ООО Фирма «Инфокрипт»

Описание интерфейса

VPNKey-TLS

ИНФК.11485466.4012.015.32 01

Редакция 2.47

(Документация актуальна для прошивок [505.23], а также указанных явно в тексте)

Москва
2020

История изменений:

№	Дата	Тема (кратко)
2.20	24.03.2016	- Описание правил построения "devcfg" в GET_SYS_INFO, - Замена "ЭЦП" на "ЭП" (согласно ГОСТ 2012), - Конкретизация формулировок, уточнения, форматирование.
2.21		- Уточнение кодов ошибок: "73"
2.22		- Исправление опечатки "FIN_UPD_ID"
2.23	12.05.2016	- Указан размер данных для режима 1. - Добавлено описание новых кодов ошибок (787, 807).
2.24	30.07.2016	- Добавлена таблица требований к аргументу "attr" функции "CREATE_PAIR_EX_ID" - Правки опечаток
2.25	13.10.2016	- Добавлен параметр "name" функции "INIT_SIGN_H_ID" - Правки опечаток
2.26	14.10.2016	- добавление кодов ошибок
2.27	23.11.2015	- Удаление параметра "cert_data" в "INIT_CHECK_H_ID"
2.28	26.12.2016	- Важные изменения в главе о форматах HTTP-данных
2.29	10.01.2017	- Добавлен раздел "ATTACHED" в п.17.
2.30	19.01.2017	- Дополнения о подписании с визуализацией файлов длиннее 15 кб.
2.31	20.01.2017	- Изменения п.4 (SET_ACT_OBJ_ID). - исправление опечаток.
2.32	31.01.2017	- Дополнения в главе о форматах HTTP-данных. - Описание дополнительных кодов возврата. - Дополнения в п.17.
2.33	09.02.2017	- Дополнения в главе о форматах HTTP-данных.
2.34	10.02.2017	- Исправления в главе о форматах HTTP-запросов. - Добавления новых ключей функции SIGN_PACKET
2.35	13.02.2017	- Поддержка "изменений алгоритма инициализации VPN-Key TLS", коды ошибок, исправления и уточнения о наследных аргументах функций криптоопераций.
2.36	27.02.2017	- Пояснения и доп. иллюстрации в раздел 17 "Визуализация подписываемых данных"
2.37	28.02.2017	- Правка раздела 17 - Приведение по всему документу СИД-ов, с которыми вызываются функции АПИ, в соответствие.
2.38	22.08.2018	- добавление описания включённого в прошивки 505 функционала прошивок 550
2.39	24.09.2018	- картинка с образцами шрифтов для экранников заменена таблицей
2.40	08.11.2018	- дополнение списка кодов возврата - уточнения АПИ подписания для экранников - обновление сценариев работы с токеном
2.41	10.04.2019	- Актуализация версий
2.42	03.07.2019	- Исправление описания ОИД-ов KeyUsage и ExtendedKeyUsage
2.43	22.08.2019	- форматирование
2.44	10.09.2019	- длины данных для криптоопераций
2.45	06.12.2019	- дополнение интерфейса INIT_SIGN_H_ID - удаление упоминаний о прошивках <500 - комментарии к GET_SIGN_CMS_H_ID - коррекция формулировок, правки опечаток
2.46	10.01.2020	- дополнение списка кодов возврата
2.47	20.03.2020	- добавление аргумента "signed_attrs" в функцию "CREATE_PAIR_EX" - синхронизация с версиями, правка опечаток, актуализация кодов возврата, уточнение формулировок

Оглавление

Введение

Устройство VPNKey-TLS обеспечивает поддержку работы через HTTP-запросы. Для этого с флеш-диска токена должна быть запущена программа-сервер "start.exe", которая будет обрабатывать HTTP-запросы, формируемые ПО клиента, на IP-адресе "localhost" (127.0.0.1). В запросах используются идентификаторы сессий в URL. Все функции API устройства расположены по пути vpnkeylocal адреса HTTP-сервера устройства.

Для выполнения процедуры аутентификации клиентское ПО должно использовать начальный идентификатор сессии SID0. Данный идентификатор находится в URL в файле sslgate.url на флеш-диске токена. Например, ниже приводится вид данного файла, где SID0 представляет собой 34 символа латиницы и цифр:
«tAh5A9e5gWUW6i33H66q7dS807ZKzRerci»

```
[InternetShortcut]
URL=http://localhost:28016/vpnkeylocal/tAh5A9e5gWUW6i33H66q7dS807ZKzRerci/
IconFile=http://localhost:28016/favicon.ico
IconIndex=1
```

Идентификатор сессии формируется динамически каждую сессию (каждое переподключение токена).

Тонкий клиент

В случае «тонкого» клиента броузером осуществляется переход непосредственно по URL в файле sslgate.url . После успешной аутентификации окно броузера отображает страницу главного меню интерфейса администрирования, а URL содержит уже идентификатор сессии SID1, который отличается от SID0 и генерируется после аутентификации.
Из главного меню в окне броузера возможен переход на страницу нужной бизнес-системы (видимой), чьи реквизиты установлены на токен. После выбора кнопки нужной бизнес-системы броузер осуществляет переход по URL вида
http://localhost:28016/ic/?infocrypt_sid=12345678901234567890123456789012 34&infocrypt_prefix=aHR0cDovL2xvY2FsaG9zdDoyODAxNi8=,
где присутствует идентификатор сессии SID2 для работы с API «12345678901234567890123456789012 34». Идентификатор SID2 должен использоваться в URL для формирования запросов от «тонкого» клиента к API токена. URL в данном случае имеет вид «http://localhost:28016/vpnkeylocal/<SID2>/».

Толстый клиент

Для начала работы «толстому» клиенту необходимо получить SID0 из sslgate.url, с его использованием сформировать POST-запрос к API для получения списка доступных учетных записей (см.GET_PIN_LIST), и выполнить аутентификацию путем POST-запроса к функции API (см.LOGIN). Ответ на данный запрос будет содержать SID2, который следует в дальнейшем использовать для формирования POST-запросов к другим функциям API.

Функции веб-приложения

1. Генерирование/обработка ключевых данных
2. Операции создания/проверки ЭП по ГОСТ Р 34.10-2001 и ГОСТ Р 34.10-2012

Вызов функций

Вызов функции осуществляется путем формирования POST-запроса, тело которого включает поля
- id функции,
- имя и значение параметра 1,
…
- имя и значение параметра N

Результат выполнения функции размещается в одном из заранее предусмотренных полей ответа на запрос.

Формат запросов

В соответствии с RFC 2068:

Request = Request-Line *(general-header | request-header | entity-header) CRLF
[message-body]

Request-Line = Method SP Request-URI SP HTTP-Version CRLF
Method = “POST”

Тело запроса должно состоять из имён полей и значений. Например:

```
ID=COMMAND_ID&Field1=%D0%12%C0%13QWERTY%11%F2%22&Field2=&Field3=11234
```

Содержимое запросов

1. Главным полем в запросе является "ID". Это команда, показывающая, какую операцию токену следует произвести.
2. Порядок полей в запросе является произвольным, однако рекомендуется ставить поле "ID" на первое место. Также рекомендуется располагать поля с короткими значениями (числами, енумераторами и хендлами) в начале запроса, а поля, длина которых может превысить 15 кб - в конце. Это относится в первую очередь к командам криптоопераций: подписания, проверки подписи, шифрования и расшифровывания. Как только токен получит имена всех обязательных полей, он может начать криптооперацию до окончания прихода всех данных, что может значительно ускорить обработку запроса.
3. Повторное вхождение одного и того же поля в запрос не допускается. Это более строгое, чем в RFC, правило введено для возможности начать выполнение команды до окончания прихода всех её аргументов, (длина которых может достигать десятков мегабайт), что, в свою очередь, позволяет ускорить работу до двух раз.

*Кодировка запросов**

1. В целях обеспечения клиенту возможности передавать на сервер токена HTTP-запросы такой длины, обработка которой Java-скриптами за одно HTTP-обращение может оказаться за гранью возможностей Java-скриптов (что бывает при операциях подписания, проверки подписи, шифрования, расшифровывания, а также загрузки обновления), сервер токена[1] поддерживает "`Transfer-Encoding: chunked`". (Для JS это выглядит как: `HttpWebRequest.SendChunked = true`). Для остальных запросов, которые Java-скрипт может гарантированно обработать одним HTTP-вызовом, не следует злоупотреблять `chunked`-кодированием, в целях экономии машинного времени.

2. Сервер токена поддерживает URL-кодировку "`Content-Type: application/x-www-form-urlencoded`" (по умолчанию). Для запросов этого типа ПО клиента должно преобразовать с помощью функции URL-encode все поля запроса. Кодировке должны быть подвергнуты только значения полей; символы "=" и "&", разделяющие имя поля и его значение следует оставить без изменения.

   - Запросы с длиной менее 15 кб рекомендуется передавать именно в этом формате.
   - Существует ПО, преобразовывающее с помощью URL-encode не все поля запроса, а только имеющие (по настоящей документации) тип "STRING", и не кодирующее поля, имеющие тип "BASE64". В целях совместимости с этим ПО, такая ситуация автоматически обнаруживается и поддерживается сервером токена. Однако она не соответствует RFC и обрабатывается как исключение, что требует дополнительного машинного времени. Поэтому для коротких запросов, передаваемых в этом формате, рекомендуется кодировать все поля, как того требует RFC, а длинные запросы отправлять в формате "`Content-Type: multipart/form-data`" (см.ниже).
   - Сервер токена поддерживает "`Content-Type: text/plain`" и "`Content-Type: text/html`". Однако для обоих форматов требование наличия URL-кодировки полей внутри запроса сохраняются, т.к. в общем случае поля типов PEMDER или STRING могут содержать символы "=" и "&", из-за которых парсер запроса, требующий формата: "`key1=value1&key2=value2`", может быть дезориентирован, что приведёт к неверной интерпретации запроса.

3. Сервер токена поддерживает "`Content-Type: multipart/form-data`". В этом случае все поля запроса должны передаваться клиентом без дополнительной кодировки. (`chunked`-кодирование при этом допускается).

   - Обратите внимание! Поля типа **BASE64** с таким типом контента должны передаваться **БЕЗ** кодировки **BASE64**. Поля типа **STRING** с таким типом контента должны передаваться **БЕЗ** кодировки **URL-encode.**

   Этот режим рекомендуется для запросов длиной в несколько мегабайт (криптоопераций в режиме 1) для достижения максимальной скорости.

Кодировка ответов

Поля ответов от HTTP-сервера, в которых отсутствует возможность использования не ASCII-символов, не кодируются. Остальные поля ответов для запросов "Content-Type: text/plain" или "Content-Type: text/html" кодируются только в BASE64. Поля запросов и ответов на запросы с "Content-Type: multipart/form-data" передаются без кодировки.

[1] В прошивках нечётных версий 550 и выше

Форматы данных, используемых с HTTP-интерфейсе:

Название формата	Направление (слева ПК)	Контроль "Стартом.exe"		Токену передаётся	Комментарии
		алфавит	другое		
BASE64	<< >>	Для "Content-Type" != "multipart/form-data" - символы Base64 и/или url-safe-Base64	Длина декодированных данных [от и до]	Для "Content-Type" != "multipart/form-data" - результат однократного декодирования из Base64/url-safe-Base64.	Поддерживаются Base64 и url-safe-Base64 одновременно. Для "multipart/form-data" - данные клиента передаются без Base64-декодирования.
ENUM(...)	<< >>	только цифры	варианты только из явно указанных в документации.	Число типа int8_t	Ведущие нули теряются
HANDLE_	<< >>	["0"; "9"], ["A"; "Z"], ["a"; "z"]	Длина [от и до]	Строка переданных символов переданной длины	
ID_FORMAT	>>	Заглавные латинские буквы и "_"	Точное соответствие значению из таблицы	Команда CCID	Используется только для идентификаторов команд, не для аргументов.
NUMBER	<< >>	только цифры	Значение [от и до]	Число типа int32: [-(2^31); (2^31)-1]	Ведущие нули теряются
NUMBER64	<< >>	только цифры	Значение [от и до]	Число типа int64: [-(2^63); (2^63)-1]	Ведущие нули теряются
PEM	<<	Base64, "-", " "	Длина [от и до]	*	ASN.1-данные, закодированные в Base64 и обрамлённые PEM-заголовками: "-----BEGIN DATA----- ..... -----END DATA-----"
PEMDER	>>	не контролируется	Длина декодированных двоичных данных [от и до]	Конечный результат декодирования из Base64/url-safe-Base64	Допускаются данные в двоичном формате ASN.1, или в PEM /Base64 /url-safe-Base64, в т.ч. кодированные больше одного раза. Декодирование осуществляется до получения не-PEM-символов, после чего результат отправляется на токен.
PIN_STR	>>	только цифры	Длина [от и до]	Строка переданных символов переданной длины	Ведущие нули сохраняются
STRING	<< >>	не контролируется	Длина [от и до]	Результат декодирования из формата, указанного в заголовке HTTP	
STRING_A	<<	Печатные символы ASCII	Длина [от и до]	*	Используется для передачи незакодированной информации о конфигурации токена, времени итд.

Коды возврата

В теле ответа на POST-запрос присутствует поле «retcode=”N”»
«N» - это знаковое число в 10-чной системе счисления, пригодное для парсинга (atoi).
Если «N» отличается от кода «OK» ("1"), то остальные поля в теле могут отсутствовать или содержать невалидную информацию, если иного не указано явно в описании команды, в ответ на которую пришёл такой retcode.

«N»	Мнемоника ответа	Пояснения
0	TOKEN_IO_ERROR	Ошибка связи с устройством
1	OK	Функция успешно выполнена
2	ARGUMENTS_BAD	Указаны неверные аргументы (несоответствие алфавита; выход за явно указанные в документе границы; недопустимые символы в base64, итд)
3	GEC_CMS_SIGNCOUNT	Недостаточное число подписей контейнера корневого сертификата
4	GEC_NOVALIDSIGN	Не удалось проверить подлинность подписи
6	GEC_NTRUSTEDRT	Попытка загрузки корневого сертификата без безопасного контейнера
7	GEC_RATTRCHECK	Ошибка классификации корневого сертификата
8	GEC_CATTRCHECK	Ошибка классификации сертификата УЦ
9	GEC_SATTRCHECK	Ошибка классификации сертификата ЭП
10	GEC_TATTRCHECK	Ошибка классификации сертификата TLS
11	GEC_UNCLASSF	Класс сертификата определить не удалось
12	GEC_DUPLICATE	Дубликат (сертификата)
14	GEC_ERROR	Прочая ошибка при обработке входящего сертификата.
15	PARSE_ERROR	Ошибка парсинга формата ASN.1/x.509
16	DATA_LEN_RANGE	Размер данных превышает допустимый (Функция получила недостаточно данных, или больше чем было обещано, или неожиданный номер блока данных.)
21	UA_NOT_ENOUGH_STORAGE	В хранилище объектов недостаточно места
24	UA_USER_ALREADY_LOGGED_IN	Пользователь уже выполнил вход
25	UA_USER_BLOCKED	Пользователь заблокирован
26	UA_RND_INIT_ERROR	Ошибка инициализации ДСЧ
27	UA_FAILED_PUK_TRIES	Израсходованы попытки ввода PUK
28	UA_FAILED_PIN_TRIES	Израсходованы попытки ввода PIN
29	UA_INCORRECT_PIN	Введен некорректный PIN
32	USER_NOT_LOGGED_IN	Пользователь не выполнил вход
33	OPERATION_NOT_INITIALIZED	Операция не проинициализирована
35	OBJECT_HANDLE_INVALID	Введен некорректный идентификатор объекта
42	NOT_IN_INIT_MODE	Устройство не в режиме инициализации

44	NOT_IN_ACTIVE_MODE	Устройство не в активном режиме
47	INVALID_BS_ID	Введен некорректный идентификатор бизнес-системы
48	PUK_INCORRECT	Введен некорректный PUK
51	TMPL_NOT_APPLICABLE	Шаблон не применяется на данном устройстве
52	TMPL_CMS_ERROR	Ошибка формата CMS при разборе шаблона
53	TMPL_SIGN_ERROR	Ошибка подписи под шаблоном
54	DOC_FORMAT_ERROR	Ошибка формата при разборе документа.
55	TMPL_DOCPARSE_ERROR	Прочая ошибка при разборе документа.
56	TMPL_SYNTAX_ERROR	Синтаксическая ошибка в документе
57	TMPL_IMPOSSIBLE_VISUALISE	Документ нельзя подписать как визуализированный.
58	TMPL_NO_TEMPLATE	Невозможно отобразить документ т.к. не был загружен шаблон.
59	TMPL_FIELDS_ERROR	Ошибка формата шаблона
60	TMPL_NOT_MATCH	Версия документа и шаблона не совместимы.
61	TMPL_NOT_ENOUGH_MEMORY	Недостаточно памяти для визуализации всех документов
62	TMPL_XML_ERROR	Ошибка парсинга XML-шаблона
63	TMPL_XML_OVERQUOTE	Превышение квоты документов на подписание.
90	INVALID_SID	Введен некорректный SID
95	FUNCTION_NOT_SUPPORTED	Нераспознанная/неподдерживаемая команда
96	SESSION_TIMEOUT	Таймаут сессии
97	OPERATION_NOT_COMPLETE	Операция (подписания; проверки подписи; шифрования) не завершена
200	CCIDSSL_ERR_PROXYCON	Ошибка соединения с HTTP прокси. Проверьте настройки.
210	CCIDSSL_ERR_PROXYAUTH	Ошибка аутентификации на HTTP прокси. Проверьте ваши логин и пароль.
220	CCIDSSL_ERR_PROXYGENERIC	Ошибка работы с HTTP прокси.
700	CORE_FAULT_00	Ошибка ядра #0.
701	TOKEN_INIT_ERROR	Ошибка инициализации токена.
702	MALLOC_INT_ERROR	Ошибка выделения защищённой памяти на токене
703	MALLOC_EXT_ERROR	Ошибка выделения внешней памяти на токене
704	MALLOC_PC_ERROR	Ошибка выделения памяти на ПК.
705	MALLOC_ERROR	Ошибка выделения памяти (наследная).
720	FS_WRONG_FILEID	Файл не найден.
722	FS_IO_WRITE_ERROR	Ошибка записи файла на токен.
723	FS_IO_READ_ERROR	Ошибка чтения файла с токена.
724	FS_PERMISSION_ERROR	Недостаточно полномочий для файловой операции на токене.

725	FS_NOT_WR_MODE	Недопустимая файловая операция.
726	FS_ZOP_ERROR	Ошибка защищённой памяти.
727	FS_INIT_ERROR	Ошибка инициализации файловой системы.
728	FS_INTERNAL_ERROR	Внутренняя ошибка файловой системы.
750	EXT_FS_INTERNAL_ERROR	Внутренняя ошибка внешней файловой системы.
760	TRY_INIT_TWICE	Попытка инициализировать хранилище дважды.
770	FS_MIGRATE_ERROR	Ошибка чтения файла, записанного в неподдерживаемом формате.
771	ZOP_MIGRATE_ERROR	Ошибка миграции защищённой памяти.
780	CO_HANDLE_INVALID	Неверный хендл криптооперации.
781	CO_NO_FREE_CONTENT	Нет свободных слотов для криптооперации.
782	CO_UI_IS_BUSY	Интерфейс занят другой интерактивной операцией.
783	CO_REJECTED_BY_USER	Операция отменена пользователем.
784	CO_USER_NOT_READY	Ожидание выбора пользователя.
785	CO_USER_WAIT_TIME_OUT	Истёк таймаут согласия пользователя на операцию.
786	CO_NO_RECEPIENT	Не удалось найти указанный сертификат в списке получателей зашифрованного документа.
787	CO_RECEPIENT_NOT_SPECIFIED	Не указан сертификат, в адрес которого выполняется шифрование.
788	CO_TOO_LONG_CERTS_CHAIN	Цепочка сертификатов указанной длины не может быть выгружена.
789	CO_TIME_WASNT_SET	На устройстве не установлено время
790	UPD_OLD_VERSION	Попытка загрузить более старую версию прошивки.
791	UPD_WRONG_SIGNATURE	Неверный заголовок в файле прошивки.
792	UPD_ERR_BL_SIGN	Подпись под обновлением неверна.
793	UPD_OLD_ERR_BL_UPDATE_FINISH	Загрузка обновления завершена.
794	UPD_ERR_WRONG_KEY	Неверный ключ шифрования прошивки.
795	UPD_OLD_ERR_BL_NO_UPDATE	Обновление не найдено.
796	UPD_OLD_ERR_BL_NOT_COMPATIBLE	Обновление не совместимо с текущей программно-аппартной конфигурацией.
797	UPD_OLD_ERR_BL_NO_SPACE	Недостаточно места для загрузки обновления.
798	UPD_OLD_ERR_UPDATE_BLOCK	Блок обновления поврежден или зашифрован на другом ключе.
799	UPD_OLD_ERR_UPDATE_KEY	Ключи шифрования обновлений не были загружены или были повреждены.
800	UPD_OLD_ERR_BLOADER_KEYNO	Неверный номер ключа для загрузки обновления.
801	UPD_OLD_ERR_LOAD_FLASH	Ошибка при записи в program flash.
802	UPD_OLD_ERR_	Прочая ошибка обновления.

803	NOVALIDSERIAL	Некорректный серийный номер.
820	UA_USER_DOESN_T_EXIST	Учётная запись с таким номером отсутствует на устройстве.
821	UA_USER_SUSPEND	PIN пользователя заблокирован. (ещё можно восстановить с помощью PUK)
822	UA_CHANGE_PIN_DIVERGENCE	Введенные PIN не совпадают.
823	UA_CHANGE_PIN_INCORRECT	Введен неверный текущий PIN. при смене PIN
824	UA_CHANGE_PUK_DIVERGENCE	Введенные PUK не совпадают.
825	UA_CHANGE_PUK_INCORRECT	Введен неверный текущий PUK. при смене PUK
830	UA_TLS_CERT_WARNING	Ошибка загрузки личного сертификата TLS. Будет использован сертификат первичного подключения
831	UA_TLS_CERT_ERROR	TLS-сертификат не загружен. Для ТЛС будет использоваться первичный
832	UA_ENTER_PREMATURE	Преждевременная попытка использования кода доступа.
833	UA_USER_ACTIVE	Команда не предназначена для активных учётных записей.
834	UA_CHANGE_PIN_REQUIRED	Для разрешения доступа к функции необходимо сменить транспортный PIN-код.
835	UA_OBSOLETE_FUNCTION	Команда не поддерживается для учётных записей, инициализированных в соответствии с новым протоколом.
850	GEC_REVOKED	Сертификат отозван.
851	GEC_CORRUPTED_CRL	Испорчен файл со списком отозванных сертификатов.
852	GEC_EXPIRED	Время действия истекло, не наступило, или выходит за пределы времени действия издателя.
855	GEC_OBSOLETE_CRL	На устройстве уже установлен более новый список отзыва сертификатов.
856	GEC_WRONGUSAGE	Неверное использование сертификата (например, ТЛС для ЭП итд).
857	CREATEOBJ_INTERNAL_ERROR	Внутренняя ошибка при создании PKCS10-RequestInfo.
863	CREATEOBJ_INVALIDDN	Структура с данными пользователя не соответствует формату
864	CREATEOBJ_INVALIATTRS	Структура с атрибутами пользователя не соответствует формату
865	GEC_NO_FREE_PAIR_CELLS	Нет свободных слотов для создания пары 'Секретный ключ - Сертификат'
866	GEC_NO_FREE_USERCRT_CELLS	Превышение допустимого числа личных сертификатов на один запрос.
867	GEC_RQST_NOT_FOUND	Не найден запрос для входящего сертификата.
868	GEC_HOLD	Действие сертификата приостановлено.

900	FUNCTION_NOT_IMPLEMENTED	Функция не реализована в этой версии прошивки.
901	MISMATCH_TOKEN_AND_PC_SW	Прошивка токена и используемое на ПК ПО несовместимы.
902	NO_PRIMARY_CERTS	На устройстве отсутствует сертификат первичного ТЛС-подключения или транспортный
950	SC_OK_DONE	Софт-криптофункция выполнена успешно, результат не выгружен.
951	CRYPTO_INVAL	Софт-криптофункция получила неверные параметры вычисления хэша/ЭП.
952	SC_CRYPTO_FAIL	Выполнение софт-криптофункции завершилось неуспешно (например, подпись не сошлась).
999	VALIDATION_ERROR	Ошибка формата одного или нескольких полей запроса
1030	CCIDSSL_CANT_OPEN_SOCKET_ON_TlsS	Не удаётся открыть сокет на ТЛС-сервере.
1300	GEC_CHAIN	Не удалось построить цепочку доверия.
-1102	CCIDSSL_ACCESSDENY	Бизнес-система не выбрана, или вход на токен не произведён.
-1103	CCIDSSL_OUTOFSTREAMS	Нет свободных ТЛС-потоков.
-1104	CCIDSSL_WANTMOREDATA	Недостаточно данных с TLS-сервера.
-1105	CCIDSSL_WANTOUTDATA	Токену есть что отправить на TLS-сервер.
-1106	CCIDSSL_SSLERROR	Ошибка разбора пакета TLS.
-1107	CCIDSSL_SSLCLOSE	TLS-сервер закрыл соединение.
-1108	CCIDSSL_PROTOERR	Ошибка протокола TLS.
-1109	CCIDSSL_NOCIPHER	Сервер не имеет совместимых наборов шифров.
-1110	CCIDSSL_NOCACERT	0030. Не удается построить цепочку доверия до сертификата сервера.
-1111	CCIDSSL_NOOWN	0040. Сертификат сервера отсутствует или повреждён.
-1112	CCIDSSL_NOCERT	0050. Ошибка получения сертификата сервера.
-1113	CCIDSSL_BADPEM	0060. Поврежден контейнер сертификата сервера.
-1114	CCIDSSL_BADSIGN	0070. Подпись сертификата сервера не верна.
-1115	CCIDSSL_CERTEXPIRED	0080. Срок действия сертификата сервера истёк.
-1116	CCIDSSL_CERTREVOCED	0090. Сертификат сервера находится в списке отзыва.
-1117	CCIDSSL_NOTTRUSTED	0100. Не удалось проверить подлинность сертификата сервера.
-1118	CCIDSSL_UNOCERT	0110. Сервер не получил сертификат клиента.
-1119	CCIDSSL_UBADPEM	0120. Сервер получил поврежденный контейнер сертификата клиента.
-1120	CCIDSSL_UBADCERT	0130. Сервер получил повреждённый сертификат клиента.

-1121	CCIDSSL_UCERTEXPIRED	0140. Сервер получил сертификат клиента с истекшим сроком действия, или истек срок действия CRL издателя.
-1122	CCIDSSL_UCERTREVOCED	0150. Сервер обнаружил сертификат клиента в списке отзыва.
-1123	CCIDSSL_UNOTTRUSTED	0160. Серверу не удалось проверить подлинность сертификата клиента.
-1124	CCIDSSL_UNRECGNAME	0170. Запрашиваемая прикладная система неизвестна серверу.
-1125	CCIDSSL_RHSREQ	Запрос рехендшейка от сервера.
-1126	CCIDSSL_BROKENKEY	0180. Несовпадение ключей клиента и сервера.
-1127	CCIDSSL_NOMOREOWNCERTS	При SwitchCert сертификаты клиента закончились
-1128	CCIDSSL_OK_WITHKEYEXPORT	Успешное завершение хендшейка для СОФТ-TLS соединения.
-1129	CCIDSSL_BROKEN_SOFTKEYS	Неправильные ключи переданы токеном на ПК при установке СОФТ-ТЛС соединения
-1202	CCIDSSL_TUN_ALREADY_EXIST	Данный туннель уже активирован.

1. Логин

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	LOGIN	id функции	
user	NUMBER	1 байт	номер учетной записи	
pin	PIN_STR	6 байт	пин-код	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID0>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="LOGIN"&user="1"&pin="123456"

Выход:

Параметр	Тип	Ограничение	Назначение
sid2	HANDLE	34 байт	идентификатор сессии
user	NUMBER	1 байт	номер залогинившейся учётной записи
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
sid2="hagstey2u7dksiu3746xga8kagstev5209"&user="1"&retcode="1"

Примечания:

- ***Для устройств с экраном передача PIN-кода с ПК (и само наличие в каком-либо виде PIN-кода на ПК) является недопустимым. Поэтому при логине на экранное устройство строка «pin» должна быть пустой, иначе устройство вернёт код ошибки.***
- ***Если при использовании экранного устройства во входных аргументах команды номер учётной записи «***user***» равен «-1», то устройство предложит пользователю самому выбрать учётную запись из списка. Иначе пользователю будет предложено ввести пин-код именно для заданной в явном виде аргументом «***user***» учётной записи.***
- ***Если при вызове команды «LOGIN» на устройстве уже был залогинен пользователь, то его сессия автоматически прекращается. *Существует также команда «LOGIN1», полностью совместимая с «LOGIN» по аргументам, которая в случае обнаружения на устройстве работающего пользователя, не «выкидывает» его, а возвращает соответствующий код ошибки.***

2. Генерация

2.1. Генерировать ключевую пару и квалифицированный запрос на сертификат PKCS10

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	CREATE_PAIR_EX_ID	id функции	
dn	BASE64	1536 байт бинарных данных (размер base64 для их кодировки не регламентирован)	Владелец сертификата (поле subject из запроса/ будущего сертификата) в виде собранной ASN.1 структуры.	
attr	BASE64	1536 байт бинарных данных (размер base64 для их кодировки не регламентирован)	Атрибуты расширенного запроса на сертификат (для размещения в секции "extensionRequest") в виде фрагмента ASN.1 структуры.	
attr2	BASE64	1536 байт бинарных данных (размер base64 для их кодировки не регламентирован)	Прочие атрибуты запроса в виде фрагмента ASN.1 структуры.	Опционально (кроме [500.00-500.59], [501.00-501.07])
req_type	ENUM	1..7	3 (или наследн. 1) - личный ЭП; 4 (или наследн. 2) - личный TLS; 6 - первичный TLS; 7 - транспортный ЭП;	Значения больше двух поддерживаются прошивками 505+
pk_alg	ENUM	2,3,4	Формат криптоопераций: 1 - RSA 2 - ГОСТ-2001 3 - ГОСТ-2012-256 4 - ГОСТ-2012-512	пп. 3 и далее поддерживаются прошивками 500+.NT4+
hash_alg	ENUM	1,2,3	Стандарт подсчёта хеша: 1 - ГОСТ-3411_1994_256 2 - ГОСТ-3411_2012_256 3 - ГОСТ-3411_2012_512	500+.NT4+; Опц.; По умолч.: "1"
enc_alg	ENUM	1, 2	Алгоритмы шифрования: 1 - ГОСТ 28147_1989, CRYPTOPRO_PARAM_A_256, 2 - ГОСТ 28147_1989, PARAM_Z_256	500+.NT4+; Опц.; По умолч.: "1".
paramset	ENUM	1..9	Наборы параметров эллиптической кривой: 1 - SET_A, 2 - SET_B, 3 - SET_C, 4 - SET_XchA (стандартный для СКЗИ Крипто Про, Верба), 5 - SET_XchB, 6 - SET_A_TK26_256, 7 - SET_A_TK26_512, 8 - SET_B_TK26_512, 9 - SET_C_TK26_512	500+.NT4+; Опц; По умолч.: "2".
sig_alg	ENUM	2	1(MD5_RSA)/ 2(GostR341194_GostR34102001)	
ow	ENUM	1, 2	1 - Если в хранилище нет места, то автоматически удалить самый старый из прошлых ключей; 2 - Не удалять; если в хранилище нет места, то вернуть код ошибки	

signed_attrs	ENUM	0, 1	Гарантированная подлинность блока "signed_attrs" в подписях, сделанных этой парой (за счёт формирования её внутри токена). (Признаком этого будет ОИД "1.3.6.1.4.1.13242.5.2.10" в "политиках" сертификата). "0" - не гарантировать; "1" - гарантировать.	550.235+ Опционально, по умолчанию: "0"
charset	ENUM	3	1(ASCII)/2(UTF-16)/3(UTF-8)	

***Примечание**: значения аргументов, выделенные серым цветом, обозначают либо заложенные на будущее варианты, либо устаревшие и далее не используемые. Работа со значениями аргументов, окрашенных серым, в ближайших релизах поддерживаться не будет!*

Пояснение.
dn – имя владельца сертификата, на который формируется запрос. Сущность Name как опредеяется пунктом 9.1.3 спецификации X.501 (т.е. значение поля Inssuer сертификата в виде «как есть»). Переданное аргументом dn имя будет без изменений вставлено в запрос сертификат и далее в сам сертификат на УЦ. Аргумент на токене проверяется исключительно на валидность ASN.1 структуры.
attr – произвольно число сущностей типа Attribute как определено RFC5280, последовательно помещенных в передаваемый массив. Аргумент на токене проверяется на валидность ASN.1 структуры. Дополнительно производится контроль по следующим атрибутам:

OID	Назначение OID	Требование в OID
1.2.643.100.111	Наименование используемого средства ЭП	Должен отсутствовать (значение ставится токеном автоматически)
1.2.643.100.113.X	Класс криптосредства	Должен отсутствовать (значение ставится токеном автоматически)
1.3.6.1.4.1.13242.5.2.10 внутри секции с OID "2.5.29.32"	Гарантированная подлинность блока "signed_attrs" в секции "certificatePolicies"	Должен отсутствовать (значение ставится в соответствии с аргументом "signed_attrs" функции)
1.2.643.3.123.3.1	Идентификатор ключа Бикрипт (аналогично полю "biid" для функции "CREATE_PAIR_ID")	Должен присутствовать
1.2.643.3.123.3.4	Идентификатор бизнес системы	Должен присутствовать
1.2.643.3.123.3.5		Должен отсутствовать (значение ставится токеном автоматически)
2.5.29.15	KeyUsage	Если присутствует, то должен иметь установленным биты: - для ЭП-сертификатов: digitalSignature(0) и nonRepudiation(1)
2.5.29.37	ExtendedKeyUsage	Если присутствует, то должен содержать идентификатор 2.5.29.37.0 (AnyUsage), либо: - для ЭП-сертификатов: "ID_KP_EMAILPROT" (1.3.6.1.5.5.7.3.4). - для ТЛС-сертификатов: "ID_KP_CLIENTAUTH" (1.3.6.1.5.5.7.3.2). Подробнее: https://tools.ietf.org/html/rfc5280

В дополнение к переданным атрибутам токеном будут добавлены дополнительные системные атрибуты.
attr2 – произвольно число сущностей типа Attribute как определено RFC5280, последовательно помещенных в передаваемый массив.

Выход:

Параметр	Тип	Ограничение	Назначение
obj_id	HANDLE	8 байт	id объекта
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
obj_id="abcdefgh"&retcode="1"

2.2. Удалить ключевую пару и соответствующие ей сертификаты, или один личный сертификат.

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	DELETE_OBJ_ID	id функции	
obj_id	HANDLE	8 байт	id объекта	
puk	PIN_STR	12 байт	пак-код	

Пример:

POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="DELETE_OBJ_ID"&obj_id="abcdefgh"

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	2 байт	код возврата ф-ии

Пример:

HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode="1"

3. Установить сертификат

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SET_CERT_D_ID	id функции	
data	PEMDER	15360 байтов бинарных данных (размер base64, требуемый для их кодировки, не регламентирован)	данные сертификата в формате PEM	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID0 или SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="SET_CERT_D_ID"&data="<certdata>"

Выход:

Параметр	Тип	Ограничение	Назначение
obj_id	HANDLE	8 байт	id объекта
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
obj_id="abcdefgh"&retcode="1"

Примечание: если устанавливается сертификат ROOT или УЦ, то сертификат устанавливается как новый объект, возвращается новый идентификатор. Если устанавливаемый сертификат соответствует уже существующему объекту (секретный+открытый ключ), то возвращается идентификатор существующего объекта, а сертификат привязывается к нему.

4. Установить текущий используемый объект

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SET_ACT_OBJ_ID	id функции	
obj_id	HANDLE	8 байт	id объекта	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="SET_ACT_OBJ_ID"&obj_id="abcdefgh"

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode="1"

Пояснения:

- ***Если id объекта соответствует ТЛС-сертификату, то этот ТЛС-сертификат будет в первую очередь предъявляться при установке связи с ТЛС-сервером.***
- ***Если это сертификат ЭП, то прошивками версий ниже 500 он будет указан в списке личных сертификатов на первом месте. (Это используется системой СББОЛ для идентификации пользователя).***
- ***Прошивки 500 и выше будут показывать данные этого сертификата на странице администрирования в правом верхнем углу. (на позицию сертификата в списке личных сертификатов это не повлияет).***
- ****В случае последующего удаления активного ЭП-сертификата, для показа в верхнем углу страниц администрирования будет взят другой личный ЭП-сертификат, время начала действия самое позднее из всех установленных, а при отсутствии личных ЭП-сертификатов будет показана строка: "Личный сертификат ЭП не установлен".***
- *****Для того, чтобы отключить показ текста про личный ЭП-сертификат в углу страниц администрирования, нужно подать в качестве obj_id строку: "DONT_SHOW_EP_CERT". Чтобы включить показ данных ЭП-сертификата, нужно подать в качестве obj_id строку: "SHOW_EP_CERT".***

5. Установить CRL

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SET_CRL_D_ID	id функции	
data	PEMDER	15360 байтов бинарных данных (размер base64, требуемый для их кодировки, не регламентирован)	данные CRL в PEM	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID0 или SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="SET_CRL_D_ID"&data="<crl_pem_data>"

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode="1"

6. Установить обновление

6.1. Начать установку частей обновления

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	INIT_UPD_ID	id функции	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="INIT_UPD_ID"

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode="1"

6.2. Установить часть обновления (поле)

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SET_UPD_D_ID	id функции	
data	PEMDER	15360 байтов бинарных данных (размер base64, требуемый для их кодировки, не регламентирован)	данные обновления в base64	
blocknum	NUMBER	2^32 - 1	номер блока	опциональное, если присутствует, то проверяется. Нумерация начинается с 1

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="SET_UPD_D_ID"&data="<upddata>"

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode="1"

6.3. Завершить установку частей обновления

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	FIN_UPD_ID	id функции	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="FIN_UPD_ID"

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode="1"

7. Получить список объектов (сертификаты ЭП, TLS, корневые и УЦ)

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	GET_OBJ_LIST_ID	id функции	
obj_type	ENUM	1 байт	0 - сертификаты ЭП; 1 - сертификаты TLS; 2 - корневые сертификаты; 3 - запрос ЭП; 4 - запрос TLS; 5 - сертификаты УЦ; 6 - первичные ТЛС; 7 - транспортные ЭП;	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="GET_OBJ_LIST_ID"&obj_type="1"

Выход:

Параметр	Тип	Ограничение	Назначение
data	STRING_A	4096 байт	список id, сепаратор;
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
data="abcdefga;abcdefgh;abcdefgk"&retcode="1"

8. Получить данные объекта

8.1. Получить сертификат X.509/запрос на сертификат PKCS10

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	GET_OBJ_CERT_D_ID	id функции	
obj_id	HANDLE	8 байт	id объекта	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="GET_OBJ_CERT_D_ID"&obj_id="abcdefgh"

Выход:

Параметр	Тип	Ограничение	Назначение
data	PEM	15360 байтов бинарных данных (размер base64, требуемый для их кодировки, не регламентирован)	данные запроса/сертификата в формате PEM
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
data="<cert_data>"&retcode="1"

10. Информация о системе

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	GET_SYS_INFO_ID	id функции	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID0 или SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="GET_SYS_INFO_ID"

Выход:

Параметр	Тип	Ограничение	Назначение
devcfg	STRING_A	128 байт	информация о конфигурации
rnginitdate	STRING_A	19 байт	HH:MM:SS dd.mm.YYYY
serial	STRING_A	12 байт	серийный номер
buildid	NUMBER	8 байт	номер сборки
curuser	NUMBER	1..6	текущая учетная запись (1 – 6)
pin_must_be_changed	ENUM	0, 1	После инициализации токена требуется изменить PIN, иначе будет недоступна часть функционала. (опционально)
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
devcfg="IC0_A16P0000L_C1_VT500BU03NT05"&rnginitdate="11:22:33 04.05.2016"&serial="TLS00001234A"&curuser="1"&buildid="100"&pin_must_be_changed="1"&retcode="1"

Примечания: Строка «devcfg» devcfg строится согласно следующему шаблону:

```
<Сведения об аппаратной части токена, не несущие полезной нагрузки для программы-клиента и гарантированно не содержащие подстроки "_VT">_VT<Версия прошивки>[<Имя Модуля 1><Версия Модуля 1>][<Имя модуля 2><Версия Модуля 2>]...[<Имя Модуля N><Версия Модуля N>]
```

где N не больше восьми.

- Версия прошивки представляет из себя число, состоящее цифр без знака в количестве от одной до пяти. Они заканчиваются концом строки, либо буквой, являющейся первой буквой имени одного из модулей прошивки.
- "Модуль" - это одна или несколько дополнительных программно-аппаратных функций (свойств) данного токена, информирующих прикладную программу о её новых возможностях или ограничениях при работе с токеном.
- Имя модуля состоит строго из двух латинских букв (с учётом регистра) и от одной до пяти цифр версии (возможно, в этом количестве присутствуют ведущие нули). Цифры заканчиваются концом строки, либо буквой, являющейся первой буквой имени очередного модуля прошивки.
- Строка конфигурации может содержать информацию о модулях, количеством до восьми.

- Имена модулей не сортированы.
- Порядок следования информации о модулях не имеет значения.
- От версии к версии прошивки не гарантируется сохранение порядка следования информации о модулях.

На момент написания настоящей документации, в различных прошивках использовались следующие модули в различных комбинациях:

- BU - кнопка
- KA - поддержка кодов аутентификации (не используется в СББОЛ)
- NT - новые технологии
- SC - экран с точскрином

Если в строке ответа значение ключа "pin_must_be_changed" равно "1", то пользователь должен изменить PIN-код по умолчанию, назначенный при инициализации устройства. Пока пользователь не задаст новый PIN-код, ему не будут доступны функции создания запросов на личные сертификаты и вход на бизнес-системы. Версии прошивки ранее 507 могут не выдавать в ответе этот ключ.

11. Найти объект по полям сертификата

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	GET_CERT_ID	id функции	
serial	BASE64	xx байт	серийный номер в base64	необяз. с v.500+
c	STRING	xx байт	поле УЦ [Country]	необяз. с v.500+
st	STRING	xx байт	поле УЦ [STate]	необяз. с v.500+
l	STRING	xx байт	поле УЦ [Location]	необяз. с v.500+
org	STRING	xx байт	поле УЦ [ORGanization]	необяз. с v.500+
ou	STRING	xx байт	поле УЦ [OrgUnit]	необяз. с v.500+
cn	STRING	xx байт	поле УЦ [CommonName]	необяз. с v.500+
ea	STRING	xx байт	поле УЦ [E-mAil]	необяз. с v.500+
openkey	BASE64	xx байт	открытый ключ в base64	с v.500+; необяз.
g	STRING	xx байт	поле УЦ []	с v.500+; необяз.
inn	STRING	xx байт	поле УЦ [ИНН]	с v.500+; необяз.
ogrn	STRING	xx байт	поле УЦ [ОГРН]	с v.500+; необяз.
snils	STRING	xx байт	поле УЦ [СНИЛС]	с v.500+; необяз.
t	STRING	xx байт	поле УЦ [должность]	с v.500+; необяз.
obj_type	NUMBER	0..2^32-1	Типы объектов, среди которых искать (возможны комбинации по схеме «или»): 16-корневые сертификаты 32-сертификаты УЦ 64-сертификаты ЭП 128-сертификаты TLS 512-запросы ЭП 1024-запросы TLS	с v.500+; необяз.; при отсутствии или значении меньше шестнадцати ищет среди всех типов по возрастанию

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="GET_CERT_ID"&serial="MDBDQTEyMzQ1Ng%3D%3D"&st="Moscow"&org="Silver Sparks"&ou="OAO"&ea="mail@ssparks.ru"

Выход:

Параметр	Тип	Ограничение	Назначение
data	HANDLE	8 байт	id объекта
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
data="abcdefga"&retcode="1"

Примечание 0: Формулировка "необяз. с v.500+" означает, что с версиях 500+ оно необязательно, но в более ранних версиях могло быть обязательным.
Формулировка "с v.500+; необяз." означает, что оно поддерживается версиями 500+ и является необязательным,

Примечание 1: Если какое-либо поле в запросе не задано, то оно игнорируется при поиске, и в выборке присутствуют объекты с любым значением этого поля. Если оно задано как "", то в выборке присутствуют только те объекты, у которых оно пустое.

Примечание 2: По историческим причинам, не все заявленные в этой таблице поля участвуют в поиске. Из соображений совместимости с работающими в проме решениями, некоторые поля при поиске игнорируются (подробнее могут рассказать разработчики). В случае необходимости поддержать все поля, не нарушая совместимости с существующими решениями, в интерфейс будет добавлена новая функция поиска, а эта приобретёт статус наследной.

12. Получить URL для перехода из бизнес-системы

С помощью этой команды можно получить URL-адрес страницы администрирования токена, на который можно вернуться по окончании работы с бизнес-системой.

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	GET_SID_URL_ID	id функции	
urlnum	NUMBER	1 байт	номер SID в URL	**«0»** содержит SID0, приводит к разлогину и выводит на страницу выбора учётной записи **«1»** содержит SID1, не приводит к разлогину и выводит на главную страницу данной учётной записи (на список бизнес-систем).

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="GET_SID_URL_ID"&urlnum="1"

Выход:

Параметр	Тип	Ограничение	Назначение
data	STRING	128 байт	URL перехода
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
data="http://`localhost:28016/vpnkeylocal/<SID1>/main.html`"&retcode="1"

14. Расширение API для работы с толстым клиентом

14.1. Получение списка доступных учетных записей

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	GET_PIN_LIST	id функции	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID0 или SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="GET_PIN_LIST"

Выход:

Параметр	Тип	Ограничение	Назначение
pin	STRING_A	8 байт	номер учетной записи
user	STRING_A	8 байт	имя учетной записи
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
pin="PIN 1"&user="1"&pin="PIN 2"&user="2"&retcode="1"

---

*) В случае, если действие PIN приостановлено (т.е. требуется восстановление PIN-кода на странице администрирования или при помощи команды «CH_PIN_BY_PUK_ID»), в начале имени присутствует слово «SUSPEND_».

**) В случае, если учётная запись была отформатирована командой «FORMAT_PIN_ID», и PIN-код после этого не задавался командой «CH_PIN_BY_PIN_ID», (то есть, функциональность учётной записи ограничена), в начале имени присутствует слово «PURGED_».

Например:

pin="PIN 1"&user="1"&pin="SUSPEND_PIN 2"&user="2"&pin="PURGED_PIN 3"&user="3"&retcode="1"

14.2. Открытие сессии с бизнес-системой по умолчанию

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	LOGIN	id функции	
user	NUMBER	1 байт	номер учетной записи	
pin	PIN_STR	6 байт	данные пин-кода	

Пример:

POST http://localhost:28016/vpnkeylocal/<SID0 или SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="LOGIN"&user="1"&pin="597346"

Выход:

Параметр	Тип	Ограничение	Назначение
sid2	HANDLE	34 байт	идентификатор сессии
retcode	ENUM	2 байт	код возврата ф-ии

Пример:

HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
sid2="<SID2>"&retcode="1"

14.3. Открытие сессии с последующим выбором бизнес-системы

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	LOGIN1	id функции	
user	NUMBER	1 байт	номер учетной записи	
pin	PIN_STR	6 байт	данные пин-кода	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID0 или SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="LOGIN1"&user="1"&pin="597346"

Выход:

Параметр	Тип	Ограничение	Назначение
sid2	HANDLE	34 байт	идентификатор сессии
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
sid2="<SID2>"&retcode="1"

14.4. Получение списка доступных бизнес-систем

Команда выдаёт только «толстые» (не доступные на главной странице Вэб-интерфейса токена) бизнес-системы.

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	GET_BS_LIST	id функции	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1
Content-Length: xxx\r\n\r\n
id="GET_BS_LIST"

Выход:

Параметр	Тип	Ограничение	Назначение
bsid	NUMBER	1 байт	идентификатор системы
name	STRING_A	34 байт	имя системы
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
bsid="0"&name="SBBOL"&bsid="2"&name="SBBOL2"&retcode="1"

14.5. Определение (установка) используемой бизнес-системы

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SET_BS_USE	id функции	
bsid	NUMBER	1 байт	идентификатор системы	

Пример:

POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1
Content-Length: xxx\r\n\r\n
id="SET_BS_USE"&bsid="0"

Выход:

Параметр	Тип	Ограничение	Назначение
sid2	HANDLE	34 байт	идентификатор сессии
retcode	ENUM	2 байт	код возврата ф-ии

Пример:

HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
sid2="<SID2>"&retcode="1"

15. Расширение API для работы с резервной бизнес-системой

15.1. Открытие сессии с резервной бизнес-системой

Для резервной бизнес-системы предусмотрен отдельный путь HTTP-сервера.

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	START_NEW_SYSTEM	id функции	
name	STRING_A	128	Fully qualified DNS hostname	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="START_NEW_SYSTEM"&name="infocrypt.ru"

Выход:

Пример:
HTTP/1.0 302 FOUND\r\n
Location: http://localhost:28016/auxch/?infocrypt_sid=<SID2>

16. ЭП в CMS

Для получения ЭП в формате CMS необходимо выполнить следующие действия:

1. Проинициализировать устройство (INIT_SIGN_H_ID). Устройство вернёт хендл криптооперации - последовательность из 8 букв и цифр. Её нужно будет использовать при вызове каждой очередной функции данной криптооперации.
2. Внести данные для подписи (SET_SIGN_DATA_H_ID – 1 или более вызовов).
3. Подтвердить готовность рассчитать подпись (CALC_SIGN_H_ID). В случае некорректного задания данных статус принимает значение FAILED. В случае использования устройства с возможностью интерактивного подтверждения расчета ЭП статус принимает значение WAITING до того момента, пока пользователь не нажмет клавишу подтверждения. В случае отрицательного решения пользователя статус принимает значение REJECTED. При положительном решении (или использовании токена без кнопки) – статус принимает значение COMPLETE.
4. Проверить статус устройства в контексте ЭП (GET_CTX_INFO_H_ID). Данную операцию необходимо выполнять в цикле, если статус принял значение WAITING, до момента, пока он не примет отличное значение.
5. Получить данные CMS (GET_SIGN_CMS_H_ID). Операция должна быть выполнена и вернёт CMS с ЭП, если статус принял значение COMPLETE. В случае вызова этой функции до того, как статус принял значение COMPLETE, устройство вернёт ошибку "Операция не завершена". В любом случае, после отработки этой функция хендл операции станет невалидным.

16.1. Инициализация ЭП

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	INIT_SIGN_H_ID	id функции	
datasize	NUMBER64	2^63 - 1	размер данных для ЭП	
hascert	NUMBER	1 байт	максимальное количество сертификатов из цепочки доверия, которое вложить в CMS	"-1" - вложить все сертификаты, в т.ч. корневой.
hasdata	ENUM	0, 1	"0" - не вкладывать в CMS подписанные данные; "1" - вложить в CMS подписанные данные	
mode	ENUM	1, 2	Стандарт криптоопераци: 1 - хеш данных считается на ПК "стартом.exe", токен только генерирует подпись хеша, 2 - хеш данных считается токеном, подпись генерируется им же	500+; Опционально, по умолчанию: "2"
obj_id	HANDLE	8 байт	id сертификата для подписи	503+ нечётные и 508.3+; Опционально. Рекомендуется к использованию.
name	BASE64	128 символов UTF8 (размер BASE64 для их кодировки не регламентирован)	Для экранных токенов. Имя подписываемого документа для отображения на экран.	500+; Опционально.
signed_attrs	ENUM	0, 1	"0" - не создавать в CMS секцию "signed_attrs" "1" - создавать в CMS секцию "signed_attrs"	507.01+ (нечётные) Опционально, по умолчанию: "1"

Пример:

POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="INIT_SIGN_H_ID"&datasize="66144"&hascert="1"&hasdata="0"&obj_id="zxcvbnma"

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	2 байт	код возврата ф-ии
ctx_handle	HANDLE	8 байт	хендл контекста операции

Пример:

HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode="1"&ctx_handle="abcdefgh"

Примечание: Для работы по ГОСТ 2012 и выше необходимо указать хендл сертификата на этапе вызова команды "INIT_SIGN_H_ID", т.к. параметры криптоопераци берутся из сертификата. Если сертификат не будет указан при вызове INIT_SIGN_H_ID, то его будет необходимо указать в конце, однако, в этом случае подписание будет произведено по ГОСТ-3411_1994_256, и если указанный в конце сертификат содержит другие параметры подписания, то операция будет завершена с ошибкой.

16.2. Загрузить порцию данных для ЭП

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SET_SIGN_DATA_H_ID	id функции	
data	BASE64	Для прошивок <550, mode=2: 15360 байтов бинарных данных; Для прошивок 500+, mode=1 и 550+: 16777216 байтов бинарных данных Размер base64, требуемый для их кодировки, не регламентирован.	данные для подписи в base64;	для прошивок версий ниже 505, если эта порция данных не последняя, размер декодированных данных должен быть кратен 8
blocknum	NUMBER	2^32 - 1	номер посылки	опциональное, если присутствует, то проверяется. Нумерация начинается с 1
ctx_handle	HANDLE	8 байт	хендл контекста операции	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="SET_SIGN_DATA_H_ID"&data="<data>"&blocknum="8"&ctx_handle="abcdefgh"

Выход:

Параметр	Тип	Ограничение	Назначение
data_length	NUMBER64	2^63 - 1	суммарное количество данных, переданных для подписи (включая переданное данным вызовом)
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
data_length="1234"&retcode="1"

16.3. Вычисление ЭП

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	CALC_SIGN_H_ID	id функции	
ctx_handle	HANDLE	8 байт	хендл контекста операции	
obj_id	HANDLE	8 байт	id сертификата для подписи	НАСЛЕДНОЕ! Не рекомендуется к использованию для прошивок, поддерживающих "obj_id" в функции INIT_SIGN_H_ID

Пример:

POST http://localhost:28016/vpnkeylocal/<<SID1 или SID2>>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="CALC_SIGN_H_ID"&ctx_handle="abcdefgh"

Выход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
retcode	ENUM	2 байт	код возврата ф-ии	

Пример:

HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode="1"

<u>Примечание:</u> Поле "obj_id" не рекомендуется использовать с командой "CALC_SIGN_H_ID", т.к. идентификатор сертификата рекомендуется передавать на этапе вызова команды "INIT_SIGN_H_ID". Подробнее смотрите примечание к "INIT_SIGN_H_ID".

16.4. Получить ЭП в формате CMS

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	GET_SIGN_CMS_H_ID	id функции	
ctx_handle	HANDLE	8 байт	хендл контекста операции	

Пример:
POST http://localhost:28016/vpnkeylocal/<<SID1 или SID2>>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="GET_SIGN_CMS_H_ID"&ctx_handle="abcdefgh"

Выход:

Параметр	Тип	Ограничение	Назначение
head	BASE64	16384 байт декодированных данных	данные начала CMS в base64
suffix	BASE64	16384 байт декодированных данных	данные суффикса CMS в base64
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
head="<head_data>"&suffix="<suffix_data>"&retcode="1"

<u>Пояснение:</u> Если при инициализации подписания было указано, что выходной файл должен содержать подписанные данные, то клиент должен сам сформировать выходной cms-файл следующим образом: Сначала записать возвращённое этой функцией поле "head", потом сами подписанные данные (чтобы не гонять их через токен туда-обратно), потом возвращённое этой функцией поле "suffix".
Если при инициализации не собирались включать подписанные данные в cms-файл, то клиент должен просто записать сначала поле "head", потом поле "suffix".

<u>Примечание:</u> Функцию необходимо вызывать только после того, как "GET_CTX_INFO_H_ID" вернёт статус "COMPLETE". В случае вызова "GET_SIGN_CMS_H_ID" до получения "COMPLETE" от "GET_CTX_INFO_H_ID" токен вернёт ошибку, операция подписания будет отменена, и дальнейшее использование её хендла будет невозможным.
****Вызов функции "GET_SIGN_CMS_H_ID" до окончания загрузки данных или до получения "COMPLETE" от "GET_CTX_INFO_H_ID" можно использовать специально для прерывания операции подписания, например, в случае отказа пользователя от операции уже после её начала.***

16.5. Информация о контексте

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	GET_CTX_INFO_H_ID	id функции	
ctx_handle	HANDLE	8 байт	хендл контекста операции	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="GET_CTX_INFO_H_ID"&ctx_handle="abcdefgh"

Выход:

Параметр	Тип	Ограничение	Назначение
status	ENUM	0 - READY, 1 - ACTIVE, 2 - COMPLETE, 3 - WAITING, 4 - FAILED, 5 - REJECTED	состояние контекста устройства
data_length	NUMBER64	2^63 - 1	количество данных, переданных для подписи
sign_num	NUMBER	2^32 - 1	число уже рассчитанных за время данной сессии ЭП
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
status="1"&data_length="1234"&sign_num="7"&retcode="1"

17. Визуализация подписываемых данных

Требования к документам.

За основу формата документа принимается используемый в данное время в системе СББОЛ. Документ состоит из нескольких разделов. Начало раздела имеет вид

<имя раздела>:

Обязательные разделы - “ATTRIBUTES” и “FIELDS” и опциональный “TABLES”. Разделы должны идти именно в такой последовательности. Раздел “TABLES” не визуализируется. Содержимое разделов “ATTRIBUTES” и “FIELDS” представляет собой множество строк вида:

<название поля>=<значение>\n (обратите внимание! “\n”, а не “\r\n”!)

Прибор переносит слишком длинные строчки автоматически, однако существует возможность принудительного разрыва строки. Для этого используйте “\r\n” в документе.

Кодировка документа — UTF8. Поддерживаются диапазоны символов от 0020h до 007Eh и от D090h до D191h включительно.

Раздел “ATTRIBUTES” обязательно содержит поля DocType, DocTypeVersion, DocNumber, DocDate, содержащие тип документа, версию его формата и дату создания.

В целях поддержки формата SWIFT, который предусматривает в рамках одного поля обработку до 6 строк длиной в 35 символов, разделенных символом перевода строки, токен визуализирует в каждой ячейке до 12 строк, (до 26 символов в каждой строке, т.к. 35 символов не помещаются на 1 строку).

Технические ограничения: Если из-за большого размера подписываемого документа токен не может отобразить его исходный код, то подпись не может быть выдана с типом "визуализировано", и выдаётся с типом "подтверждено кнопкой". Если тип подписи "подтверждение кнопкой" недопустим для данного документа, то при его загрузке следует установить опциональный параметр "vis_required" в "1". Тогда, если документ слишком велик, то вместо автоматического переключения типа подписи на "подтверждено кнопкой", токен выдаст ошибку "TMPL_IMPOSSIBLE_VISUALISE". Операция подписания при этом прервётся.

Принцип работы.

Процесс подписи с визуализацией проводится над массивом из нескольких документов. Визуальная форма на экране представляет собой непрерывную последовательную форму, представленную блоками, соответствующими документам из передаваемого массива. Для каждого документа при визуализации на экран выводится форма, представленная шаблоном визуализации и заполненная данными подписываемого документа. Каждому типа документа соответствует собственный шаблон. Шаблоны не хранятся на устройстве. Шаблон под конкретный тип документа загружается непосредственно перед одним или последовательностью документов данного типа в процессе передачи массива единоразово подписываемых документов.

Каждый блок имеет заголовок, состоящий из названия, номера и даты документа и таблицу из двух столбцов — название и значение поля

Формат шаблонов.

Шаблон устанавливает принадлежность к типу и версии формата визуализируемого при его помощи документа. Устанавливает состав визуализируемых из исходного документа данных, человеко-читаемые название документа и визуализируемых полей, положение разделителя столбцов.

Формат шаблона аналогичен формату документа. Состоит из обязательных разделов “HEADER”, “BODY” и опционального “FOOTER”.

Заголовок задает соответствие типу документа - поля DocType и DocTypeVersion должны соответствовать значению одноименных полей из раздела “ATTRIBUTES” подписываемого документа. LimiterPos задает положение разделителя полей названия и значения по горизонтали в пикселах.

Раздел “BODY” построчно содержит визуализируемые поля в формате

<имя поля>=<описание поля>.

Раздел “FOOTER” выводится после блока документов и содержит итоговые данные по блоку однотипных документов. Построчно содержит визуализируемые поля в формате

<имя поля>=<описание поля>. Выводимое значение поля является суммой значений данного поля по всем обрабатываемым документам. Данное поле должно иметь числовое значение (допустимые символы 0123456789). Кроме полей из документа, возможно использование ключевого слова COUNT – в этом случае будет выведено количество обработанных документов в серии, относящейся к текущему шаблону.

Шаблон документа помещается в контейнер, защищенный ЭП/кодом аутентификации (а зависимости от типа криптографии на устройстве) с возможностью проверить целостность/подлинность содержимое контейнера на ключах устройства.

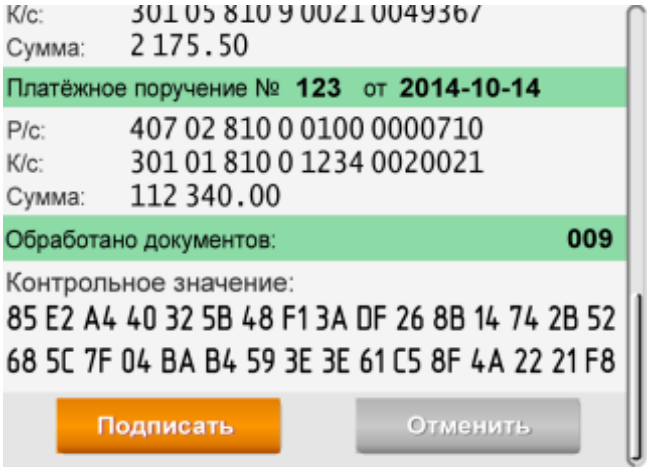
Пример шаблона

HEADER:
DocType=PayDocCur
DocTypeVersion=1.0
DocName=Платежное поручение
LimiterPos=60
BODY:
ChargesAccount=P/c
DocAccount=K/c
DocSum=Сумма
FOOTER:
COUNT=Обработано документов

Пример документа.

ATTRIBUTES:
GlobalID=62169f8b-9e79-4af6-847d-f21d3a3f8f3c
DocType=PayDocCur
DocTypeVersion=1.0
DocDate=2014-10-14
DocNumber=123
FIELDS:
DocAccount=40702810001000000710
DocSum=112340.00
DocSumCurrency=978
BenefBankBIC=AAGOCHZ1XXX
BenefBankBICType=SWIFT
ChargesAccount=30101810012340020021
ChargesBankBIC=044525219
PayerBankBIC=QWER12345

Пример результата визуализации



Кастомизация шаблонов.

Прошивка прибора позволяет задавать пользовательские цвета для отображения документов, выбирать тип и размер шрифтов (из записанных в прошивке) и ширину вертикальных границ до и после каждой строки.

Это делается с помощью ключей «BgColorH, BgColorL1, BgColorL2, BgColorR1, BgColorR2, ForeColorH, ForeColorL1, ForeColorL2, ForeColorR1, ForeColorR2, FontSizeH, FontSizeL, FontSizeR, FontH, FontL, FontR, YAfterH, YAfterL, YAfterR, YBeforeH, YBeforeL, YBeforeR», которые могут опционально располагаться в секции «HEADER» шаблона.

Все цвета указываются в 16-ричном виде в RGB;

Все ширины вертикальных границ указываются в пикселах и могут иметь значения от 0 до 10 включительно.

Размеры шрифтов (для прошивки 500.23) могут иметь один из трёх возможных вариантов: 10, 20 или 30.

Имена шрифтов (для прошивки 500.23) могут иметь значения только: "Gost", "FreeSet", "LycidaCon" (моноширинный).

Внешний вид всех указанных шрифтов приведён в таблице:

Название шрифта \ размер	**10**	**20**	**30**
FreeSet	0123456789 ABC...xyz	0123456789 ABC...xyz	0123456789 ABC...xyz
Gost	0123456789 ABC...xyz	0123456789 ABC...xyz	0123456789 ABC...xyz
LycidaCon	0123456789 ABC...xyz	0123456789 ABC...xyz	0123456789 ABC...xyz

Если значение поля не указано в шаблоне, используется значение по умолчанию.

Пример шаблона со значениями по умолчанию для прошивки 500.23:

```
HEADER:
DocType=...
DocTypeVersion=...
DocName=...
LimiterPos=60
BgColorH=8CDAA6
BgColorL1=FFFFFF
BgColorL2=E0E0E0
BgColorR1=FFFFFF
BgColorR2=E0E0E0
ForeColorH=0
ForeColorL1=FF0000
ForeColorL2=FF0000
ForeColorR1=0000A0
ForeColorR2=0000A0
FontSizeH=20
FontSizeL=20
FontSizeR=20
FontH=Gost
FontL=FreeSet
FontR=LycidaCon
YAfterH=4
YAfterL=3
YAfterR=3
YBeforeH=4
YBeforeL=5
YBeforeR=5
BODY...
```

17.1. Инициализировать интерактивное подписание (только для устройств с экраном, с прошивками с чётными номерами)

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	INIT_TMPL_H_ID	id функции	
obj_id	HANDLE	8 байт	id сертификата для подписи	Опционально

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="INIT_TMPL_H_ID"&obj_id="zxcvbnma"

Выход:

Параметр	Тип	Ограничение	Назначение
ctx_handle	HANDLE	8 байт	хендл контекста операции
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode="1"&ctx_handle="asdfghjk"

Примечание: Принимаем hascert=1; hasdata=0.

17.2. Установить шаблон (только для устройств с экраном, с прошивками с чётными номерами)

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SET_TMPL_H_ID	id функции	
data	PEMDER	15360 байтов бинарных данных (размер base64, требуемый для их кодировки, не регламентирован)	Шаблон для отображения документа в CMS-пакете, подписанном сертификатом УЦ, до которого токен может выстроить цепочку доверия	
ctx_handle	HANDLE	8 байт	хендл контекста операции	

Пример:

POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="SET_TMPL_H_ID"&data="<tmpldata>"&ctx_handle="asdfghjk"

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	2 байт	код возврата ф-ии

Пример:

HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode="1"

Пояснения:

- ***Шаблоны предназначены для целых классов документов (одни - для платёжных поручений, другие - для зарплатных ведомостей итд).***
- ***Шаблоны изготавливаются в банке и редко заменяются.***
- ***Шаблоны не участвуют в формировании подписи. Они только описывают, какие поля из загружаемых документов показывать на экран, и стили их показа.***
- ***После загрузки одного шаблона, можно загружать подряд несколько документов того типа, для визуализации которого предназначен шаблон.***
- ***Шаблоны не сохраняются устройством. Устройство имеет понятие «текущий шаблон» - это шаблон, загруженный последним. Используя его и только его, устройство будет пытаться отобразить загружаемые в него документы.***

17.3. Загрузить документ для визуализации (только для устройств с экраном, с прошивками с чётными номерами)

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SET_TMPL_DOC_H_ID	id функции	
data	BASE64	16 мб, обёрнутых в base64. (размер base64, требуемый для их кодировки, не регламентирован)	Один документ для визуализации	
ctx_handle	HANDLE	8 байт	хендл контекста операции	
vis_required	ENUM	{0, 1}	0 = В случае нехватки памяти на визуализацию исходных данных документа, разрешается автоматически переключить тип подписания с "визуализировано" на "подтверждено кнопкой" с занесением информации об этом в CMS с подписью. 1 = Необходимо выдать подпись для этого документа с типом "визуализировано", исключив автоматическое переключение типа на "подтверждено кнопкой". При нехватке памяти возвращается ошибка.	Опционально; по умолчанию: "0".

Пример:

POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="SET_TMPL_DOC_H_ID"&data="<docdata>"&ctx_handle="asdfghjk"

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	2 байт	код возврата ф-ии

Пример:

HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode="1"

Пояснения:

- ***Если документ будет иметь тип, не соответствующий шаблону, или не будет содержать каких-нибудь полей, записанных в шаблоне как обязательные для отображения, то функция вернёт ошибку добавления документа.***
- ***В случае успешной загрузки, документ будет сразу же показан на экране.***
- ***В случае несоответствия показанного на устройстве документа ожидаемому, пользователь будет иметь возможность нажать кнопку «Отмена». О решении пользователя отменить подписание, клиент узнает с кодом возврата в ответ на ближайшую функцию из последовательности визуализации, поэтому для ПО клиента имеет смысл периодически вызывать GET_CTX_INFO_H_ID.***

17.4. Закончить загрузку визуализируемых документов и рассчитать подпись (только для устройств с экраном, с прошивками с чётными номерами)

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	CALC_SIGN_H_ID	id функции	
obj_id	HANDLE	8 байт	id сертификата для подписи	НАСЛЕДНОЕ, НЕ РЕКОМЕНДУЕТСЯ К ПРИМЕНЕНИЮ!
ctx_handle	HANDLE	8 байт	хендл контекста операции	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="CALC_SIGN_H_ID"&ctx_handle="abcdefgh"

Выход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
retcode	ENUM	2 байт	код возврата ф-ии	

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode="1"

Пояснение: После успешного выполнения этой команды следует вызывать GET_CTX_INFO_H_ID до тех пор, пока статус не станет отличным от «WAITING», после чего можно забрать CMS(-ы) с подписями документов командой «GET_TMPL_SIGN_CMS_H_ID».
Если клиенту необходимо прервать ожидание подписи, можно вызвать «GET_TMPL_SIGN_CMS_H_ID» до установки контекста в состояние готовности. В этом случае вернётся ошибка, операция будет прервана, а хендл - инвалидирован.

17.5. Получить ЭП визуализированных документов в формате CMS (только для устройств с экраном, с прошивками с чётными номерами)

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	GET_TMPL_SIGN_CMS_H_ID	id функции	
ctx_handle	HANDLE	8 байт	хендл контекста операции	

Пример:

POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="GET_TMPL_SIGN_CMS_H_ID"&ctx_handle="abcdefgh"

Выход:

Параметр	Тип	Ограничение	Назначение
count	NUMBER	не регламентировано	Количество CMS-ов с подписями
cms...	BASE64	не регламентировано	Последовательность полей «cms...=<...>», содержащих полные подписи загруженных для визуализации документов в том же порядке, в котором они были загружены.
retcode	ENUM	2 байт	код возврата ф-ии

Пример:

HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
count="10"&cms1="<cms1_data>"&cms2="<cms2_data>"&...&retcode="1"

17.6. Подписание пакетов документов (только для прошивок с нечётными номерами)

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SIGN_PACKET	id функции	
obj_id	HANDLE	8 байт	id сертификата для подписи	
spec	PEMDER	16000 байтов бинарных данных. (размер base64 для их кодировки не регламентирован)	CMS с XML-правилами обработки подписываемого пакета	Опционально; прошивками <550 на "старой" аппаратной платформе не используется
vis_required	ENUM	0, 1	1 = Необходимо выдать подписи с типом "визуализировано", исключив автоматическое переключение типа на "подтверждено кнопкой" при нехватке памяти на визуализацию исходных данных.	Опционально; по умолчанию: "0".
mode	ENUM	1, 2	Стандарт криптооперации: 1 - хеш данных считается на ПК "стартом.exe", токен только генерирует подпись хеша, 2 - хеш данных считается токеном, подпись генерируется им же (по умолчанию)	Опционально; по умолчанию: "2".
docs_count	NUMBER	1..2^32 - 1	Количество документов в пакете	
doc1...doc4294967295	BASE64	не регламентировано	Документы на подпись	

Пример:

POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: 100500\r\n\r\n
id="SIGN_PACKET"&obj_id="asdf2534"&spec="77u/PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz1VVEYtOCI/Pg08VGV..."&docs_count="10"&doc1="77u/QVRUUklCVV..."&doc2="..."&doc...

Выход:

Параметр	Тип	Ограничение	Назначение
count	NUMBER	не регламентировано	Количество CMS-ов с подписями
cms1...cms4294967295	BASE64	не регламентировано	Последовательность полей «cms...=<...>», содержащих полные подписи загруженных для визуализации документов с теми же номерами, с которыми они были загружены.
retcode	ENUM	2 байт	код возврата ф-ии

Пример:

HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
count="10"&cms1="<cms1_data>"&cms2="<cms2_data>"&...&retcode="1"

Пояснение 0. В качестве параметра «spec» должны быть использованы специальные шаблоны в формате XML, описанные в документе «СББОЛ.VPNkeyTLS. Токен с экраном.Шаблоны.БТ.docx». Шаблоны, пригодные для команды «SET_TMPL_H_ID» не могут быть использованы для команды «SIGN_PACKET»!

Пояснение 1. Блок документов должен иметь следующий формат: «doc<номер документа>=<base64-закодированный документ>&doc<номер следующего документа>=<base64-закодированный следующий документ>итд...». Документы в блоке должны быть отсортированы по порядку и не перемежаться с другими параметрами. Номера, переданные в именах входных параметров «data...», будут в именах выходных параметров "cms...data". Например, "cms15_data", полученный из функции на выходе, будет подписью для данных, переданных на вход функции в блоке «doc15».

Пояснение 2. Если устройству недостаточно памяти для визуализации всех переданных документов, то оно возвращает параметром «retcode***» ошибку «TMPL_NOT_ENOUGH_MEMORY», а параметром «***count***» количество документов из данного списка, которое устройство могло бы визуализировать.***

18.1. Инициализация проверки ЭП

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	INIT_CHECK_H_ID	id функции	
mode	ENUM	1, 2	Стандарт криптооперации: 1 - хеш данных считается на ПК "стартом.exe", токен проверяет подпись только хеша, 2 - хеш данных считается токеном, подпись проверяется им же.	500+; Опционально, умолч.2
cert_data	PEMDER	15360 байтов бинарных данных (размер base64, требуемый для их кодировки, не регламентирован)	данные внешнего сертификата	НАСЛЕДНОЕ ! Прошивками 500+ игнорируется.
cms_data	PEMDER	15360 байтов бинарных данных (размер base64, требуемый для их кодировки, не регламентирован)	данные CMS ЭП для проверки.	503+ нечётные и 508.6+ чётные; Опционально. Рекомендуется к использованию.

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="INIT_CHECK_H_ID"

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	2 байт	код возврата ф-ии
ctx_handle	HANDLE	8 байт	хендл контекста операции

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode="1"&ctx_handle="abcdefgh"

<u>Примечание:</u> Если одновременно имеется и отдельно указанный сертификат, и сертификат внутри CMS, то последний для проверки не используется.

18.2. Данные для проверки ЭП

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SET_CHECK_DATA_H_ID	id функции	
data	BASE64	Для прошивок <550, mode=2: 15360 байтов бинарных данных; Для прошивок 500+, mode=1 и 550+: 16777216 байтов бинарных данных Размер base64, требуемый для их кодировки, не регламентирован.	данные для проверки подписи;	для прошивок версий ниже 505, если эта порция данных не последняя, размер декодированных данных должен быть кратен 8
blocknum	NUMBER	2^32 - 1	номер посылки	опциональное, если присутствует, то проверяется. Нумерация начинается с 1
ctx_handle	HANDLE	8 байт	хендл контекста операции	

Пример:
POST http://localhost:28016/vpnkeylocal/<<SID1 или SID2>>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="SET_CHECK_DATA_H_ID"&data="<data>"&blocknum="8"&ctx_handle="abcdefgh"

Выход:

Параметр	Тип	Ограничение	Назначение
data_length	NUMBER64	2^63 - 1	суммарное количество данных, переданных для проверки подписи (включает то, что передано данным вызовом)
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
data_length="1234"&retcode="1"

18.3. Проверка ЭП

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	CHECK_SIGN_H_ID	id функции	
ctx_handle	HANDLE	8 байт	хендл контекста операции	
cert_data	BASE64	15360 байтов бинарных данных (размер base64, требуемый для их кодировки, не регламентирован)	данные сертификата в base64	НАСЛЕДНОЕ! Не рекомендуется к использованию! Прошивками 500+ игнорируется.
cms_data	BASE64	15360 байтов бинарных данных (размер base64, требуемый для их кодировки, не регламентирован)	данные CMS ЭП для проверки в base64	НАСЛЕДНОЕ! Не рекомендуется к использованию для прошивок, поддерживающих "cms_data" в функции INIT_CHECK_H_ID

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="CHECK_SIGN_H_ID"&ctx_handle="abcdefgh"

Выход:

Параметр	Тип	Ограничение	Назначение
data	ENUM ('0'/'1')	1 байт	атавизм
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
id_data="<id_data>"&retcode="1"

Пояснение: ***Реткод "1" в ответ именно на эту функцию означает верность подписи.***

Примечание: ***Агрументы "cert_data" и "cms_data" рекомендуется передавать не команде "CHECK_SIGN_H_ID", а команде "INIT_CHECK_H_ID", т.к. только таким способом можно будет работать по ГОСТ-ам 2012+.***

19. Шифрование данных в CMS

Для шифрования в формате CMS необходимо выполнить следующие действия:
1. Проинициализировать устройство (INIT_ENCIPHER_ID).
2. Внести данные сторонних сертификатов (ADD_RECIPIENT_Y_ID – 1 или более вызовов).
3. Внести данные внутренних сертификатов (ADD_RECIPIENT_X_ID – 1 или более вызовов).
Если значение идентификатора объекта отсутствует — используется текущий активный сертификат.
4. Внести данные для шифрования (ENCIPHER_ID – 1 или более вызовов).
5. Получить данные начала и окончания CMS (GET_ENCIPHER_CMS_ID).
6. Сформировать CMS с использованием полученных начала и окончания, а также зашифрованных данных в соответствии со схемой:
Шифрованные_данные_в_формате_CMS = PEM_head + base64(head +[Recipient_info] + suffix + зашифрованные данные) + PEM_tail

19.1. Инициализация шифрования

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	INIT_ENCIPHER_ID	id функции	
datasize	NUMBER64	2^63 - 1	размер данных для шифрования	
mode	ENUM	1, 2	Стандарт криптооперации: 1 - токен только генерирует сессионный ключ, данные шифруются на ПК "стартом.exe" 2 - токен генерирует сессионный ключ и шифрует данные.	Только нечётные версии выше 503 и чётные от 508.07 и выше. Опционально; по умолчанию: "2".

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="INIT_ENCIPHER_ID"&datasize="2048"

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	2 байт	код возврата ф-ии
ctx_handle	HANDLE	8 байт	хендл контекста операции

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode="1"&ctx_handle="abcdefgh"

19.2. Добавление стороннего сертификата получателя

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	ADD_RECIPIENT_Y_ID	id функции	
data	BASE64	15360 байтов бинарных данных (размер base64, требуемый для их кодировки, не регламентирован)	данные стороннего сертификата в base64	
ctx_handle	HANDLE	8 байт	хендл контекста операции	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="ADD_RECIPIENT_Y_ID"&data="<cert_data>"&ctx_handle="abcdefgh"

Выход:

Параметр	Тип	Ограничение	Назначение
data	BASE64	2048 байтов декодированных данных (размер base64, требуемый для их кодировки, не регламентирован)	часть данных для формирования CMS, содержит информацию о получателе
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
data="<recipient_info>"&retcode="1"

19.3. Добавление собственного сертификата

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	ADD_RECIPIENT_X_ID	id функции	
obj_id	HANDLE	8 байт	id собственного сертификата	
ctx_handle	HANDLE	8 байт	хендл контекста операции	

Пример:

POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="ADD_RECIPIENT_X_ID"&obj_id="<cert_obj_id>"&ctx_handle="abcdefgh"

Выход:

Параметр	Тип	Ограничение	Назначение
data	BASE64	2048 байтов декодированных данных (размер base64, требуемый для их кодировки, не регламентирован)	часть данных для формирования CMS, содержит информацию о получателе
retcode	ENUM	2 байт	код возврата ф-ии

Пример:

HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
data="<recipient_info>"&retcode="1"

19.4. Зашифрование данных

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	ENCIPHER_ID	id функции	
data	BASE64	Для прошивок <550, mode=2: 15360 байтов бинарных данных; Для прошивок 500+, mode=1 и 550+: 16777216 байтов бинарных данных Размер base64, требуемый для их кодировки, не регламентирован.	данные для зашифрования;	для прошивок версий ниже 505, если эта порция данных не последняя, размер декодированных данных должен быть кратен 8
ctx_handle	HANDLE	8 байт	хендл контекста операции	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="ENCIPHER_ID"&data="<data_to_encipher>"&ctx_handle="abcdefgh"

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	2 байт	код возврата ф-ии
enciphered_blob	BASE64	Для прошивок <550, mode=2: 16384 байтов бинарных данных; Для прошивок 500+, mode=1 и 550+: 16793600 байтов бинарных данных Размер base64, требуемый для их кодировки, не регламентирован.	часть зашифрованных данных для CMS

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode="1"&enciphered_blob="<data_blob>"

19.5. Получить части для CMS

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	GET_ENCIPHER_CMS_ID	id функции	
ctx_handle	HANDLE	8 байт	хендл контекста операции	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="GET_ENCIPHER_CMS_ID"&ctx_handle="abcdefgh"

Выход:

Параметр	Тип	Ограничение	Назначение
head	BASE64	2048 байтов декодированных данных	данные начала CMS в base64
suffix	BASE64	2048 байтов декодированных данных	данные суффикса CMS в base64
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
head="<head_data>"&suffix="<suffix_data>"&retcode="1"

20. Расшифрование данных в CMS

Для расшифрования в формате CMS необходимо выполнить следующие действия:
1. Проинициализировать устройство (INIT_DECIPHER_ID) путем передачи части с заголовком.
2. Внести данные для расшифрования (DECIPHER_ID – 1 или более вызовов) и получить в ответ расшифрованные данные

20.1. Инициализация расшифрования

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	INIT_DECIPHER_ID	id функции	
head	BASE64	Для прошивок нечётных версий ниже 505 и чётных ниже 508: 2048 байтов бинарных данных; Для остальных: 8192 байта бинарных данных; (размер base64, требуемый для их кодировки, не регламентирован)	данные начала CMS в base64	
obj_id	HANDLE	8 байт	id сертификата для расшифрования	с версии 500 игнорируется
mode	ENUM	1, 2	Стандарт криптооперации: 1 - токен только генерирует сессионный ключ, данные шифруются на ПК "стартом.exe" 2 - токен генерирует сессионный ключ и шифрует данные.	Только нечётные версии выше 503 и чётные от 508.07 и выше. Опционально; по умолчанию: "2".

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="INIT_DECIPHER_ID"&head="<cms_head_containing_blob>"&obj_id="zxcvbnmm"

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	2 байт	код возврата ф-ии
ctx_handle	HANDLE	8 байт	хендл контекста операции
body_displ	NUMBER	2^32 - 1	размер (смещение) начала CMS в чистом виде (до кодирования base64)

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode="1"&ctx_handle="abcdefgh"&body_displ="1456"

20.2. Расшифровывание данных

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	DECIPHER_ID	id функции	
data	BASE64	Для прошивок <550, mode=2: 15360 байтов бинарных данных; Для прошивок 500+, mode=1 и 550+: 16777216 байтов бинарных данных Размер base64, требуемый для их кодировки, не регламентирован.	данные для расшифрования, начиная со смещения "body_displ" от начала CMS в чистом виде (до кодирования base64);	для прошивок версий ниже 505, если эта порция данных не последняя, размер декодированных данных должен быть кратен 8
ctx_handle	HANDLE	8 байт	хендл контекста операции	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="DECIPHER_ID"&data="<data_to_decipher>"&ctx_handle="abcdefgh"

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	2 байт	код возврата ф-ии
deciphered_blob	BASE64	не превышает размер поданного на вход поля "data"	часть расшифрованных данных

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode="1"&deciphered_blob="<data_blob>"

21. Установить конфигурацию бизнес-систем

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SET_MC_D_ID	id функции	
data	BASE64	15360 байтов бинарных данных (размер base64, требуемый для их кодировки, не регламентирован)	данные конфигурации в CMS-пакете, подписанном сертификатом УЦ, до которого токен может выстроить цепочку доверия	

Пример:

POST http://localhost:28016/vpnkeylocal/<SID0 или SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="SET_MC_D_ID"&data="<mc_cms_data>"

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	2 байт	код возврата ф-ии

Пример:

HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode="1"

Примечание 1: количество БС в загружаемом CMS-е должно для прошивок 399+ не превышать 10, а для прошивок 500+ может быть произвольным, при условии, что размер текстового файла с описанием бизнес-систем, помещённого внутрь CMS, не превышает 3000 байтов.

Примечание 2: Блок с описанием бизнес-систем должен:

- ***Представлять из себя текстовый файл в кодировке Windows-1251.***
- ***Начинаться на сигнатуру "ICMSC" + CRLF (что соответствует С-строке "\r\n" или же символам с кодами 0x0d 0x0a).***
- ***Далее должны следовать строки-записи, каждая из которых соответствует одной бизнес-системе. Про каждую бизнес-систему должны быть записаны следующие параметры:***
  - ***название бизнес-системы, как его будет видеть пользователь;***
  - ***название web-ресурса, используемое расширением Server Name Indication протокола TLS, транслируемое TLS-сервером;***
  - ***доменное имя или IP-адрес TLS-сервера (прошивки 550+ поддерживают также опционально указанный номер порта);***
  - ***параметр видимости и возможности использования "тонкими" бизнес-системами: 1 – "видимая" (для использования "тонкими" бизнес-системами), 2 – "невидимая" (для использования "толстыми" бизнес-системами).***
  - ***остальные параметры должны игнорироваться.***
- ***Все параметры являются обязательными и должны следовать в перечисленном порядке и разделяться символом ";".***
- ***Каждая строка-запись о бизнес-системе должна заканчиваться последовательностью CRLF. Весь файл конфигурации должен также заканчиваться дополнительной последовательностью CRLF.***

22. Смена кодов

22.1. Смена PIN по PUK коду

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	CH_PIN_BY_PUK_ID	id функции	
user	NUMBER	1 байт	номер учетной записи	
puk	PIN_STR	12 байт	данные пак-кода	
pin	PIN_STR	6 байт	данные нового пин-кода	
pin2	PIN_STR	6 байт	повтор нового пин-кода	с версии 500+; опциональный;

Пример:
POST http://localhost:28016/vpnkeylocal/<SID0>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="CH_PIN_BY_PUK_ID"&user="1"&puk="597346123456"&pin="123456"

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode="1"

Примечания:

- ***Для устройств с экраном передача PIN- и PUK-кодов с ПК (и само наличие в каком-либо виде этих кодов на ПК) является недопустимым. Поэтому при смене кодов на экранном устройстве строки «puk», «pin» и «pin2» должны быть пустыми, иначе устройство вернёт код ошибки.***
- ***Если при использовании экранного устройства во входных аргументах команды номер учётной записи «***user***» равен «-1», то устройство предложит пользователю самому выбрать учётную запись для смены кода из списка. Иначе пользователю будет предложено сменить пин-код именно для заданной в явном виде аргументом «***user***» учётной записи.***

22.2. Смена PIN по PIN коду

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	CH_PIN_BY_PIN_ID	id функции	
user	NUMBER	1 байт	номер учетной записи	
pin_old	PIN_STR	6 байт	данные старого пин-кода	
pin_new	PIN_STR	6 байт	данные нового пин-кода	
pin_new2	PIN_STR	6 байт	повтор нового пин-кода	с версии 500+; опциональный;

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="CH_PIN_BY_PIN_ID"&user="1"&pin_old="123456"&pin_new="098765"

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode="1"

Примечания:

- ***Для устройств с экраном передача PIN--кодов с ПК (и само наличие в каком-либо виде этих кодов на ПК) является недопустимым. Поэтому при смене кодов на экранном устройстве строки «pin_old», «pin_new» и «pin_new2» должны быть пустыми, иначе устройство вернёт код ошибки.***
- ***Если при использовании экранного устройства во входных аргументах команды номер учётной записи «user» равен «-1», то устройство предложит пользователю самому выбрать учётную запись для смены кода из списка. Иначе пользователю будет предложено сменить код именно для заданной в явном виде аргументом «user» учётной записи.***

23. Сценарии работы с устройствами VPNKeyTLS

23.1. Установление сессии на Устройстве

а) [опционально] Определить SID0 (входит в состав URL в файле "sslgate.url" на флеш-диске Устройства):
```
[InternetShortcut]
URL=http://localhost:28016/vpnkeylocal/<SID0>/
```

б) Вызвать метод API для получения списка доступных учётных записей:

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID0 или без SID>/ HTTP/1.1
Content-Type: text/html;
Content-Length: xxx

id=GET_PIN_LIST
```
===пример запроса; окончание===

===пример ответа; начало===
```
pin="PIN 1"&user="1"&pin="PIN 2"&user="2"&retcode="1"
```
===пример ответа; окончание===

в) Вызвать метод API для открытия сессии, указав необходимое имя и PIN:

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID0 или без SID>/ HTTP/1.1
Content-Type: text/html;
Content-Length: xxx

id=LOGIN&user=1&pin=597346
```
===пример запроса; окончание===

===пример ответа; начало===
```
sid2="1234qwertyu5678"&retcode="1"
```
===пример ответа; окончание===

г) Полученный идентификатор сессии SID2 (далее "SID") необходимо использовать для следующих сценариев.

23.2. Установление сессии на Устройстве и канала TLS с поддержкой мультисистемности

а) Вход на Устройство произведён, SID известен. Получаем с Устройства список установленных на нём TLS-систем.

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID>/ HTTP/1.1
Content-Type: text/html;
```

```
Content-Length: xxx

id=GET_BS_LIST
```
===пример запроса; окончание===

===пример ответа; начало===
```
bsid="0"&name="SBBOL"&bsid="2"&name="SBBOL2"&retcode="1"
```
===пример ответа; окончание===

б) Из полученного списка TLS-систем необходимо выбрать для дальнейшей работы и указать ее идентификатор

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID>/ HTTP/1.1
Content-Type: text/html;
Content-Length: xxx

id=SET_BS_USE&bsid=0
```
===пример запроса; окончание===

===пример ответа; начало===
```
retcode="1"
```
===пример ответа; окончание===

в) После успешного выполнения последней команды все HTTP-запросы, отправленные на http://localhost:28016/***/ (где "***" не равно "vpnkeylocal" и "auxch"), будут транслироваться по защищённому TLS-каналу на сервер выбранной TLS-системы.

23.3. Формирование ключевой пары

а) Выполнить действия по сценарию «Установление сессии на Устройстве», получить в результате SID.

б) Сформировать ASN.1-структуры: "dn" (с данными субъекта ключевой пары), "attr" (с желаемыми атрибутами запрашиваемого сертификата) и "attr2" (с прочими атрибутами запрашиваемого сертификата).

в) Вызвать метод API для формирования ключевой пары и выдачи её идентификатора ("хендла"):

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID>/ HTTP/1.1
Content-Type: text/html;
Content-Length: xxx

id=CREATE_PAIR_EX_ID&dn=MIIBkjENMAsGA1UEAwwESFNCQzEVMBMGA1UEBAwM0KLRj
9C%2F0LrQuNC9MSowKAYDVQQqDCHQlNC20Y3RhCDQktC10L3QtdC00LjQutGC0L7QstC4
0YcxCzAJBgNVBAYTAlJVMRUwEwYDVQQHDAzQntC80YHQutCw0Y8xGDAWBgNVBAgMDzU1I
NCe0LzRgdC60LDRjzErMCkGA1UECQwi0J3QsNCx0LXRgNC10LbQvdCw0Y8g0YPQu9C40Y
bQsCwgMTENMAsGA1UECgwESFNCQzEkMCIGA1UECwwb0J7RgtC00LXQuyDQvtC%2F0LXRg
NCw0YbQuNC5MTAwLgYDVQQMDCfQodGC0LDRgNGI0LjQuSDQvtC%2F0LXRgNCw0YbQuNC%
2B0L3QuNGB0YIxGjAYBggqhQMDgQMBARIMMDA3NzEwMzUzNjA2MRgwFgYFKoUDZAESDTE
wMjc3MzkyMDc0NjIxFjAUBgUqhQNkAxILMTEyMjMzNDQ1OTUxHjAcBgkqhkiG9w0BCQEW
D29AbG9jYWxob3N0LmNvbQ%3D%3D&req_type=4&pk_alg=3&hash_alg=2&enc_alg=2
&paramset=1&sig_alg=2&ow=2&attr=MA4GA1UdDwEB/wQEAwIE8A==&attr2=
```

===пример запроса; окончание===

===пример ответа; начало===
obj_id="XYZ5efgh"&retcode="1"
===пример ответа; окончание===

г) [опционально] Используя полученный идентификатор ("хендл") запроса выгрузить запроса из Устройства

===пример запроса; начало===
POST http://localhost:28016/vpnkeylocal/<SID>/ HTTP/1.1
Content-Type: text/html;
Content-Length: xxx

id=GET_OBJ_CERT_D_ID&obj_id=XYZ5efgh
===пример запроса; окончание===

===пример ответа; начало===
data="<request_data>"&retcode="1"
===пример ответа; окончание===

д) Выгруженные данные запроса можно передать в УЦ для выпуска сертификата.

е) Следует помнить, что все идентификаторы ("хендлы") объектов, выданные Устройством, актуальны только в течение данной сессии. После завершения данной и открытии следующей сессии все хендлы теряют актуальность. Поэтому, чтобы получить хендл этого запроса в одной из следующих сессий, нужно будет либо искать этот запрос среди выданных командой "GET_OBJ_LIST_ID", либо получать его командой поиска "GET_CERT_ID" (например, по открытому ключу).

23.4. Установка сертификата к уже сформированной ключевой паре

а) <Данные запроса на сертификат должны быть ранее выгружены с Устройства и переданы в УЦ. УЦ должен выпустить сертификат на этот запрос. На момент установки сертификат должен быть в наличии.>

б) Выполнить действия по сценарию «Установление сессии на Устройстве», получить в результате SID.

в) Вызвать метод API для установки выданного в УЦ сертификата на Устройство:

===пример запроса; начало===
POST http://localhost:28016/vpnkeylocal/<SID>/ HTTP/1.1
Content-Type: text/html;
Content-Length: xxx

id=SET_CERT_D_ID&data=-----BEGIN CERTIFICATE-----
%0d%0aMIID4jCCA5GgAwIBAgIIMDBDQTg2NjIwCAYGKoUDAgIDMFAxJDAiBgNVBAMMG9C
f%0d%0a0LXRh9Cw0YLRjCDQo9CmINCh0JEg0KDQpDEoMCYGA1UEIQwf0KLQtdGF0L3QuN
GH%0d%0a0LXRgdC60LjQuSDQutC70Y7RhzAeFw0xMzAxMjgwMDAwMDBaFw0xNzAxMjgwM
DAw%0d%0aMDBaMIIBAjELMAkGA1UEBhMCUlUxRTBDBgNVBAMMPNCf0LDQvdC60YDQsNGC
0L7Q%0d%0asiDQkNGA0YHQtdC90LjQuSDQkNC70LXQutGB0LDQvdC00YDQvtCy0LjRhzE
bMBkG%0d%0aA1UEBAwS0J/QsNC90LrRgNCw0YLQvtCyMUUwQwYDVQQqDDzQn9Cw0L3Qut

GA0LDR%0d%0agtC+0LIg0JDRgNGB0LXQvdC40Lkg0JDQu9C10LrRgdCw0L3QtNGA0L7Qs
tC40Ycx%0d%0aFjAUBgNVBAcMDdCc0L7RgdC60LLQsC4xIzAhBgNVBAkMGtGD0LsuINCS
0LDQstC4%0d%0a0LvQvtCy0LAsIDE5MQswCQYFKoUDZAMTADBjMBwGBiqFAwICEzASBgc
qhQMCAiMC%0d%0aBgcqhQMCAh4BA0MABECdhi5+fSybOCVCj6mw9zwYTrhuIBBO9rruCt
huTT6eL2EI%0d%0aV1uxHb5hkAzzEUwGjpyExesGZ6Fj/Mf83QsaCAJUo4IBlzCCAZMwH
QYDVR0OBBYE%0d%0aFLK1qLk8DBIaQqTiLeubs2x/ATAgMA8GA1UdEwEB/wQFMAMBAf8w
GAYDVR0lBBEw%0d%0aDwYEVR0lAAYHKoUDA3sFATAOBgNVHQ8BAf8EBAMCAcYwMwYDVR0
fBCwwKjAooCag%0d%0aJIYiaHR0cDovL3d3dy5zYnJmLnJ1L2NhLzAwMDB4NTA5LmNybD
BBBgcqhQMDewMB%0d%0aBDYMNDAwQ0E4NjYyadCi0LXQutGD0YnQsNGPINGC0LXRgdGC0
L7QstCw0Y8g0JDQ%0d%0akdChINCh0JEwIwYFKoUDZG8EGgwY0JHQuNC60YDQuNC/0YIt
0JrQodCRLdChMB0G%0d%0aA1UdIAQWMBQwCAYGKoUDZHEBMAgGBiqFA2RxAjBEBgUqhQN
kcAQ7MDkMAAwY0JHQ%0d%0auNC60YDQuNC/0YIt0JrQodCRLdChDAAMGdCk0KHQkSDQoN
CkINCh0KQvMTI0LTEw%0d%0aNTEwFAYHKoUDA3sDBAQJBgcqhQMDewUHMB8GA1UdIwQYM
BaAFFep7BWqnjB7CAzR%0d%0a99V6SDFMCkOhMAgGBiqFAwICAwNBAFLM8HwhTnIzrKS/
BBHZ4X5ETrpAhj4WM5Ni%0d%0abTwAHitgMUAJqxrKbGevkOmVzHqFQS1D9aorxVWwBk6
mrxzxJkI=%0d%0a-----END CERTIFICATE-----
===пример запроса; окончание===

===пример ответа; начало===
obj_id="ASDFbvcx"&retcode="1"
===пример ответа; окончание===

г) Полученный в поле "data" идентификатор ("хендл") установленного сертификата следует использовать в операциях, требующих его.

23.5. Поиск сертификата

а) Выполнить действия по сценарию «Установление сессии на Устройстве», получить в результате SID.

б) Вызвать метод API для поиска сертификата по серийному номеру или по серийному номеру + фрагментам DN:

===пример запроса; начало===
POST http://localhost:28016/vpnkeylocal/<SID>/ HTTP/1.1
Content-Type: text/html;
Content-Length: xxx

id=GET_CERT_ID&serial=dZI3swuXnzKS8w==&charset=1
===пример запроса; окончание===

===пример ответа; начало===
data="ASDFbvcx"&retcode="1"
===пример ответа; окончание===

в) Полученный в поле "data" идентификатор ("хендл") найденного сертификата следует использовать в операциях, требующих его.
*Если поле "data" отсутствует в ответе, или не заполнено, значит сертификат, соответствующий заданным критериям поиска, не найден на Устройстве.

23.6. Формирование ЭП в CMS

а) <На Устройстве должен быть установлен хотя бы один личный сертификат ЭП.>

б) Выполнить действия по сценарию «Установление сессии на Устройстве», получить в результате SID.

в) Выполнить действия по сценарию «Поиск сертификата» и узнать хендл личного сертификата ЭП.

г) Вызвать метод API для инициализации расчета ЭП:

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID>/ HTTP/1.1
Content-Type: text/html;
Content-Length: xxx

id=INIT_SIGN_H_ID&datasize=99990&hascert=1&hasdata=0&mode=1&obj_id=ASDFbvcx
```
===пример запроса; окончание===

===пример ответа; начало===
```
retcode="1"&ctx_handle="abcdefgh"
```
===пример ответа; окончание===

д) Вызвать (возможно более одного раза, если одним вызовом переданы не все данные для подписания) метод API для подачи данных для расчета ЭП:

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID>/ HTTP/1.1
Content-Type: text/html;
Content-Length: xxx

id=SET_SIGN_DATA_H_ID&blocknum=12&ctx_handle=abcdefgh&data=YWJjZGVmZ2g=
```
===пример запроса; окончание===

В ответ возвращается суммарный объем уже переданных на расчет данных:

===пример ответа; начало===
```
data_length="1212"&retcode="1"
```
===пример ответа; окончание===

е) Вызвать метод API для расчета ЭП:

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID>/ HTTP/1.1
Content-Type: text/html;
Content-Length: xxx

id=CALC_SIGN_H_ID&ctx_handle=abcdefgh
```
===пример запроса; окончание===

===пример ответа; начало===
```
retcode="1"
```
===пример ответа; окончание===

ж) Вызвать (возможно более одного раза) метод API для получения информации о контексте. Контекст определяет, возможно ли уже получить данные ЭП (статус

COMPLETE=2), или требуется подождать (статус WAITING=3).

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID>/ HTTP/1.1
Content-Type: text/html;
Content-Length: xxx

id=GET_CTX_INFO_H_ID&ctx_handle=abcdefgh
```
===пример запроса; окончание===

В ответ возвращается статус, суммарный объем уже переданных на расчет данных, количество рассчитанных ЭП:

===пример ответа; начало===
```
status="1"&data_length="99990"&sign_num="2"&retcode="1"
```
===пример ответа; окончание===

з) По достижении статуса COMPLETE (2) вызвать метод API для получения данных ЭП:

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID>/ HTTP/1.1
Content-Type: text/html;
Content-Length: xxx

id=GET_SIGN_CMS_H_ID&ctx_handle=abcdefgh
```
===пример запроса; окончание===

В ответ возращаются данные ЭП в base64:

===пример ответа; начало===
```
head="<head_base64>"&suffix="<suffix_base64>"&retcode="1"
```
===пример ответа; окончание===

и) Для формирования CMS-сообщения ПО клиента должно:
- записать в результат декодированное поле "head"
- если при инициализации ЭП поле "hasdata" было установлено в "1", то далее записать двоичные данные, подававшиеся для подписания
- дописать в результат декодированное поле "suffix"

Результат будет в формате "DER". При необходимости его можно преобразовать в формат PEM. Для этого нужно закодировать его в BASE64 и приписать в начале и в конце PEM-заголовки вида: "-----BEGIN CMS-----" и "-----END CMS-----".

23.7. Проверка ЭП в CMS

а) Выполнить действия по сценарию «Установление сессии на Устройстве», получить в результате SID.

б) Вызвать метод API для инициализации проверки ЭП:

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID>/ HTTP/1.1
Content-Type: text/html;
Content-Length: xxx

id=INIT_CHECK_H_ID&cms_data=MIIFvAYJKoZIhvcNAQcCoIIFrTCCBakCAQExDjAMB
```

```
ggqhQMHAQECAgUAMAsG
```
===пример запроса; окончание===

===пример ответа; начало===
```
retcode="1"&ctx_handle="abcdefgh"
```
===пример ответа; окончание===

в) Вызвать (возможно более одного раза, если одним вызовом переданы не все данные для подписания) метод API для подачи данных для проверки ЭП:

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID>/ HTTP/1.1
Content-Type: text/html;
Content-Length: xxx

id=SET_CHECK_DATA_H_ID&blocknum=12&ctx_handle=abcdefgh&data=YWJjZGVmZ
2g=
```
===пример запроса; окончание===

В ответ возвращается суммарный объем уже переданных на проверку данных

===пример ответа; начало===
```
data_length="1212"&retcode="1"
```
===пример ответа; окончание===

г) Вызвать метод API для проверки ЭП:

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID>/ HTTP/1.1
Content-Type: text/html;
Content-Length: xxx

id=CHECK_SIGN_H_ID&ctx_handle=abcdefgh
```
===пример запроса; окончание===

===пример ответа; начало===
```
retcode="1"
```
===пример ответа; окончание===

24. Форматирование учётных записей

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	FORMAT_PIN_ID	id функции	С версии 505+
user	NUMBER	-1, 1..5	номер учетной записи ("-1" - все учётные записи)	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID0>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="FORMAT_PIN_ID"&user="1"

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode="1"

Пояснение:

Функция "FORMAT_PIN_ID" форматирует заданную учётную запись. А именно, удаляет из памяти токена все личные ключи и запросы, сгенерированные заданной учётной записью, и все выданные для неё сертификаты. Для этой записи очищается PIN-код (т.е. вход на эту учётную запись становится возможным по вводу любого PIN), и запрещается создание ключевых пар и вход на бизнес-системы до смены пользователем ПИН-кода (т.е. успешного выполнения команды "CH_PIN_BY_PIN_ID"). При этом функция "GET_PIN_LIST" в ответе пишет про эту учётную запись: "PURGED_PIN", а на странице администрирования справа вверху отображается надпись: "Для включения полной функциональности учётной записи необходимо сменить PIN-код".

**** Учётные записи, отформатированные этой командой, не имеют PUK-кода, и при вводе неверного PIN более 3 раз устанавливается задержка перед каждой последующей попыткой ввода PIN-кода. В случае последующих неверных попыток задержка увеличивается с 15 минут до 2 часов. В случае 20 неверных попыток ввода PIN-кода подряд учётная блокируется навсегда без возможности восстановления.***

***** На токенах без экрана необходимо подтвердить выполнение этой команды в течение 3 секунд отключением токена от ПК. При этом токен будет мигать красным светодиодом. В случае неподтверждения выполнения этой команды в течение указанного времени, выполнение команды отменяется, и на ПК приходит код ошибки "Операция отменена пользователем"***

Наследные (не поддерживаемые и не рекомендуемые к использованию функции):

1.1. Генерировать пару ключей и запрос на сертификат PKCS10 (Не рекомендуется к использованию)

Вход:

Параметр	Тип	Ограничение*	Назначение	Дополнительно
id	ID_FORMAT	CREATE_PAIR_ID	id функции	
cn	STRING	128 байт	фио	
org	STRING	128 байт	название организации	
orgunit	STRING	64 байт	название подразделения	
email	STRING	32 байт	адрес эл.почты	
country	STRING	2 байт	страна	
rooccu	STRING	60 байт	должность	
bs_id	ENUM	1 байт	0 (Бизнес Онлайн)	
pk_alg	ENUM	1 байт	1(RSA)/2(ГОСТ 34.10-2001)	
req_type	ENUM	1 байт	1(ЭП)/2(TLS)	
sig_alg	ENUM	1 байт	1(MD5_RSA)\ 2(GostR341194_GostR34102001)	
label	STRING_A	8 байт	прикладное имя ключа	не используется
ow	ENUM	1 байт	1 (удалить неактивный)\ 2 (не удалять)	
biid	STRING	32 байт	идентификатор ключа Бикрипт	
charset	ENUM	1 байт	1(ASCII)\2(UTF-16)\3(UTF-8)	

Пример:

POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id=CREATE_PAIR_ID&cn=F_I_O&org=ORGname&orgunit=DEPTname&email=em@a.il&country=RU&rooccu=officer&pk_alg=1&req_type=0&sign_alg=0&label=<прикладное имя ключа>&ow=1&biid=<A8DE0012sИвановИИ>&charset=1

Выход:

Параметр	Тип	Ограничение	Назначение
obj_id	HANDLE	8 байт	id объекта
retcode	ENUM	2 байт	код возврата ф-ии

Пример:

HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
obj_id=”abcdefgh”&retcode=”1”

9. Информация о контексте (Не рекомендуется к использованию)

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	GET_CTX_INFO_ID	id функции	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id=GET_CTX_INFO_ID

Выход:

Параметр	Тип	Ограничение	Назначение
status	ENUM	1 байт: READY(0), ACTIVE(1), COMPLETE(2), WAITING(3), FAILED(4), REJECTED(5)	состояние контекста устройства
data_length	NUMBER	11 байт	количество данных, переданных для подписи
sign_num	NUMBER	11 байт	число уже рассчитанных за время данной сессии ЭП
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
status=”1”&data_length=”1234”&sign_num=”7”&retcode=”1”

13.1. Инициализация ЭП *(Не рекомендуется к использованию)*

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	INIT_SIGN_ID	id функции	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id=INIT_SIGN_ID

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode=”1”

13.2. данные для ЭП (Не рекомендуется к использованию)

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SET_SIGN_DATA_ID	id функции	
data	BASE64	Для прошивок 399+: 6144 URL-encoded байтов base64; Для прошивок 500+: 15360 байтов бинарных данных (размер base64, требуемый для их кодировки, не регламентирован)	данные для подписи в base64	
blocknum	NUMBER	11 байт	номер посылки	опциональное, если присутствует, то проверяется. Нумерация начинается с 1

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id=SET_SIGN_DATA_ID&data=<data2sign>&blocknum=8

Выход:

Параметр	Тип	Ограничение	Назначение
data_length	NUMBER	11 байт	суммарное количество данных после декодирования из base64, переданных для подписи (включая переданные данным вызовом)
accepted_b64_length	NUMBER	11 байт	количество данных, переданных для подписи в данном вызове, в base64 , которые удалось декодировать
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
data_length=”1234”& accepted_b64_length=”1586”&retcode=”1”

13.3. вычисление ЭП *(Не рекомендуется к использованию)*

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	CALC_SIGN_ID	id функции	
obj_id	HANDLE	8 байт	id сертификата для подписи	
ccdata	BASE64	Для прошивок 401+: исходые 32 байта контрольной суммы затем кодированные в base64;	данные контрольной суммы в base64	опциональное, если присутствует, то проверяется.

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id=CALC_SIGN_ID&obj_id=abcdefgh&ccdata=<data_cc_b64>

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode=”1”

13.4. получить ЭП *(Не рекомендуется к использованию)*

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	GET_SIGN_D_ID	id функции	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id=GET_SIGN_D_ID

Выход:

Параметр	Тип	Ограничение	Назначение
data	BASE64	256 байт ?	данные подписи в base64
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
data=”<sign_data>”&retcode=”1”

13.5. Инициализация проверки ЭП *(Не рекомендуется к использованию)*

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	INIT_CHECK_ID	id функции	

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id=INIT_CHECK_ID

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
retcode=”1”

13.6. данные для проверки ЭП (Не рекомендуется к использованию)

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SET_CHECK_DATA_ID	id функции	
data	BASE64	Для прошивок 399+: 6144 URL-encoded байтов base64; Для прошивок 500+: 15360 байтов бинарных данных (размер base64, требуемый для их кодировки, не регламентирован)	данные для проверки подписи в base64	
blocknum	NUMBER	11 байт	номер посылки	опциональное, если присутствует, то проверяется. Нумерация начинается с 1

Пример:
POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id=SET_CHECK_DATA_ID&data=<data2sign>&blocknum=8

Выход:

Параметр	Тип	Ограничение	Назначение
data_length	NUMBER	11 байт	суммарное количество данных, переданных для проверки подписи (включая переданное данным вызовом)
retcode	ENUM	2 байт	код возврата ф-ии

Пример:
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
data_length=”1234”&retcode=”1”

13.7. проверка ЭП (Не рекомендуется к использованию)

Вход:

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	CHECK_SIGN_ID	id функции	
cert_data	PEMDER	Для прошивок 399+: 6144 URL-encoded байтов base64; Для прошивок 500+: 15360 байтов бинарных данных (размер base64, требуемый для их кодировки, не регламентирован)	данные сертификата в base64	
sign_data	BASE64	88 байт	данные ЭП для проверки в base64	

Пример:

POST http://localhost:28016/vpnkeylocal/<SID1 или SID2>/ HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id=CHECK_SIGN_ID&cert_data=<certificate data>&sign_data=<signature data>

Выход:

Параметр	Тип	Ограничение	Назначение
data	ENUM ('0'/'1' - атавизм)	1 байт	данные проверки
retcode	ENUM	2 байт	код возврата ф-ии

Пример:

HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
id_data="<id_data>"&retcode="1"