34.12-2015/ГОСТ Р 34.13-2015 (Кузнечик). Соответствует разделу 5 документа **Р 1323565.1.017—2018**
CALG_KDF_EXPORT	Выполнение экспорта и импорта ключа посредством хэширования по ГОСТ Р 34.11-2012 (для хэш-значений длиной 256 бит), а также вычисления имитовставки и шифрования по ГОСТ 28147-89 в соответствии с разделом 4.6 документа **Р 50.1.113-2016**. Поддерживаются наборы параметров из ГОСТ, A, B, C, D из RFC 4357 и Z (ISO/IEC 18033-3) из рекомендаций Технического комитета №26
CALG_PRO_EXPORT	Выполнение экспорта и импорта ключа с усложнением посредством вычисления имитовставки и шифрования по ГОСТ 28147-89. Поддерживаются наборы параметров из ГОСТ, A, B, C, D из RFC 4357 и Z (ISO/IEC 18033-3) из рекомендаций Технического комитета №26
CALG_SIMPLE_EXPORT	Выполнение экспорта и импорта ключа без усложнения посредством вычисления имитовставки и шифрования по ГОСТ 28147-89. Поддерживаются наборы параметров из ГОСТ, A, B, C, D из RFC 4357 и Z (ISO/IEC 18033-3) из рекомендаций Технического комитета №26
CALG_TLS1_MASTER	Мастер-ключ протокола TLS, используется для формирования ключей шифрования и вычисления имитовставки клиентской и серверной сторон с использованием псевдо-случайной функции

Алгоритм	Описание
CALG_TLS1_MASTER_-HASH	Вычисление псевдо-случайной функции на основании мастер-ключа протокола TLS с использованием алгоритма хэширования по ГОСТ Р 34.11-2012 (для хэш-значений длиной 256 бит). Соответствует разделу 4.2.1 документа **Р 50.1.113-2016**

3 ИСПОЛЬЗОВАНИЕ НИЗКОУРОВНЕВЫХ ФУНКЦИЙ MICROSOFT CRYPTO API

3.1 Функция получения дескриптора криптопровайдера - CryptAcquireContext

Данная функция должна быть вызвана до использования остальных функций библиотеки.

Прототип:

```
BOOL WINAPI CryptAcquireContext(
  HCRYPTPROV*      phProv,
  CHAR*            szContainer,
  CHAR*            szProvider,
  DWORD            dwProvType,
  DWORD            dwFlags
);
```

Параметры:

phProv

[out] Адрес, в который функция копирует созданный дескриптор криптопровайдера. По завершению работы с дескриптором его следует удалить вызовом функции CryptReleaseContext.

szContainer

[in] Заканчивающаяся нулем строка размером не более 255 символов, содержащая имя ключевого контейнера, размер которого не должен превышать 64 символов (без учета нуля). В случае если в параметре dwFlags установлен флаг CRYPT_VERIFYCONTEXT, то значение данного параметра должно быть NULL. В остальных режимах создания дескриптора криптопровайдера данная строка должна содержать имя ключевого контейнера (за исключением случая использования флага CRYPT_SELECT_CONTAINER).

Данная строка может дополнительно содержать имя ключевого устройства, используемое при операциях с ключевым(и) контейнером(ами):

Считыватель	**Строка ключевого устройства и контейнера**
Съемный диск	`\\.\02_REMOVABLE_F:\1234ABCDEF01`
ПАК Соболь	`\\.\03_SOBOL\1234ABCDEF01`
ПАК Аккорд	`\\.\04_ACCORD\1234ABCDEF01`
Реестр Windows	`\\.\06_REGISTRY\1234ABCDEF01`

Считыватель ruToken	\\.\07_RUTOKEN_Aktiv Co. ruToken 0\1234ABCDEF01 \\.\Aktiv Co. ruToken 0\1234ABCDEF01
Считыватель eToken	\\.\11_ETOKEN_Aladdin Token JC 0\1234ABCDEF01 \\.\Aladdin Token JC 0\1234ABCDEF01
Считыватель jaCarta	\\.\13_JACARTA_ARDS ZAO JaCarta LT 0\1234ABCDEF01
Считыватель vdToken (ФКН)	\\.\16_VDTOKEN_FKN_Validata vdToken 0\1234ABCDEF01 \\.\Validata vdToken 0\1234ABCDEF01
Считыватель vdToken	\\.\17_VDTOKEN_Validata vdToken 0\1234ABCDEF01 \\.\Validata vdToken 0\1234ABCDEF01
Считыватель Dallas	\\.\21_DALLAS\1234ABCDEF01
ПАК Secret Net	\\.\23_SEC_NET\1234ABCDEF01

szProvider

[in] Имя криптографического провайдера. Должно быть равно значению константы GR3410_2012_PROV_A/GR3410_2012_PROV_W (для провайдера по ГОСТ Р 34.10-2012) или GR3410_2001_PROV_A/GR3410_2001_PROV_W (для провайдера по ГОСТ Р 34.10-2001).

dwProvType

[in] Тип криптографического провайдера. Должен быть равен значению константы PROV_GOST_2012_DH (для провайдера по ГОСТ Р 34.10-2012) или PROV_GOST_2001_DH (для провайдера по ГОСТ Р 34.10-2001).

dwFlags

[in] Флаги, используемые для вызова функции. Флаги CRYPT_VERIFYCONTEXT, CRYPT_NEWKEYSET и CRYPT_DELETEKEYSET являются взаимоисключающими.

Значение dwFlags	Описание
CRYPT_VERIFYCONTEXT	Дескриптор криптопровайдера создаётся в режиме проверки. Приложение не имеет доступа к закрытому ключу. Данный режим работы криптопровайдера может быть использован для вычисления хэш-значений, проверки ЭП, получения параметров криптопровайдера (например, информации о поддерживаемых алгоритмах), выработки случайных чисел.
CRYPT_NEWKEYSET	Дескриптор криптопровайдера создается в режиме создания нового закрытого ключа. Имя создаваемого ключевого контейнера задаётся параметром **szContainer**. Для сохранения ключевого контейнера в устройстве хранения необходимо создать хотя бы одну ключевую пару, для чего необходимо использовать функцию CryptGenKey.
CRYPT_DELETEKEYSET	Дескриптор криптопровайдера создается в режиме удаления закрытого ключа. Имя удаляемого ключевого контейнера задаётся параметром **szContainer**. В отличие от остальных режимов создания дескриптора криптопровайдера, вызов функции с данным значением флага не создаёт действующий дескриптор. Создаваемый дескриптор существует только во время выполнения функции. На выходе из функции значение возвращаемое в параметре **phProv** не определено, поэтому функция освобождения дескриптора CryptReleaseContext **не должна вызываться**.

CRYPT_MACHINE_-KEYSET	Флаг определяет принадлежность ключевого контейнера. В тех случаях, когда данный флаг установлен, считается, что ключевой контейнер принадлежит системе (компьютеру), а не конкретному пользователю. Данный флаг применяется в случаях использования дескриптора криптопровайдера приложениями, не ассоциированными с конкретным пользователем. Следует учитывать, что если ключевой контейнер создан с использованием данного флага, то он должен применяться во всех вызовах CryptAcquireContext, относящихся к этому контейнеру.
CRYPT_SILENT	Флаг, запрещающий отображение графического интерфейса при выполнении операций криптопровайдером. Флаг можно комбинировать с любым режимом использования дескриптора криптопровайдера. В случае, если для выполнения одной из операций криптопровайдера требуется отображение графического интерфейса, а при создании дескриптора использовался данный флаг, то вызов функции устанавливает специальный код ошибки - NTE_SILENT_CONTEXT.
CRYPT_SELECT_-CONTAINER	Флаг позволяет вместо задания конкретного имени ключевого контейнера использовать диалоговое окно выбора ключевого контейнера из списка существующих. Данный флаг не может быть использован одновременно с любым из флагов CRYPT_-VERIFYCONTEXT, CRYPT_NEWKEYSET или CRYPT_SILENT.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.2 Функция получения параметров криптопровайдера - CryptGetProvParam

Прототип:

```
BOOL WINAPI CryptGetProvParam(
  HCRYPTPROV   hProv,
  DWORD        dwParam,
  BYTE*        pbData,
  DWORD*       pdwDataLen,
  DWORD        dwFlags
);
```

Параметры:

hProv

[in] Дескриптор криптопровайдера. Дескриптор криптопровайдера должен быть предварительно получен вызовом функции CryptAcquireContext.

dwParam

[in] Тип запрашиваемого параметра.

Тип параметра определяет тип используемых данных в параметре функции pbData.

Тип параметра	**Описание**
PP_NAME	Параметр позволяет получить имя криптопровайдера в виде заканчивающейся нулем строки. Возвращаемая строка идентична строке, передаваемой в параметре **szProvider** функции CryptoAPI CryptAcquireContext, для задания используемого криптопровайдера. В параметре **pbData** необходимо передавать указатель на массив типа BYTE (BYTE*).
PP_VERSION	Параметр позволяет получить версию криптопровайдера. Младший значащий байт содержит младший номер версии, старший байт - основной номер версии. В параметре **pbData** необходимо передавать указатель на переменную типа DWORD.
PP_PROVTYPE	Параметр позволяет получить тип криптопровайдера. В параметре **pbData** необходимо передавать указатель на переменную типа DWORD.
PP_CONTAINER PP_UNIQUE_CONTAINER	Параметр позволяет получить имя загруженного в настоящий момент ключевого контейнера в виде заканчивающейся нулем строки. В параметре **pbData** необходимо передавать указатель на массив типа BYTE (BYTE*).
PP_IMPTYPE	Параметр позволяет получить тип реализации криптопровайдера. Возвращаемые данные могут содержать одно из предопределённых значений: – CRYPT_IMPL_HARDWARE - аппаратно; – CRYPT_IMPL_SOFTWARE - программно; – CRYPT_IMPL_MIXED - аппаратно-программно; – CRYPT_IMPL_UNKNOWN - неизвестно. В параметре **pbData** необходимо передавать указатель на переменную типа DWORD.

PP_KEYSPEC	Параметр позволяет получить спецификаторы ключей поддерживаемых криптопровайдером. Под спецификаторами ключей в данном случае понимаются типы ключей, которые могут быть сформированы криптопровайдером. В общем случае криптопровайдер может поддерживать ключи двух типов, определяемых как AT_SIGNATURE или AT_KEYEXCHANGE. В параметре **pbData** необходимо передавать указатель на переменную типа DWORD.
PP_KEYSET_TYPE	Параметр позволяет узнать кому принадлежит загруженный ключевой контейнер: пользователю или системе. Возвращаемое значение определяется использованием вспомогательного флага CRYPT_MACHINE_KEYSET при создании дескриптора криптопровайдера функцией CryptAcquireContext. В случае, если данный флаг использовался, то он будет возвращен в параметре pbData в качестве выходного значения. В противном случае (контейнер принадлежит пользователю) возвращается 0. В параметре **pbData** необходимо передавать указатель на переменную типа DWORD.
PP_SIG_KEYSIZE_INC	Параметр позволяет определить приращение размера ключей ЭП (AT_SIGNATURE) в битах. Полученное значение используется совместно с данными возвращаемыми при получении параметра PP_ENUMALGS_EX для определения допустимых длин ключей подписи. В параметре **pbData** необходимо передавать указатель на переменную типа DWORD.
PP_KEYX_KEYSIZE_INC	Параметр позволяет определить приращение размера закрытых ключей шифрования (AT_KEYEXCHANGE) в битах. Полученное значение используется совместно с данными возвращаемыми при получении параметра PP_ENUMALGS_EX для определения допустимых длин ключей подписи. В параметре **pbData** необходимо передавать указатель на переменную типа DWORD.

PP_USE_HARDWARE_RNG	Параметр позволяет определить, используется ли для генерации случайных чисел аппаратный датчик случайных чисел (ДСЧ). При использовании аппаратного ДСЧ функция успешно завершается, в противном случае возвращается код ошибки.
PP_ENUMALGS	Параметр позволяет получить информацию об алгоритмах, поддерживаемых криптопровайдером. Информация об одном алгоритме возвращается в виде структуры PROV_ENUMALGS. Для получения информации об алгоритмах, поддерживаемых криптопровайдером, необходимо многократно вызывать функцию с этим параметром. Для начала перечисления алгоритмов (получения первого в списке) необходимо в параметре dwFlags передать флаг CRYPT_FIRST. Размер данных, возвращаемый в параметре pdwDataLen при вызове функции с указанным флагом, содержит максимальный размер для любого из элементов списка. При последующих вызовах значение параметра dwFlags должно быть равно CRYPT_NEXT. Признаком завершения перечисления элементов списка (в данном случае информации об алгоритмах) является установка функцией кода ошибки ERROR_NO_MORE_ITEMS (достигнут конец списка). Вызов функции с данным параметром нельзя производить одновременно из нескольких потоков на одном и том же дескрипторе. В параметре **pbData** необходимо передавать указатель на структуру типа PROV_ENUMALGS (PROV_ENUMALGS*).

PP_ENUMCONTAINERS	Параметр позволяет получить информацию о ключевых контейнерах, находящихся на ключевых носителях. Информация об одном ключевом контейнере возвращается в виде текстовой строки. Для получения информации о ключевых контейнерах необходимо многократно вызывать функцию с этим параметром. Для начала перечисления ключевых контейнеров (получения первого в списке) необходимо в параметре dwFlags передать флаг CRYPT_FIRST. Размер данных, возвращаемый в параметре pdwDataLen при вызове функции с указанным флагом, содержит максимальный размер для любого из элементов списка. При последующих вызовах значение параметра dwFlags должно быть равно CRYPT_NEXT. Признаком завершения перечисления элементов списка (в данном случае информации о ключевых контейнерах) является установка функцией кода ошибки ERROR_NO_MORE_ITEMS (достигнут конец списка). Вызов функции с данным параметром нельзя производить одновременно из нескольких потоков на одном и том же дескрипторе. В параметре **pbData** необходимо передавать указатель на текстовую строку LPSTR.

PP_ENUMALGS_EX	Параметр позволяет получить расширенную информацию об алгоритмах, поддерживаемых криптопровайдером. Информация об одном алгоритме возвращается в виде структуры PROV_NUMALGS_EX. Перечисление списка алгоритмов работает так же как при вызове функции с параметром PP_-ENUMALGS. В параметре **pbData** необходимо передавать указатель на структуру типа PROV_ENUMALGS_EX (PROV_ENUMALGS_EX*).
PP_USER_CERTSTORE	Параметр позволяет получить контекст хранилища сертификатов, находящихся на ключевых носителях типа смарт-карта. По завершении использования контекста его необходимо освободить с помощью функции CertCloseStore. В параметре **pbData** необходимо передавать указатель на контекст хранилища сертификатов HCERTSTORE.
PP_SMARTCARD_READER	Параметр позволяет получить имя считывателя смарт-карты, которое использовалось как часть имени ключевого контейнера при инициализации контекста. В параметре **pbData** необходимо передавать указатель на текстовую строку LPSTR.
PP_SECURITY_LEVEL	Параметр позволяет получить уровень обеспечения безопасности СКЗИ в виде константы PROV_KC_LEVEL_1 для уровня КС1 или константы PROV_KC_LEVEL_2 для уровня КС2. В параметре **pbData** необходимо передавать указатель на переменную типа DWORD.
PP_VERSION_EX	Параметр позволяет получить расширенную версию криптопровайдера. В параметре **pbData** необходимо передавать указатель на структуру типа VD_CSP_VERSION_EX (VD_CSP_VERSION_EX*).
PP_VERSION_EXTEND	Параметр позволяет получить расширенную версию криптопровайдера с дополнительной информацией о конкретной сборке. В параметре **pbData** необходимо передавать указатель на структуру типа VD_VERSION_EXTEND (VD_VERSION_EXTEND*).

PP_ENUMCACHEDKEYS	Параметр позволяет получить информацию об именах загруженных закрытых ключей, находящихся в памяти вызывающего процесса. Информация об имени загруженного закрытого ключа возвращается в виде текстовой строки. Для получения информации об именах загруженных закрытых ключей необходимо многократно вызывать функцию с этим параметром. Для начала перечисления имен загруженных закрытых ключей (получения первого в списке) необходимо в параметре dwFlags передать флаг CRYPT_FIRST. Размер данных, возвращаемый в параметре pdwDataLen при вызове функции с указанным флагом, содержит максимальный размер для любого из элементов списка. При последующих вызовах значение параметра dwFlags должно быть равно CRYPT_NEXT. Признаком завершения перечисления элементов списка (в данном случае информации об именах загруженных закрытых ключей) является установка функцией кода ошибки ERROR_NO_MORE_ITEMS (достигнут конец списка). Вызов функции с данным параметром нельзя производить одновременно из нескольких потоков на одном и том же дескрипторе. В параметре **pbData** необходимо передавать указатель на текстовую строку LPSTR.

PP_GET_ERROR_TEXT	Параметр позволяет получить текстовое описание ошибки криптопровайдера по ее коду в виде заканчивающейся нулем строки. В параметре **pbData** необходимо передавать указатель на текстовую строку LPSTR длиной не менее 264 байт. Первые 4 байта указанной строки должны содержать код ошибки.

pbData

[out] Указатель на буфер, в который копируются запрошенные данные. Тип возвращаемых данных зависит от значения **dwParam**. В случае, если **pbData** равен NULL, то копирование данных не производится. Вместо этого, в параметре **pdwDataLen** возвращается требуемый размер буфера в байтах.

pdwDataLen

[in, out] Адрес, указывающий на значение типа DWORD. Данный параметр на входе содержит размер буфера, определяемого параметром **pbData**, в байтах. На выходе, для большинства используемых значений параметра **dwParam**, в данном параметре содержится размер, в байтах, фактически скопированных в буфер данных. В случае, если в параметре **pbData** передан NULL, то в данном параметре на выходе возвращается размер памяти требуемой для копирования запрошенных данных.

В случае, если размер буфера в параметре **pbData** недостаточно велик для копирования в него требуемых данных, устанавливается код ошибки ERROR_MORE_DATA.

Следует отметить, что для перечисляемых типов (PP_ENUMALGS, PP_ENUMCONTAINERS, PP_ENUMALGS_EX, PP_ENUMCACHEDKEYS) в данном параметре возвращается не размер текущего элемента, а максимально возможный размер. Для корректной установки размера для перечисляемых параметров должен быть установлен флаг CRYPT_FIRST.

dwFlags

[in] Допустимо использование следующих флагов:

– CRYPT_FIRST. Данный флаг используется в случае, если значение параметра **dwParam** определяет один из перечисляемых типов: PP_ENUMALGS, PP_ENUMCONTAINERS, PP_ENUMALGS_EX, PP_ENUMCACHEDKEYS. В этом случае возвращается первый элемент соответствующего списка;

– CRYPT_NEXT. Данный флаг используется в случае, если значение параметра **dwParam** определяет один из перечисляемых типов: PP_ENUMALGS, PP_ENUMCONTAINERS, PP_ENUMALGS_EX, PP_ENUMCACHEDKEYS. В этом случае возвращается следующий элемент соответствующего списка.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.3 Функция установки параметров криптопровайдера - CryptSetProvParam

Прототип:

```
BOOL WINAPI CryptSetProvParam(
  HCRYPTPROV   hProv,
  DWORD        dwParam,
  BYTE*        pbData,
  DWORD        dwFlags
);
```

Параметры:

hProv

[in] Дескриптор криптопровайдера, в контексте которого производится установка параметра. Дескриптор криптопровайдера должен быть предварительно получен вызовом функции CryptAcquireContext.

dwParam

[in] Тип устанавливаемого параметра. Тип параметра определяет тип и формат данных передаваемых в параметре **pbData**.

Тип параметра	**Описание и используемый тип параметра pbData**
PP_CLIENT_HWND	Данный параметр позволяет задавать дескриптор окна, который криптопровайдер должен использовать в качестве родительского для всех создаваемых им интерфейсных окон. Параметр **pbData** должен содержать указатель на HWND.
PP_SIGNATURE_PIN PP_KEYEXCHANGE_PIN	Данные параметры позволяют задавать ПИН-код соответственно ключа ЭП и закрытого ключа шифрования для ключевого носителя типа смарт-карта. Параметр **pbData** должен содержать указатель на текстовую строку LPSTR.
PP_CARRIER_SETPIN	Данный параметр позволяет вывести диалоговый интерфейс смены ПИН-кода ключевого носителя. В настоящее время поддерживается смена ПИН-кода ключевых носителей vdToken (ФКН) и vdToken. Параметр **pbData** должен быть равен **NULL**.
PP_CNG_CERTSTORES	Данный параметр позволяет задавать тип привязки закрытых ключей к сертификатам (CSP или CNG) при вызове функции CryptGetProvParam() с параметром PP_USER_CERTSTORE (по умолчанию используется тип привязки CSP). Параметр **pbData** должен содержать указатель на переменную типа BOOL.

PP_CARRIER_FORMAT	Данный параметр позволяет вывести диалоговый интерфейс форматирования ключевого носителя. В настоящее время поддерживается форматирование ключевых носителей vdToken (ФКН), vdToken и Touch Memory через считыватели ПАК Соболь и СЗИ Secret Net. Параметр **pbData** должен быть равен **NULL**.
PP_SILENT_CONTEXT	Данный параметр позволяет принудительно включить запрет отображения графического интерфейса при выполнении операций криптопровайдером. Параметр **pbData** не используется.

pbData

[in] Указатель на буфер, содержащий устанавливаемое значение. Тип, формат и размер буфера определяется значением параметра dwParam.

dwFlags

[in] При установке параметров криптопровайдера поле флагов не используется и должно быть равно 0.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.4 Функция освобождения дескриптора криптопровайдера - CryptReleaseContext

Данная функция должна вызываться для освобождения дескрипторов криптопровайдера, полученных посредством вызова функции CryptAcquireContext.

Прототип:

```
BOOL WINAPI CryptReleaseContext(
  HCRYPTPROV   hProv,
  DWORD        dwFlags
);
```

Параметры:

hProv

[in] Удаляемый дескриптор криптопровайдера.

dwFlags

[in] Флаги (в настоящее время не используются и должны быть равны 0).

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.5 Функция формирования сессионного ключа по результату вычисления хэш-функции - CryptDeriveKey

Прототип:

```
BOOL WINAPI CryptDeriveKey(
  HCRYPTPROV   hProv,
  ALG_ID       Algid,
  HCRYPTHASH   hBaseData,
  DWORD        dwFlags,
  HCRYPTKEY*   phKey
);
```

Параметры:

hProv

[in] Дескриптор криптопровайдера. Дескриптор криптопровайдера должен быть предварительно получен вызовом функции CryptAcquireContext.

Algid

[in] Идентификатор алгоритма шифрования, для которого необходимо сформировать сессионный ключ. В качестве значения данного параметра должен использоваться идентификатор алгоритма шифрования CALG_GR3412_15MG, CALG_GR3412_15GH или CALG_G28147 для хэша с идентификатором алгоритма CALG_GR3411_12_256, или же идентификатор алгоритма шифрования CALG_TLS1_ENC_KEY или вычисления имитовставки CALG_TLS1_MAC_KEY для хэша с идентификатором алгоритма CALG_TLS1_MASTER_HASH.

hBaseData

[in] Дескриптор объекта хэширования, используемого для формирования ключа. Данный дескриптор должен быть создан при вызове функции CryptCreateHash с использованием идентификатора алгоритма хэширования CALG_GR3411_12_256 или CALG_TLS1_MASTER_HASH.

dwFlags

[in] Флаги, позволяющие устанавливать некоторые атрибуты создаваемого сессионного ключа.

Значение	**Описание**
CRYPT_EXPORTABLE	В случае, если данный флаг установлен, формируемый сессионный ключ может быть экспортирован из криптопровайдера в ключевой блоб функцией CryptExportKey. В противном случае создаваемый ключ существует только до момента уничтожения дескриптора ключа.
CRYPT_SERVER	В случае, если данный флаг установлен, формируются ключи шифрования и вычисления имитовставки серверной стороны протокола TLS. В противном случае формируются ключи шифрования и вычисления имитовставки клиентской стороны протокола TLS. Данный флаг может использоваться только с сессионными ключами с идентификатором алгоритма CALG_TLS1_ENC_KEY или CALG_TLS1_MAC_KEY.

phKey

[out] Адрес, по которому функция записывает дескриптор созданного ключа. Освобождение полученного дескриптора должно производиться через вызов функции CryptDestroyKey.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае

возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.6 Функция освобождения дескриптора ключа - CryptDestroyKey

Приложения верхнего уровня должны удалять дескрипторы ключей по завершении их использования с помощью данной функции.

Прототип:

```
BOOL WINAPI CryptDestroyKey(
  HCRYPTKEY    hKey
);
```

Параметры:

hKey

[in] Дескриптор удаляемого ключа.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.7 Функция экспорта криптографических ключей из криптопровайдера в ключевой блоб (массив байт) - CryptExportKey

С помощью данной функции можно экспортировать следующие типы криптографических ключей: сессионные ключи, открытые ключи, закрытые ключи. Экспортируемые ключи сохраняются в виде ключевых блобов (массивов байт), формат и содержание которых зависят от типа экспортируемого ключа.

Прототип:

```
BOOL WINAPI CryptExportKey(
  HCRYPTKEY    hKey,
  HCRYPTKEY    hPubKey,
  DWORD        dwBlobType,
  DWORD        dwFlags,
  BYTE*        pbData,
  DWORD*       pdwDataLen
);
```

Параметры:

hKey

[in] Дескриптор экспортируемого ключа.

hPubKey

[in] Дескриптор ключа, на котором необходимо выполнить экспорт. Ключ, определяемый указанным дескриптором, используется для шифрования экспортируемого ключа. Данный параметр имеет значение только при экспорте сессионных и закрытых ключей.

dwBlobType

[in] Тип ключевого блоба, создаваемого для экспорта ключа.

Тип блоба	**Описание**
SIMPLEBLOB	Данный тип блоба используется при экспорте сессионных ключей в зашифрованном виде. Параметр **hKey** должен содержать дескриптор экспортируемого сессионного ключа с одним из следующих алгоритмов: CALG_GR3412_15MG, CALG_GR3412_15GH, CALG_G28147, CALG_TLS1_MASTER, CALG_TLS1_ENC_KEY, CALG_TLS1_MAC_KEY. Параметр **hPubKey** должен содержать дескриптор ключа, на котором будет выполняться зашифрование экспортируемого сессионного ключа. При выполнении экспорта сессионного ключа с алгоритмом CALG_GR3412_15MG, CALG_GR3412_15GH, CALG_TLS1_MASTER, CALG_TLS1_ENC_KEY или CALG_TLS1_MAC_KEY алгоритм ключа экспорта может принимать одно из следующих значений: CALG_15MG_EXPORT, CALG_15GH_EXPORT, CALG_KDF_EXPORT, CALG_PRO_EXPORT, CALG_SIMPLE_EXPORT. При выполнении экспорта сессионного ключа с алгоритмом CALG_G28147 алгоритм ключа экспорта может принимать одно из следующих значений: CALG_KDF_EXPORT, CALG_PRO_EXPORT, CALG_SIMPLE_EXPORT.

PUBLICKEYBLOB	Данный тип блоба используется для экспорта открытых ключей в незашифрованном виде. Параметр **hKey** должен содержать контекст ключа одного из следующих алгоритмов: CALG_GR3410EL_12_256, CALG_DH_EL_12_256_SF, CALG_DH_EL_12_256_EPHEM, CALG_GR3410EL_12_512, CALG_DH_EL_12_512_SF, CALG_DH_EL_12_512_EPHEM, CALG_GR3410EL, CALG_DH_EL_SF, CALG_DH_EL_EPHEM. Параметр **hPubKey** не используется и должен быть равен 0.
PRIVATEKEYBLOB	Данный тип блоба используется при экспорте закрытых ключей в зашифрованном виде. Параметр **hKey** должен содержать дескриптор экспортируемого закрытого ключа с одним из следующих алгоритмов: CALG_GR3410EL_12_256, CALG_DH_EL_12_256_SF, CALG_GR3410EL_12_512, CALG_DH_EL_12_512_SF. Параметр **hPubKey** должен содержать дескриптор ключа, на котором будет выполняться зашифрование экспортируемого закрытого ключа. При выполнении экспорта закрытых ключей алгоритм ключа экспорта может принимать одно из следующих значений: CALG_PRO_EXPORT, CALG_SIMPLE_EXPORT, CALG_KDF_EXPORT.

dwFlags

[in] Флаги экспорта. Данный параметр не используется и должен быть равен 0.

pbData

[out] Указатель на буфер, в который записывается созданный ключевой блоб. Для получения требуемого размера буфера, необходимо в данном параметре указать NULL. Размер буфера возвращается в параметре **pdwDataLen**.

pdwDataLen

[in, out] Адрес, указывающий на значение типа DWORD. Данный параметр на входе содержит размер памяти, выделенной под буфер в параметре **pbData**, в байтах. На выходе из функции, в данном параметре содержится размер, в байтах, фактически скопированных в буфер данных (размер сформированного ключевого блоба). В случае, если размер буфера в параметре pbData недостаточно велик для записи созданного ключевого блоба, устанавливается код ошибки ERROR_MORE_DATA.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.8 Функция формирования (генерации) криптографических ключей - CryptGenKey

Функция позволяет сформировать как сессионные ключи, так и закрытые ключи.

Прототип:

```
BOOL WINAPI CryptGenKey(
  HCRYPTPROV   hProv,
  ALG_ID       Algid,
  DWORD        dwFlags,
  HCRYPTKEY*   phKey
);
```

Параметры:

hProv

[in] Дескриптор криптопровайдера. Дескриптор криптопровайдера должен быть предварительно получен вызовом функции CryptAcquireContext.

Algid

[in] Идентификатор алгоритма, для которого необходимо сформировать ключ.

Идентификатор	**Описание**
AT_SIGNATURE	Сформировать ключ ЭП. Использование данного идентификатора эквивалентно использованию идентификатора CALG_GR3410EL_12_256.
AT_KEYEXCHANGE	Сформировать закрытый ключ шифрования. Использование данного идентификатора эквивалентно использованию идентификатора CALG_DH_EL_12_256_SF.
CALG_GR3410EL_12_256	Сформировать ключ ЭП длиной 256 бит в соответствии с ГОСТ Р 34.10-2012. При создании дескриптора криптопровайдера должен использоваться флаг CRYPT_NEWKEYSET.
CALG_DH_EL_12_256_SF	Сформировать закрытый ключ шифрования длиной 256 бит в соответствии с ГОСТ Р 34.10-2012. При создании дескриптора криптопровайдера должен использоваться флаг CRYPT_-NEWKEYSET.
CALG_DH_EL_12_256_-EPHEM	Сформировать эфемерный закрытый ключ шифрования длиной 256 бит в соответствии с ГОСТ Р 34.10-2012.
CALG_GR3410EL_12_512	Сформировать ключ ЭП длиной 512 бит в соответствии с ГОСТ Р 34.10-2012. При создании дескриптора криптопровайдера должен использоваться флаг CRYPT_NEWKEYSET.
CALG_DH_EL_12_512_SF	Сформировать закрытый ключ шифрования длиной 512 бит в соответствии с ГОСТ Р 34.10-2012. При создании дескриптора криптопровайдера должен использоваться флаг CRYPT_-NEWKEYSET.

CALG_DH_EL_12_512_-EPHEM	Сформировать эфемерный закрытый ключ шифрования длиной 512 бит в соответствии с ГОСТ Р 34.10-2012.
CALG_GR3412_15MG	Сформировать сессионный ключ шифрования в соответствии с с ГОСТ Р 34.12-2015/ГОСТ Р 34.13-2015 (Магма).
CALG_GR3412_15GH	Сформировать сессионный ключ шифрования в соответствии с с ГОСТ Р 34.12-2015/ГОСТ Р 34.13-2015 (Кузнечик).
CALG_G28147	Сформировать сессионный ключ шифрования в соответствии с ГОСТ 28147-89.
CALG_TLS1_MASTER	Сформировать мастер-ключ клиента протокола TLS в состоянии пре-мастер.

dwFlags

[in] Флаги, позволяющие задавать некоторые атрибуты формируемого ключа.

Значение	**Описание**
CRYPT_EXPORTABLE	В случае, если данный флаг установлен, формируемый ключ может быть экспортирован из криптопровайдера в ключевой блоб функцией CryptExportKey. В противном случае, создаваемый (не закрытый) ключ существует только до момента уничтожения дескриптора ключа.
CRYPT_INITIATOR	В случае, если данный флаг установлен, формируемый эфемерный закрытый ключ шифрования используется для вычисления ключа парной связи Диффи-Хелмана.
CRYPT_PRIVATE_3_FROM_6	В случае, если данный флаг установлен, формируемый закрытый ключ разбивается и записывается на 6 ключевых носителей по схеме «3 из 6».
CRYPT_PRIVATE_2_FROM_3	В случае, если данный флаг установлен, формируемый закрытый ключ разбивается и записывается на 3 ключевых носителя по схеме «2 из 3».

phKey

[out] Адрес, по которому в случае успешного завершения работы функции записывается дескриптор сформированного ключа.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.9 Функция формирования случайного числа заданной длины - CryptGenRandom

Прототип:

```
BOOL WINAPI CryptGenRandom(
```

```
  HCRYPTPROV   hProv,
  DWORD        dwLen,
  BYTE*        pbBuffer
);
```

Параметры:

hProv

[in] Дескриптор криптопровайдера. Дескриптор криптопровайдера должен быть предварительно получен вызовом функции CryptAcquireContext.

dwLen

[in] Размер требуемых случайных данных в байтах.

pbBuffer

[in, out] Указатель на буфер в который копируются созданные случайные данные. Размер буфера в байтах передается в параметре **dwLen**.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.10 Функция получения параметров ключа по его дескриптору - CryptGetKeyParam

Прототип:

```
BOOL WINAPI CryptGetKeyParam(
  HCRYPTKEY    hKey,
  DWORD        dwParam,
  BYTE*        pbData,
  DWORD*       pdwDataLen,
  DWORD        dwFlags
);
```

Параметры:

hKey

[in] Дескриптор ключа, который может быть получен вызовом одной из следующих функций: CryptGenKey, CryptDeriveKey, CryptImportKey, CryptGetUserKey.

dwParam

[in] Тип запрашиваемого параметра.

Тип параметра	**Описание**
KP_ALGID	Идентификатор алгоритма ключа (в виде ALG_ID). В параметре **pbData** необходимо передавать указатель на переменную типа ALG_ID.
KP_BLOCKLEN	Для сессионных ключей шифрования возвращается размер блока шифрования данного алгоритма в битах. В параметре **pbData** необходимо передавать указатель на переменную типа DWORD.

KP_KEYLEN	Длина сессионного ключа шифрования или открытого ключа в битах. В параметре **pbData** необходимо передавать указатель на переменную типа DWORD.
KP_- PERMISSIONS	Разрешения использования ключа. Разрешения использования ключа задаются в виде битового поля, которое может содержать одно или несколько из следующих значений: – CRYPT_ENCRYPT - ключ может использоваться для зашифрования; – CRYPT_DECRYPT - ключ может использоваться для расшифрования; – CRYPT_EXPORT - разрешен экспорт ключа; – CRYPT_READ - разрешено чтение параметров ключа; – CRYPT_WRITE - разрешена установка параметров ключа; – CRYPT_MAC - разрешено использование ключа в алгоритмах вычисления имитовставки; – CRYPT_EXPORT_KEY - ключ может использоваться для экспорта других ключей; – CRYPT_IMPORT_KEY - ключ может использоваться для импорта других ключей. В параметре **pbData** необходимо передавать указатель на переменную типа DWORD.

KP_MODE	Используемый режим шифрования для сессионных ключей. Может быть возвращено одно из следующих значений: – CRYPT_MODE_ECB - режим простой замены для ГОСТ Р 34.12-2015/ГОСТ Р 34.13-2015 (Магма, Кузнечик) и ГОСТ 28147-89; – CRYPT_MODE_CTR - режим гаммирования CTR-ACPKM для ГОСТ Р 34.12-2015/ГОСТ Р 34.13-2015 (Магма, Кузнечик); – CRYPT_MODE_CBC - режим простой замены с зацеплением для ГОСТ 28147-89; – CRYPT_MODE_OFB - режим гаммирования для ГОСТ 28147-89; – CRYPT_MODE_CFB - режим гаммирования с обратной связью для ГОСТ 28147-89. В параметре **pbData** необходимо передавать указатель на переменную типа DWORD.
KP_IV	Вектор инициализации (синхропосылка) для сессионных ключей. Размер буфера, необходимого для вектора инициализации, не превышает размер блока шифрования. Размер блока шифрования в битах можно получить с использованием параметра KP_BLOCKLEN. В параметре **pbData** необходимо передавать указатель на массив типа BYTE (BYTE*).
KP_CERTIFICATE	Сертификат, находящийся на ключевом носителе типа смарт-карта и соответствующий закрытому ключу. В параметре **pbData** необходимо передавать указатель на массив типа BYTE (BYTE*).
KP_CIPHEROID	Идентификатор набора используемых параметров шифрования для сессионных ключей, возвращаемый как строка содержащая текстовое представление OID. В параметре **pbData** необходимо передавать указатель на массив типа BYTE (BYTE*).
KP_HASHOID	Идентификатор набора используемых параметров хэширования для ассиметричных ключей, возвращаемый как строка содержащая текстовое представление OID. В параметре **pbData** необходимо передавать указатель на массив типа BYTE (BYTE*).

KP_-SIGNATUREOID	Идентификатор набора используемых параметров эллиптической кривой для ассиметричных ключей для вычисления ЭП, возвращаемый как строка содержащая текстовое представление OID. В параметре **pbData** необходимо передавать указатель на массив типа BYTE (BYTE*).
KP_DHOID	Идентификатор набора используемых параметров эллиптической кривой для ассиметричных ключей для вычисления ключа парной связи Диффи-Хелмана, возвращаемый как строка содержащая текстовое представление OID. В параметре **pbData** необходимо передавать указатель на массив типа BYTE (BYTE*).
KP_KEYSTATE	Текущее состояние сессионного ключа шифрования с идентификатором алгоритма CALG_TLS1_ENC_KEY. В параметре **pbData** необходимо передавать указатель на структуру типа VD_G28147_STATE.
KP_KEYFLAGS	Флаги закрытого ключа. Могут быть возвращены следующие значения: CRYPT_PRIVATE_3_FROM_6 для закрытого ключа, сформированного по схеме «3 из 6»; CRYPT_-PRIVATE_2_FROM_3 для закрытого ключа, сформированного по схеме «2 из 3»; CRYPT_FUNCTIONAL_CARRIER для закрытого ключа, сформированного на функциональном ключевом носителе в неизвлекаемом виде. В параметре **pbData** необходимо передавать указатель на переменную типа DWORD.
KP_EXIMSEED	Значение **seed** алгоритма **KEG**, используемое при формировании ключей экспорта сессионных ключей по ГОСТ Р 34.12-2015/ГОСТ Р 34.13-2015 (Магма, Кузнечик). Описание данного алгоритма приведено в документе **Р 1323565.1.020-2018**. В параметре **pbData** необходимо передавать указатель на массив типа BYTE (BYTE*).
KP_EXIMUKM	Значение **UKM** алгоритма **KEG**, используемое при формировании ключей экспорта сессионных ключей по ГОСТ Р 34.12-2015/ГОСТ Р 34.13-2015 (Магма, Кузнечик). Описание данного алгоритма приведено в документе **Р 1323565.1.020-2018**. В параметре **pbData** необходимо передавать указатель на массив типа BYTE (BYTE*).

pbData

[out] Указатель на буфер, в который копируются запрошенные данные. Тип возвращаемых данных зависит от значения **dwParam**. В случае, если **pbData** равен NULL, то копирование данных не производится. Вместо этого в параметре **pdwDataLen** возвращается требуемый размер буфера в байтах.

pdwDataLen

[in, out] Адрес, указывающий на значение типа DWORD. Данный параметр на входе содержит размер буфера определяемого параметром **pbData**, в байтах.

На выходе в данном параметре содержится размер, в байтах, фактически скопированных в буфер данных. В случае, если в параметре **pbData** передан NULL, то в данном параметре на выходе возвращается размер памяти требуемой для копирования запрошенных данных. В случае, если размер буфера в параметре **pbData** недостаточно велик для копирования в него требуемых данных, устанавливается код ошибки ERROR_MORE_DATA.

dwFlags

[in] Флаги. Данный параметр зарезервирован для дальнейшего использования и всегда должен быть установлен в 0.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.11 Функция загрузки и получения дескриптора закрытого ключа - CryptGetUserKey

Прототип:

```
BOOL WINAPI CryptGetUserKey(
  HCRYPTPROV   hProv,
  DWORD        dwKeySpec,
  HCRYPTKEY*   phUserKey
);
```

Параметры:

hProv

[in] Дескриптор криптопровайдера. Дескриптор криптопровайдера должен быть предварительно получен вызовом функции CryptAcquireContext.

dwKeySpec

[in] Параметр определяющий тип закрытого ключа. Данный параметр может принимать одно из следующих значений:

- AT_SIGNATURE - ключ ЭП;
- AT_KEYEXCHANGE - закрытый ключ шифрования.

phUserKey

[out] Адрес, по которому в случае успешного завершения работы функции записывается требуемый дескриптор ключа.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.12 Функция импорта криптографических ключей из ключевого блоба (массива байт) в криптопровайдер - CryptImportKey

Прототип:

```
BOOL WINAPI CryptImportKey(
```

```
  HCRYPTPROV   hProv,
  const BYTE*  pbData,
  DWORD        dwDataLen,
  HCRYPTKEY    hPubKey,
  DWORD        dwFlags,
  HCRYPTKEY*   phKey
);
```

Параметры:

hProv

[in] Дескриптор криптопровайдера. Дескриптор криптопровайдера должен быть предварительно получен вызовом функции CryptAcquireContext.

pbData

[in] Указатель на буфер, содержащий импортируемый ключевой блоб, который был создан посредством вызова функции CryptExportKey.

dwDataLen

[in] Длина импортируемого ключевого блоба (буфера в параметре **pbData**), в байтах.

hPubKey

[in] Дескриптор ключа, используемый для импорта ключевого блоба.

Тип блоба	**Описание импортируемого ключа и тип используемого в параметре дескриптора**
SIMPLEBLOB	Ключевой блоб этого типа используется для импорта сессионных ключей. Для создания дескриптора ключа необходимо расшифровать ключевую информацию в блобе. Алгоритм ключа дескриптора **hPubKey**, с помощью которого будет расшифровываться ключевая информация, должен принимать одно из следующих значений: CALG_15MG_EXPORT, CALG_15GH_EXPORT, CALG_KDF_EXPORT, CALG_PRO_EXPORT, CALG_SIMPLE_EXPORT.
PUBLICKEYBLOB	Ключевой блоб этого типа используется для импорта открытых ключей. При импорте открытого ключа шифрования может быть указан (через дескриптор **hPubKey**) закрытый ключ шифрования, который будет использоваться в последствии при формировании ключевой пары с импортируемым открытым ключом.

PRIVATEKEYBLOB	Ключевой блоб этого типа используется для импорта закрытых ключей. Для создания дескриптора ключа необходимо расшифровать ключевую информацию в блобе. Алгоритм ключа дескриптора **hPubKey**, с помощью которого будет расшифровываться ключевая информация, должен принимать одно из следующих значений: CALG_KDF_EXPORT, CALG_PRO_EXPORT, CALG_SIMPLE_EXPORT.

dwFlags

[in] Флаги, позволяющие определить атрибуты импортируемого ключа.

Значение	**Описание**
CRYPT_EXPORTABLE	Данный флаг используется в случае если предполагается производить реэкспорт импортируемого ключа. В случае если флаг не установлен при импорте, то попытка экспорта ключа через вызов функции CryptExportKey приведет к ошибке.
CRYPT_IMPORT_CACHE_ONLY	Данный флаг используется в случае если необходимо импортировать закрытый ключ без записи последнего на ключевой носитель.
CRYPT_DO_NOT_UNLOAD_KEY	Данный флаг используется в случае если необходимо гарантировать присутствие импортируемого закрытого ключа в памяти вызывающего процесса вплоть до его завершения.
CRYPT_PRIVATE_3_FROM_6	Данный флаг используется в случае если необходимо импортировать закрытый ключ с разбиением и записью последнего на 6 ключевых носителей по схеме «3 из 6».
CRYPT_PRIVATE_2_FROM_3	Данный флаг используется в случае если необходимо импортировать закрытый ключ с разбиением и записью последнего на 3 ключевых носителя по схеме «2 из 3».

phKey

[out] Адрес, по которому в случае успешного завершения работы функции записывается дескриптор импортированного ключа.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.13 Функция установки параметров ключей - CryptSetKeyParam

Прототип:

```
BOOL WINAPI CryptSetKeyParam(
  HCRYPTKEY    hKey,
  DWORD        dwParam,
```

```
  BYTE*         pbData,
  DWORD         dwFlags
);
```

Параметры:

hKey

[in] Дескриптор ключа, который может быть получен вызовом одной из следующих функций: CryptGenKey, CryptDeriveKey, CryptImportKey, CryptGetUserKey.

dwParam

[in] Тип устанавливаемого параметра. Тип, переданный в этом параметре определяет тип данных (и их формат), передаваемый в параметре **pbData**.

Тип параметра	**Описание и используемый в параметре pbData тип данных**
KP_MODE	Параметр позволяет задавать используемый режим шифрования для сессионных ключей с идентификатором алгоритма CALG_GR3412_15MG, CALG_GR3412_15GH, CALG_TLS1_ENC_KEY и CALG_G28147. Может быть указано одно из следующих значений: – CRYPT_MODE_ECB - режим простой замены для ГОСТ Р 34.12-2015/ГОСТ Р 34.13-2015 (Магма, Кузнечик) и ГОСТ 28147-89; – CRYPT_MODE_CTR - режим гаммирования CTR-ACPKM для ГОСТ Р 34.12-2015/ГОСТ Р 34.13-2015 (Магма, Кузнечик); – CRYPT_MODE_CBC - режим простой замены с зацеплением для ГОСТ 28147-89; – CRYPT_MODE_OFB - режим гаммирования для ГОСТ 28147-89; – CRYPT_MODE_CFB - режим гаммирования с обратной связью для ГОСТ 28147-89. В параметре **pbData** необходимо передавать указатель на переменную типа DWORD.

KP_IV	Использование параметра позволяет задавать вектор инициализации (синхропосылку) для сессионных ключей. Размер буфера, необходимого для вектора инициализации, не превышает размер блока шифрования. Размер блока шифрования в битах можно получить с использованием параметра KP_BLOCKLEN. В параметре **pbData** необходимо передавать указатель на массив типа BYTE (BYTE*).
KP_ALGID	Используемый идентификатор алгоритма ключа. Новое значение идентификатора алгоритма может быть CALG_15MG_EXPORT, CALG_15GH_EXPORT, CALG_KDF_EXPORT, CALG_PRO_EXPORT или CALG_SIMPLE_EXPORT, и оно может быть установлено у ключей с текущим идентификатором алгоритма CALG_GR3412_15MG, CALG_GR3412_15GH или CALG_G28147, или у ключевой пары. В параметре **pbData** необходимо передавать указатель на переменную типа ALG_ID.
KP_X	Параметр используется для фактической генерации эфемерного закрытого и соответствующего ему открытого ключа. В параметре **pbData** должен быть передан NULL.
KP_CLIENT_RANDOM KP_SERVER_RANDOM	Параметр используется для установки случайных данных клиента или сервера сессионного ключа с идентификатором алгоритма CALG_TLS1_MASTER. В параметре **pbData** необходимо передавать указатель на массив со случайными данными.
KP_PREHASH	Параметр используется для перевода сессионного ключа с идентификатором алгоритма CALG_TLS1_MASTER из состояния пре-мастер в состояние мастер. В параметре **pbData** должен быть передан NULL.
KP_CERTIFICATE	Параметр используется для записи сертификата, соответствующего закрытому ключу, на ключевой носитель типа смарт-карта. В параметре **pbData** необходимо передавать указатель на массив типа BYTE (BYTE*).
KP_CIPHEROID	Параметр используется для задания идентификатора набора используемых параметров шифрования для сессионных ключей. В параметре **pbData** необходимо передавать указатель на массив типа BYTE (BYTE*).

KP_HASHOID	Параметр используется для задания идентификатора набора используемых параметров хэширования для ассиметричных ключей. В параметре **pbData** необходимо передавать указатель на массив типа BYTE (BYTE*).
KP_DHOID	Параметр используется для задания идентификатора набора используемых параметров эллиптической кривой для ассиметричных ключей для вычисления ключа парной связи Диффи-Хелмана. В параметре **pbData** необходимо передавать указатель на массив типа BYTE (BYTE*).
KP_KEYSTATE	Параметр используется для импорта состояния сессионного ключа с идентификатором алгоритма CALG_TLS1_ENC_KEY. В параметре **pbData** необходимо передавать указатель на структуру типа VD_G28147_STATE.
KP_EXIMSEED	Параметр используется для задания значения **seed** алгоритма **KEG**, используемого при формировании ключей экспорта сессионных ключей по ГОСТ Р 34.12-2015/ГОСТ Р 34.13-2015 (Магма, Кузнечик). Описание данного алгоритма приведено в документе **Р 1323565.1.020-2018**. В параметре **pbData** необходимо передавать указатель на массив типа BYTE (BYTE*).
KP_EXIMUKM	Параметр используется для задания значения **UKM** алгоритма **KEG**, используемого при формировании ключей экспорта сессионных ключей по ГОСТ Р 34.12-2015/ГОСТ Р 34.13-2015 (Магма, Кузнечик). Описание данного алгоритма приведено в документе **Р 1323565.1.020-2018**. В параметре **pbData** необходимо передавать указатель на массив типа BYTE (BYTE*).
KP_KEYMESH	Параметр используется для задания размера секции режима гаммирования CTR-ACPKM для сессионных ключей по ГОСТ Р 34.12-2015/ГОСТ Р 34.13-2015 (Магма, Кузнечик). В параметре **pbData** необходимо передавать указатель на переменную типа DWORD.

pbData

[in] Указатель на буфер, содержащий значение устанавливаемого параметра. Значение в параметре **dwParam** определяет тип, формат устанавливаемых данных и их размер.

dwFlags

[in] Флаги. Данный параметр зарезервирован для дальнейшего использования и всегда должен быть установлен в 0.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки

может быть получен с помощью вызова системной функции GetLastError().

3.14 Функция зашифрования данных с одновременным их хэшированием - CryptEncrypt

Прототип:

```
BOOL WINAPI CryptEncrypt(
  HCRYPTKEY    hKey,
  HCRYPTHASH   hHash,
  BOOL         Final,
  DWORD        dwFlags,
  BYTE*        pbData,
  DWORD*       pdwDataLen,
  DWORD        dwBufLen
);
```

Параметры:

hKey

[in] Дескриптор используемого сессионного ключа шифрования. Дескриптор сессионного ключа шифрования может быть получен вызовом следующих функций: CryptGenKey, CryptImportKey и CryptDeriveKey.

hHash

[in] Дескриптор объекта хэширования. Данный параметр используется в случае, если необходимо одновременно с зашифрованием данных вычислить по ним значение хэш-функции. Дескриптор объекта хэширования создается вызовом функции CryptCreateHash. В случае, если нет необходимости вычислять значение хэш-функции, в данном параметре передается 0. Одновременное вычисление хэша поддерживается только для сессионных ключей с идентификатором алгоритма CALG_G28147, при этом идентификатор алгоритма хэша должен быть CALG_G28147_IMIT.

Final

[in] Признак, определяющий шифруется ли последний блок данных. Должен быть установлен в значение TRUE, если шифруется последний (или единственный) блок данных. В противном случае устанавливается в значение FALSE.

dwFlags

[in] Флаги. Передаваемое в функцию значение всегда должно быть равно 0.

pbData

[in, out] Указатель на буфер, содержащий данные, которые необходимо зашифровать и, опционально, захэшировать. Размер буфера, в байтах, передается в параметре **dwBufLen**. Размер шифруемых данных передается в параметре **pdwDataLen**. В качестве значения данного параметра может быть передан NULL. В этом случае в параметре **pdwDataLen** возвращается размер памяти необходимый для записи шифрованных данных. Шифрованные данные помещаются в этот же буфер (открытые данные перезаписываются).

pdwDataLen

[in, out] Адрес, указывающий на значение типа DWORD. На входе в функцию в данном параметре передается размер открытых данных, которые необходимо зашифровать. На выходе содержит размер полученных зашифрованных данных. В случае, если в буфере, переданном в параметре **pbData** недостаточно места,

устанавливается код ошибки ERROR_MORE_DATA. В случае, если в параметре **pbData** передан NULL, то код ошибки не устанавливается, а в качестве выходного значения указывается размер памяти, необходимой для записи шифрованных данных.

dwBufLen

[in] Размер буфера, переданного в параметре **pbData**, в байтах.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.15 Функция расшифрования данных с одновременным их хэшированием - CryptDecrypt

Прототип:

```
BOOL WINAPI CryptDecrypt(
  HCRYPTKEY    hKey,
  HCRYPTHASH   hHash,
  BOOL         Final,
  DWORD        dwFlags,
  BYTE*        pbData,
  DWORD*       pdwDataLen
);
```

Параметры:

hKey

[in] Дескриптор используемого сессионного ключа шифрования. Дескриптор сессионного ключа шифрования может быть получен вызовом следующих функций: CryptGenKey, CryptImportKey и CryptDeriveKey.

hHash

[in] Дескриптор объекта хэширования. Данный параметр используется в случае, если необходимо одновременно с расшифрованием данных вычислить по ним значение хэш-функции. Дескриптор объекта хэширования создается вызовом функции CryptCreateHash. В случае, если нет необходимости вычислять значение хэш-функции, в данном параметре передается 0. Одновременное вычисление хэша поддерживается только для сессионных ключей с идентификатором алгоритма CALG_G28147, при этом идентификатор алгоритма хэша должен быть CALG_G28147_IMIT.

Final

[in] Признак, определяющий шифруется ли последний блок данных. Должен быть установлен в значение TRUE, если шифруется последний (или единственный) блок данных. В противном случае устанавливается в значение FALSE.

dwFlags

[in] Флаги. Передаваемое в функцию значение всегда должно быть равно 0.

pbData

[in, out] Указатель на буфер, содержащий данные, которые необходимо расшифровать. Размер расшифровываемых данных передается в параметре **pdwDataLen**. Расшифрованные данные помещаются в этот же буфер (перезаписывают шифрованные). Причем, размер расшифрованных данных не может

быть больше чем размер шифрованных данных.

pdwDataLen

[in, out] Адрес, указывающий на переменную типа DWORD. На входе в функцию данный параметр содержит размер зашифрованных данных. На выходе из функции в параметре возвращается размер полученных открытых данных.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.16 Функция создания и инициализации дескриптора хэша - CryptCreateHash

Прототип:

```
BOOL WINAPI CryptCreateHash(
  HCRYPTPROV   hProv,
  ALG_ID       Algid,
  HCRYPTKEY    hKey,
  DWORD        dwFlags,
  HCRYPTHASH*  phHash
);
```

Параметры:

hProv

[in] Дескриптор криптопровайдера, в контексте которого создается дескриптор хэша. Дескриптор криптопровайдера должен быть предварительно получен вызовом функции CryptAcquireContext.

Algid

[in] Идентификатор алгоритма хэширования.

Идентификатор	**Описание**
CALG_GR3411_12_256	Алгоритм хэширования в соответствии с ГОСТ Р 34.11-2012 (для хэш-значений длиной 256 бит). Значение дескриптора **hKey** должно быть равно 0.
CALG_GR3411_12_512	Алгоритм хэширования в соответствии с ГОСТ Р 34.11-2012 (для хэш-значений длиной 512 бит). Значение дескриптора **hKey** должно быть равно 0.
CALG_GR3411	Алгоритм хэширования в соответствии с ГОСТ Р 34.11-94. Значение дескриптора **hKey** должно быть равно 0.

CALG_GR3412_15MG_MAC	Алгоритм вычисления имитовставки в соответствии с ГОСТ Р 34.12-2015/ГОСТ Р 34.13-2015 (Магма). Значение дескриптора **hKey** должно ссылаться на сессионный ключ с идентификатором алгоритма CALG_GR3412_15MG.
CALG_GR3412_15GH_MAC	Алгоритм вычисления имитовставки в соответствии с ГОСТ Р 34.12-2015/ГОСТ Р 34.13-2015 (Кузнечик). Значение дескриптора **hKey** должно ссылаться на сессионный ключ с идентификатором алгоритма CALG_GR3412_15GH.
CALG_G28147_IMIT	Алгоритм вычисления имитовставки в соответствии с ГОСТ 28147-89. Значение дескриптора **hKey** должно ссылаться на сессионный ключ с идентификатором алгоритма CALG_G28147 или CALG_TLS1_MAC_KEY.
CALG_TLS1_MASTER_HASH	Алгоритм хэширования для вычисления псевдослучайной функции. Значение дескриптора **hKey** должно ссылаться на сессионный ключ с идентификатором алгоритма CALG_TLS1_MASTER.

hKey

[in] Дескриптор сессионного ключа шифрования. Данный параметр используется только в случае, если в параметре **Algid** указан алгоритм вычисления имитовставки CALG_GR3412_15MG_MAC, CALG_GR3412_15GH_MAC или CALG_G28147_IMIT. Для всех остальных значений параметра **Algid** данный параметр должен быть равен 0.

dwFlags

[in] Флаги. Значение данного параметра всегда должно быть равно 0.

phHash

[out] Адрес по которому записывается созданный дескриптор объекта хэширования.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.17 Функция освобождения дескриптора хэша - CryptDestroyHash

Приложения верхнего уровня должны удалять дескрипторы хэшей по завершении их использования с помощью данной функции.

Прототип:

```
BOOL WINAPI CryptDestroyHash(
  HCRYPTHASH   hHash
);
```

Параметры:

hHash

[in] Освобождаемый дескриптор хэша.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.18 Функция создания дубликата хэша по его дескриптору - CryptDuplicateHash

Прототип:

```
BOOL WINAPI CryptDuplicateHash(
  HCRYPTHASH   hHash,
  DWORD*       pdwReserved,
  DWORD        dwFlags,
  HCRYPTHASH*  phHash
);
```

Параметры:

hHash

[in] Дескриптор копируемого объекта хэширования. Указанный дескриптор создается вызовом функции CryptCreateHash.

pdwReserved

[in] Параметр зарезервирован для использования в следующих версиях. В качестве значения данного параметра необходимо всегда передавать NULL.

dwFlags

[in] Флаги. В качестве значения данного параметра необходимо всегда передавать 0.

phHash

[out] Адрес по которому записывается созданный дескриптор хэша. Указанный дескриптор определяет хэш, являющийся дубликатом хэша, определяемого дескриптором, переданным в параметре **hHash**.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.19 Функция получения параметров хэша по его дескриптору - CryptGetHashParam

Прототип:

```
BOOL WINAPI CryptGetHashParam(
  HCRYPTHASH   hHash,
  DWORD        dwParam,
  BYTE*        pbData,
  DWORD*       pdwDataLen,
  DWORD        dwFlags
);
```

Параметры:

hHash

[in] Дескриптор хэша, параметры которого необходимо получить. Дескриптор объекта хэширования может быть создан одной из следующих функций: CryptCreateHash, CryptDuplicateHash.

dwParam

[in] Тип запрашиваемого параметра. Тип запрашиваемого параметра определяет тип данных, возвращаемых в параметре **pbData**.

Тип параметра	**Описание**
HP_ALGID	Позволяет получить идентификатор реализуемого объектом алгоритма хэширования в виде типа ALG_ID. Полученный идентификатор алгоритма хэширования совпадает с идентификатором, использованным при создании дескриптора объекта хэширования функцией CryptCreateHash. В параметре **pbData** необходимо передавать указатель на переменную типа ALG_ID.
HP_HASHSIZE	Позволяет получить размер результата вычисления хэш-функции в байтах. Указанный тип параметра используется перед вызовом функции с параметром HP_HASHVAL и позволяет определить размер памяти, необходимый для записи результата вычисления хэш-функции. Размер результата вычисления хэш-функции зависит от использованного при создании дескриптора объекта хэширования функцией CryptCreateHash идентификатора алгоритма: – CALG_GR3411_12_256 - 32 байта; – CALG_GR3411_12_512 - 64 байта; – CALG_GR3411 - 32 байта; – CALG_GR3412_15MG_MAC - 8 байт; – CALG_GR3412_15GH_MAC - 16 байт; – CALG_G28147_IMIT - 4 байта (размер, необходимый для записи половины результата вычисления имитовставки). В параметре **pbData** необходимо передавать указатель на переменную типа DWORD.

HP_HASHVAL	Позволяет получить результат вычисления значения хэш-функции. Значение хэш-функции вычисляется по данным добавляемым в объект хэширования функциями CryptHashData, CryptEncrypt и CryptDecrypt. Необходимо учитывать, что вызов функции с данным параметром приводит к завершению хэша, определяемого дескриптором, переданным в параметре **hHash**. После завершения объекта хэширования в него невозможно добавить новые данные. При использовании хэша с идентификатором алгоритма CALG_G28147_IMIT и при наличии в буфере для значения хэша достаточного места (как минимум 8 байт), в него копируется полный результат вычисления имитовставки. В параметре **pbData** необходимо передавать указатель на массив типа BYTE (BYTE*).
HP_HASHOID	Позволяет получить идентификатор набора используемых параметров хэширования. В параметре **pbData** необходимо передавать указатель на массив типа BYTE (BYTE*).
HP_- HASHSTATE	Позволяет получить текущее состояние хэша с идентификатором алгоритма CALG_G28147_IMIT. В параметре **pbData** необходимо передавать указатель на структуру типа VD_G28147_STATE.

pbData

[out] Указатель на буфер, в который копируются запрошенные данные. Тип возвращаемых данных зависит от значения **dwParam**. В случае, если **pbData** равен NULL, то копирование данных не производится. Вместо этого, в параметре **pdwDataLen** возвращается требуемый размер буфера в байтах.

pdwDataLen

[in, out] Адрес, указывающий на значение типа DWORD. Данный параметр на входе содержит размер буфера определяемого параметром **pbData**, в байтах. На выходе в данном параметре содержится размер, в байтах, фактически скопированных в буфер данных. В случае, если в параметре **pbData** передан NULL, то в данном параметре на выходе возвращается размер памяти требуемой для копирования запрошенных данных. В случае, если размер буфера в параметре **pbData** недостаточно велик для копирования в него требуемых данных, устанавливается код ошибки ERROR_MORE_DATA.

dwFlags

[in] Флаги, позволяющие определить атрибуты параметра хэша.

Значение	**Описание**

CRYPT_IGNORE_PRF_SECRET	Данный флаг используется в случае, если нет необходимости использовать ключевой материал при вычислении псевдо-случайной функции.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.20 Функция добавления новых данных в хэш - CryptHashData

Прототип:

```
BOOL WINAPI CryptHashData(
  HCRYPTHASH   hHash,
  const BYTE*  pbData,
  DWORD        dwDataLen,
  DWORD        dwFlags
);
```

Параметры:

hHash

[in] Дескриптор хэша, в который производится добавление данных. Дескриптор объекта хэширования может быть создан функцией CryptCreateHash или CryptDuplicateHash.

pbData

[in] Указатель на буфер, содержащий данные добавляемые в объект хэширования. Размер буфера передается в параметре **dwDataLen**.

dwDataLen

[in] Размер переданного в параметре **pbData** буфера, содержащего данные добавляемые в объект хэширования. **dwFlags**

[in] Флаги. В качестве значения параметра необходимо всегда передавать 0.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.21 Функция добавления сессионного ключа шифрования в хэш - CryptHashSessionKey

Прототип:

```
BOOL WINAPI CryptHashSessionKey(
  HCRYPTHASH   hHash,
  HCRYPTKEY    hKey,
  DWORD        dwFlags
);
```

Параметры:

hHash

[in] Дескриптор хэша с идентификатором алгоритма CALG_GR3411_12_256 или CALG_GR3411_12_512, в который производится добавление сессионного ключа. Дескриптор объекта хэширования может быть создан функцией CryptCreateHash или CryptDuplicateHash.

hKey

[in] Дескриптор используемого сессионного ключа шифрования с идентификатором алгоритма CALG_GR3412_15MG, CALG_GR3412_15GH, CALG_TLS1_ENC_KEY или CALG_TLS1_MAC_KEY. Дескриптор сессионного ключа шифрования может быть получен вызовом следующих функций: CryptGenKey, CryptImportKey и CryptDeriveKey.

dwFlags

[in] Флаги, позволяющие устанавливать атрибуты хэширования сессионного ключа.

Значение	**Описание**
CRYPT_LITTLE_ENDIAN	Данный флаг используется в случае, если следует хэшировать сессионный ключ шифрования начиная с первого (младшего) байта ключа.
CRYPT_HMAC_XOR_IPAD CRYPT_HMAC_XOR_OPAD	Данные флаги (взаимоисключающие) используются в случае, если сессионный ключ применяется для вычисления HMAC (hash-based message authentication code) в соответствии с RFC 2104.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.22 Функция установки параметров хэша по его дескриптору - CryptSetHashParam

Прототип:

```
BOOL WINAPI CryptSetHashParam(
  HCRYPTHASH   hHash,
  DWORD        dwParam,
  BYTE*        pbData,
  DWORD        dwFlags
);
```

Параметры:

hHash

[in] Дескриптор хэша, параметры которого необходимо установить. Дескриптор объекта хэширования может быть создан одной из следующих функций: CryptCreateHash, CryptDuplicateHash.

dwParam

[in] Тип устанавливаемого параметра хэша. Тип параметра определяет тип и размер данных, передаваемых в параметре **pbData**.

Тип параметра	**Описание**
HP_HASHVAL	Позволяет установить результат вычисления значения хэш-функции. Вызов функции с данным типом параметра завершает объект хэширования, определяемый дескриптором, переданным в параметре **hHash**. В параметре **pbData** необходимо передавать указатель на буфер типа BYTE (BYTE*). Размер данного буфера зависит от алгоритма создания объекта хэширования, для которого устанавливается параметр: – CALG_GR3411_12_256 - 32 байта; – CALG_GR3411_12_512 - 64 байта; – CALG_GR3411 - 32 байта; – CALG_GR3412_15MG_MAC - 8 байт; – CALG_GR3412_15GH_MAC - 16 байт; – CALG_G28147_IMIT - 4 байта (размер, необходимый для хранения половины результата вычисления имитовставки).
HP_TLS1PRF_LABEL	Позволяет установить метку для хэша с идентификатором алгоритма CALG_TLS1_MASTER_HASH, используемую при вычислении псевдо-случайной функции. В параметре **pbData** необходимо передавать указатель на структуру типа CRYPT_DATA_BLOB.
HP_TLS1PRF_SEED	Позволяет установить начальные данные для хэша с идентификатором алгоритма CALG_TLS1_-MASTER_HASH, используемые при вычислении псевдо-случайной функции. В параметре **pbData** необходимо передавать указатель на структуру типа CRYPT_DATA_BLOB.
HP_HASHOID	Позволяет установить идентификатор набора используемых параметров хэширования. В параметре **pbData** необходимо передавать указатель на массив типа BYTE (BYTE*).

HP_HASHSTATE	Позволяет импортировать состояние хэша с идентификатором алгоритма CALG_G28147_IMIT. В параметре **pbData** необходимо передавать указатель на структуру типа VD_G28147_STATE.

pbData

[in] Указатель на буфер, содержащий значение устанавливаемого параметра. Используемый тип данных и их размер определяются значением в параметре **dwParam**.

dwFlags

[in] Флаги. В качестве значения данного параметра необходимо всегда передавать 0.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.23 Функция вычисления ЭП по значению хэша - CryptSignHash

Прототип:

```
BOOL WINAPI CryptSignHash(
  HCRYPTHASH   hHash,
  DWORD        dwKeySpec,
  LPCWSTR      sDescription,
  DWORD        dwFlags,
  BYTE*        pbSignature,
  DWORD*       pdwSigLen
);
```

Параметры:

hHash

[in] Дескриптор подписываемого хэша с идентификатором алгоритма CALG_GR3411_12_256 или CALG_GR3411_12_512. Дескриптор объекта хэширования может быть создан одной из следующих функций: CryptCreateHash, CryptDuplicateHash. При вычислении ЭП будет произведена загрузка закрытого ключа, идентификатор контейнера которого использовался при создании дескриптора криптопровайдера и который, в свою очередь, использовался при создании дескриптора подписываемого хэша. Требования к соответствию идентификаторов алгоритма хэша и закрытого ключа описываются следующие таблицей:

Идентификатор алгоритма хэша	**Идентификатор алгоритма закрытого ключа**

CALG_GR3411_12_256	CALG_GR3410EL_12_256 CALG_DH_EL_12_256_SF
CALG_GR3411_12_512	CALG_GR3410EL_12_512 CALG_DH_EL_12_512_SF

dwKeySpec

[in] Тип закрытого ключа. В качестве значения данного параметра необходимо использовать одну из констант AT_SIGNATURE или AT_KEYEXCHANGE.

sDescription

[in] Данный параметр не используется и всегда должен быть равен NULL.

dwFlags

[in] Флаги. В качестве значения данного параметра необходимо всегда передавать 0.

pbSignature

[out] Указатель на буфер, в который записывается вычисленное по данному объекту хэширования значение ЭП. В случае, если **pbSignature** равен NULL, то запись вычисленного значения ЭП не производится. Вместо этого, в параметре **pdwSigLen** возвращается требуемый для записи размер буфера в байтах.

pdwSigLen

[in, out] Адрес, указывающий на значение типа DWORD. Данный параметр на входе содержит размер буфера переданного в параметре **pbSignature**, в байтах. На выходе из функции в параметре возвращается количество байт, записанных в буфер, определяемый параметром **pbSignature**, т.е. размер выработанной ЭП. В случае, если в параметре **pbSignature** передан NULL, то в данном параметре на выходе возвращается размер памяти требуемой для записи вычисленного значения ЭП. В случае, если размер буфера в параметре **pbSignature** недостаточно велик для записи в него требуемых данных, устанавливается код ошибки ERROR_MORE_DATA.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.24 Функция проверки ЭП по значению хэша - CryptVerifySignature

Прототип:

```
BOOL WINAPI CryptVerifySignature(
  HCRYPTHASH   hHash,
  const BYTE*  pbSignature,
  DWORD        dwSigLen,
  HCRYPTKEY    hPubKey,
  LPCWSTR      sDescription,
  DWORD        dwFlags
);
```

Параметры:

hHash

[in] Дескриптор хэша для проверки ЭП с идентификатором алгоритма

CALG_GR3411_12_256, CALG_GR3411_12_512 или CALG_GR3411. Дескриптор объекта хэширования может быть создан одной из следующих функций: CryptCreateHash, CryptDuplicateHash.

pbSignature

[in] Указатель на буфер, содержащий проверяемое значение ЭП. Передаваемое значение вырабатывается при вызове функции CryptSignHash.

dwSigLen

[in] Размер буфера переданного в параметре **pbSignature** (размер проверяемого значения ЭП), в байтах.

hPubKey

[in] Дескриптор открытого ключа, используемый для проверки значения ЭП. Открытый ключ, определяемый данным дескриптором должен соответствовать закрытому ключу, использованному при формировании значения ЭП функцией CryptSignHash. Требования к соответствию идентификаторов алгоритма хэша и открытого ключа описываются следующие таблицей:

Идентификатор алгоритма хэша	**Идентификатор алгоритма открытого ключа**
CALG_GR3411_12_256	CALG_GR3410EL_12_256 CALG_DH_EL_12_256_SF
CALG_GR3411_12_512	CALG_GR3410EL_12_512 CALG_DH_EL_12_512_SF
CALG_GR3411	CALG_GR3410EL CALG_DH_EL_SF

sDescription

[in] Данный параметр не используется и всегда должен быть равен NULL.

dwFlags

[in] Флаги. В качестве значения данного параметра необходимо всегда передавать 0.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

3.25 Функция создания дубликата ключа по его дескриптору - CryptDuplicateKey

Прототип:

```
BOOL WINAPI CryptDuplicateKey(
  HCRYPTKEY    hKey,
  DWORD*       pdwReserved,
  DWORD        dwFlags,
  HCRYPTKEY*   phKey
);
```

Параметры:

hKey

[in] Дескриптор копируемого объекта ключа. Указанный дескриптор создается вызовом функций CryptGenKey, CryptDeriveKey, CryptImportKey,

CryptGetUserKey. В настоящее время возможно создание дубликатов исключительно открытых ключей и сессионных ключей.

pdwReserved

[in] Параметр зарезервирован для использования в следующих версиях. В качестве значения данного параметра необходимо всегда передавать NULL.

dwFlags

[in] Флаги. В качестве значения данного параметра необходимо всегда передавать 0.

phHash

[out] Адрес по которому записывается созданный дескриптор ключа. Указанный дескриптор определяет ключ, являющийся дубликатом ключа, определяемого дескриптором, переданным в параметре **hKey**.

Возвращает:

При успешном завершении функция возвращает TRUE, в противном случае возвращается FALSE. Если возвращается FALSE, соответствующий код ошибки может быть получен с помощью вызова системной функции GetLastError().

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

КЗИ	Криптографическая защита информации
ЭП	Электронная подпись (Digital Signature)

ПЕРЕЧЕНЬ ТАБЛИЦ

Лист регистрации изменений

Номера листов (страниц)					Всего листов (страниц) в докум	№ документа	Входящий № сопроводительного докум и дата	Подп	Дата
Изм	изменённых	заменённых	новых	аннулированных					