ООО Фирма «ИнфоКрипт»

СРЕДСТВО КРИПТОГРАФИЧЕСКОЙ ЗАЩИТЫ ИНФОРМАЦИИ
«VPN-Key-TLS»

вариант исполнения 7

ИНСТРУКЦИЯ ПРОГРАММИСТА

Описание HTTP-интерфейса

111485466.72.21.12.124 33 01

2022

Содержание

1. Введение

СКЗИ «VPN-Key-TLS» в варианте исполнения 7 (далее «СКЗИ» или СКЗИ «VPN-Key-TLS») реализует криптографические функции и функции, работающие с данными, записанными в ASN.1 формате: сертификатами x.509, запросами на сертификаты, списками отозванных сертификатов и данными, помещёнными в CMS-конверты, а также построение каналов связи по протоколу TLS с ресурсами из списка (далее «бизнес-системами»), используя шифрование в соответствии с ГОСТ 28147-89.

Программное обеспечение (ПО) СКЗИ «VPN-Key-TLS» состоит из двух частей:

- программный модуль микрокода, установленный в устройство «VPN-Key-TLS» (далее «Устройство») и исполняемый непосредственно аппаратными средствами Устройства,
- программные модули, записанные на встроенный носитель Устройства, доступный для операционной системы ПЭВМ, к которой подключено Устройство, как CD-ROM и исполняемые аппаратными средствами ПЭВМ:
    - программный модуль "start.exe" для Windows,
    - "Linux_x64" для Linux,
    - "\Start.app\Contents\MacOS\Start" для MacOS.

СКЗИ «VPN-Key-TLS» предоставляет доступ с своему функционалу через HTTP-запросы на IP-адрес «localhost» (127.0.0.1).

Для этого со встроенного носителя Устройства должна быть запущена программа-сервер (например, "start.exe" для MS Windows) из состава СКЗИ, которая будет обрабатывать HTTP-запросы, формируемые прикладным ПО пользователя.

Все функции API СКЗИ доступны по адресу HTTP-сервера СКЗИ, записанному в файле "sslgate.url", расположенном на встроенном носителе Устройства. Адрес начинается после подстроки "URL=" в строке, начинающейся на неё, и заканчивается последним символом "/" из этой же строки.

Все вызовы HTTP-сервера СКЗИ не по пути «http://host:port/vpnkeylocal/*» перенаправляются им на TLS-сервер основной бизнес-системы, выбранной на устройстве специальной API-командой, а вызовы по пути «host:port/auxch/*» – на TLS-сервер резервной бизнес-системы, выбранной на устройстве другой API-командой.

Устройство может содержать до пяти независимых учётных записей, каждая из которых имеет независимый от других набор сертификатов и уникальный PIN-код доступа.

Для вызова функций СКЗИ, требующих доступа к личным объектам (сертификатам, ключевым парам), необходимо предъявление СКЗИ "Идентификатора сессии" - "SID2", путём размещения его в адресе HTTP-запросов после подстроки "vpnkeylocal/". Получить SID2 можно, вызвав у СКЗИ соответствующую функцию и передав в неё номер учётной записи, для которой следует открыть сессию, и её PIN-код. Одновременно на Устройстве может быть открыта только одна сессия.

СКЗИ имеет веб-интерфейс локального администрирования (его адрес записан в "sslgate.url" и "sslgate.html"), позволяющий с помощью браузера открыть сессию для любой из присутствующих на СКЗИ учётных записей, получить SID2, выбрать бизнес-систему для связи (из числа загруженных на Устройство), осуществить запрос на неё (по тоннелю, защищённому ГОСТ), загрузить оттуда веб-приложение ("тонкий клиент") и передать ему SID2 и управление. Далее весь необходимый функционал СКЗИ будет вызываться браузером, на страницах которого будет работать код "тонкого клиента".

Клиент может также запустить на ПК исполняемый модуль ("толстый клиент"). "Толстый клиент", самостоятельно генерируя HTTP-запросы, после ввода PIN-кода получит

у устройства SID2 и далее сможет использовать API СКЗИ из своего интерфейса.

2. Клиенты СКЗИ

2.1. Тонкий клиент

В случае «тонкого» клиента при запуске программы-сервера осуществляется открытие браузера по умолчанию и переход на страницу аутентификации пользователей (адрес которой может быть взят из секции "URL" файла "sslgate.url"). Откроется страница аутентификации пользователя для ввода PIN-кода. После успешной аутентификации устройством генерируется случайный идентификатор авторизованной сессии администратора SID2, длина которого составляет 34 символа (латиница + цифры), и происходит переход на главную страницу интерфейса управления устройством, в URL-адресе которой присутствует SID2.

С главной страницы возможен переход на страницу нужной бизнес-системы, чьи реквизиты установлены на устройство. После выбора в меню главной страницы нужной бизнес-системы устройство начинает сеанс TLS-связи с веб-сервером этой бизнес-системы и браузер осуществляет переход по переданному устройством URL вида:

http://localhost:28016/?infocrypt_sid=12345678901234567890123456789012 34&infocrypt_prefix=aHR0cDovL2xvY2FsaG9zdDoyODAxNi8=,

где присутствует идентификатор сессии SID2 для работы с API: «12345678901234567890123456789012 34». После этого все запросы браузера по адресу http://localhost:28016/*** будут транслироваться устройством по TLS-тоннелю веб-серверу по адресу "***".

Идентификатор SID2 должен использоваться в URL для формирования запросов от «тонкого» клиента к API устройства. URL в данном случае имеет вид «http://localhost:28016/vpnkeylocal/<SID2>/».

2.2. Толстый клиент

Для начала работы «толстого» клиента нужно сначала сформировать POST-запрос к API для получения списка доступных учетных записей (см. "GET_PIN_LIST"). Далее выбрать нужную учётную запись и выполнить аутентификацию путем POST-запроса к функции API (см. "LOGIN" или "LOGIN1"). Ответ на данный запрос будет содержать SID2, который следует в дальнейшем использовать для формирования POST-запросов к другим функциям API, в том числе для установления TLS-связи с бизнес-системами.

3. Вызов функций

Вызов функции осуществляется путем формирования POST-запроса, тело которого включает поля:

- идентификатор функции,
- имя и значение параметра 1,
- …
- имя и значение параметра N

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

3.1. 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 пар: «имя поля» и «значение поля».

Пример запроса, у которого значение поля с именем «ID» равно «COMMAND_ID», значение поля с именем «FIELD2» пустое, значение поля с именем «FIELD3» равно «1234»:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
ID=COMMAND_ID&Field2=&Field3=1234
```

Пример ответа сервера устройства на такой запрос:

```
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
RetField1="qwertyu"&RetField2="1234567"&RetField3="xx"&retcode="1"
```

3.2. Содержимое запросов

- Главным полем в запросе является «ID». Оно идентифицирует команду, определяющую, какую операцию устройству следует произвести.
- Порядок полей в запросе является произвольным, однако рекомендуется ставить поле "ID" на первое место. Также рекомендуется располагать поля с короткими значениями (числами, енумераторами и идентификаторами ("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, зашифрования и расшифрования). Как только устройство получит имена всех обязательных полей, оно может начать операцию до окончания прихода всех данных, что может значительно ускорить обработку запроса.
- 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 также может ускорить работу.

3.3. Кодировка запросов и ответов

- Сервер устройства поддерживает «`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.
- Сервер устройства 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.

- Поля ответов от HTTP-сервера, в которых, в соответствии с типом, отсутствует возможность использования не ASCII-символов, а также символа "&", не кодируются. Остальные поля ответов для запросов "Content-Type: text/plain" или "Content-Type: text/html" кодируются только в BASE64.

3.4. Форматы данных, используемых с HTTP-интерфейсом

Название формата	Направление (слева ПК)	Контроль средствами start.exe		Устройству передаётся	Комментарии
		алфавит	другое		
BASE64	<< >>	Для "Content-Type" != "multipart/form-data" - символы Base64 и/или web-safe-Base64	Длина декодированных данных [от и до]	Для "Content-Type" != "multipart/form-data" - результат однократного декодирования из Base64/web-safe-Base64	Поддерживаются Base64 и web-safe-Base64 одновременно.
ENUM(...)	<< >>	только цифры	варианты только из явно указанных в документации.	Число типа int32_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	Возможно, одна или несколько команд 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, 0x0d, 0x0a, "-", " "	-	-	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 PEM/Base64/web-safe-Base64	Допускаются данные в двоичном формате ASN.1, или в PEM /Base64 /web-safe-Base64, в т. ч. кодированные больше одного раза. Декодирование осуществляется до получения не PEM-символов, после чего результат используется как ASN.1

PIN_STR	>>	ASCII-символы из диапазона: [0x21, 0x7E]	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 устройства, времени и т. д.

3.5. Коды возврата

В теле ответа на POST-запрос присутствует поле «retcode=”N”», где «N» – это знаковое число в десятичной системе счисления, пригодное для распознания (atoi).

Если «N» отличается от кода «OK» ("1"), то остальные поля в теле могут отсутствовать или содержать некорректную информацию, если иного не указано явно в описании команды, в ответ на которую пришёл такой код возврата.

«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, и т. д.) или обязательный аргумент отсутствует
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 CMS
16	DATA_LEN_RANGE	Функция получила недостаточно данных, или больше, чем допустимо, или больше, чем было обещано, или неожиданный номер блока данных
21	UA_NOT_ENOUGH_STORAGE	В хранилище объектов недостаточно места

«N»	Мнемоника ответа	Пояснения
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	TMPL_FMT_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	Версии 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	Передан некорректный SID
95	FUNCTION_NOT_SUPPORTED	Нераспознанная или неподдерживаемая команда
96	SESSION_TIMEOUT	Таймаут сессии
97	OPERATION_NOT_COMPLETE	Операция (подписания; проверки подписи; шифрования) не выполнена
200	CCIDSSL_ERR_PROXYCON	Ошибка соединения с HTTP прокси
210	CCIDSSL_ERR_PROXYAUTH	Ошибка аутентификации на HTTP прокси

«N»	Мнемоника ответа	Пояснения
220	CCIDSSL_ERR_PROXYGENERIC	Ошибка работы с HTTP прокси
700	CORE_FAULT_00	Ошибка ядра #0
701	TOKEN_INIT_ERROR	Ошибка инициализации устройства
702	MALLOC_INT_ERROR	Ошибка выделения защищённой памяти на устройстве
703	MALLOC_EXT_ERROR	Ошибка выделения внешней памяти на устройстве
704	MALLOC_PC_ERROR	Ошибка выделения памяти на ПК
705	MALLOC_ERROR	Ошибка выделения памяти (наследная)
713	PC_FW_DAMAGED	Исполняемые модули для ПК на внешнем носителе повреждены
714	PC_SW_CHECKING	Происходит проверка исполняемых модулей для ПК на внешнем носителе устройства. Операцию следует повторить через несколько секунд
720	FS_WRONG_FILEID	Файл не найден
722	FS_IO_WRITE_ERROR	Ошибка записи файла на устройство
723	FS_IO_READ_ERROR	Ошибка чтения файла с устройства
724	FS_PERMISSION_ERROR	Недостаточно полномочий для файловой операции на устройстве
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 идентификатор 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 сертификат с доступным ключом в списке получателей зашифрованного документа
787	CO_RECEPIENT_NOT_SPECIFIED	Не указан сертификат, в адрес которого выполняется шифрование

«N»	Мнемоника ответа	Пояснения
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 встроенного ПО
791	UPD_WRONG_SIGNATURE	Неверный заголовок в файле встроенного ПО
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 встроенного ПО
795	UPD_OLD_ERR_BL_NO_UPDATE	Обновление не найдено
796	UPD_OLD_ERR_BL_NOT_COMPATIBLE	Обновление не совместимо с текущей программно-аппаратной 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 защиты 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-код пользователя заблокирован (можно восстановить с помощью PUK-кода)
822	UA_CHANGE_PIN_DIVERGENCE	Введенные PIN-коды не совпадают
823	UA_CHANGE_PIN_INCORRECT	Введен неверный текущий PIN-код при смене PIN
824	UA_CHANGE_PUK_DIVERGENCE	Введенные PUK-коды не совпадают
825	UA_CHANGE_PUK_INCORRECT	Введен неверный текущий PUK-код при смене PUK-кода
830	UA_TLS_CERT_WARNING	Ошибка загрузки личного сертификата TLS
831	UA_TLS_CERT_ERROR	TLS-сертификат не загружен
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

«N»	Мнемоника ответа	Пояснения
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 установлен более новый список отзыва сертификатов
856	GEC_WRONGUSAGE	Неверное использование объекта (например, сертификата TLS для ЭП и т.п.)
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 Закрытый 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
869	GEC_SHARED_ACCESS	Совместный доступ к объекту, принадлежащему другому интерфейсу
870	GEC_NO_CRL	CRL отсутствует
900	FUNCTION_NOT_IMPLEMENTED	Функция не реализована в этой версии встроенного ПО
901	MISMATCH_TOKEN_AND_PC_SW	Встроенное ПО устройства и используемое на ПК ПО несовместимы
902	NO_PRIMARY_CERTS	На устройстве отсутствует сертификат первичного TLS -подключения или транспортный
950	SC_OK_DONE	Софт-криптофункция выполнена успешно, результат не выгружен
951	CRYPTO_INVAL	Софт-криптофункция получила неверные параметры
952	SC_CRYPTO_FAIL	Софт-криптофункция завершилось неуспешно
953	GEC_INTERNAL	Внутренняя ошибка хранилища сертификатов
999	VALIDATION_ERROR	Ошибка формата одного или нескольких полей запроса

«N»	Мнемоника ответа	Пояснения
1030	CCIDSSL_CANT_OPEN_SOCKET_ON_TlsS	Не удаётся открыть сокет на TLS-сервере
1300	GEC_CHAIN	Не удалось построить цепочку доверия
-1102	CCIDSSL_ACCESSDENY	Бизнес-система не выбрана, или вход на устройство не произведён
-1103	CCIDSSL_OUTOFSTREAMS	Нет свободных TLS-потоков
-1104	CCIDSSL_WANTMOREDATA	Недостаточно данных с TLS-сервера
-1105	CCIDSSL_WANTOUTDATA	Устройству 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	Не удается построить цепочку доверия до сертификата сервера
-1111	CCIDSSL_NOOWN	Сертификат сервера отсутствует или повреждён
-1112	CCIDSSL_NOCERT	Ошибка получения сертификата сервера
-1113	CCIDSSL_BADPEM	Поврежден контейнер сертификата сервера
-1114	CCIDSSL_BADSIGN	Подпись сертификата сервера не верна
-1115	CCIDSSL_CERTEXPIRED	Срок действия сертификата сервера истёк
-1116	CCIDSSL_CERTREVOKED	Сертификат сервера находится в списке отзыва
-1117	CCIDSSL_NOTTRUSTED	Не удалось проверить подлинность сертификата сервера
-1118	CCIDSSL_UNOCERT	Сервер не получил сертификат клиента
-1119	CCIDSSL_UBADPEM	Сервер получил поврежденный контейнер сертификата клиента
-1120	CCIDSSL_UBADCERT	Сервер получил повреждённый сертификат клиента
-1121	CCIDSSL_UCERTEXPIRED	Сервер получил сертификат клиента с истекшим сроком действия, или истек срок действия CRL издателя
-1122	CCIDSSL_UCERTREVOKED	Сервер обнаружил сертификат клиента в списке отзыва
-1123	CCIDSSL_UNOTTRUSTED	Серверу не удалось проверить подлинность сертификата клиента
-1124	CCIDSSL_UNRECGNAME	Запрашиваемая прикладная система неизвестна серверу
-1125	CCIDSSL_RHSREQ	Запрос рехендшейка от сервера
-1126	CCIDSSL_BROKENKEY	Несовпадение ключей клиента и сервера
-1127	CCIDSSL_NOMOREOWNCERTS	При SwitchCert сертификаты клиента закончились

«N»	Мнемоника ответа	Пояснения
-1128	CCIDSSL_OK_WITHKEYEXPORT	Успешное завершение хендшейка для СОФТ-TLS соединения
-1129	CCIDSSL_BROKEN_SOFTKEYS	Неправильные ключи переданы устройством на ПК при установке СОФТ-TLS соединения
-1202	CCIDSSL_TUN_ALREADY_EXIST	Данный туннель уже активирован

4. Описание функций

Значения аргументов функций, которые записаны в графе «Назначение» серым цветом, обозначают устаревшие либо заложенные на будущее варианты. Поддержка таких значений в ближайших релизах не планируется.

4.1. Функции управления сессией пользователя

В этом разделе описано, как ПО клиента может получить идентификатор сессии SID своими силами, а не по ссылке, передаваемой из интерфейса администрирования, как для Тонкого клиента.

4.1.1. Получение списка доступных учетных записей

Вход:

SID	Допускается использование SID2+"/", SID0+"/", некорректный или отсутствующий SID.

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	GET_PIN_LIST	идентификатор функции	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID 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 символов	номер учетной записи (после которого обязательно следует имя этой учётной записи)
user	STRING_A	8 символов	имя учетной записи
retcode	ENUM	2 символов	код возврата функции

Пример:

```
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
pin="PIN 1"&user="1"&pin="SUSPEND_PIN 2"&user="2"&pin="PURGED_PIN 3"&user=
"3"&retcode="1"
```

Примечания:

- В случае если действие PIN-кода 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 (см. "FORMAT_PIN_ID"), и PIN-код после этого не задавался командой «CH_PIN_BY_PIN_ID» (т. е. функциональность учётной записи ограничена), в начале имени присутствует слово «PURGED_».

4.1.2. Открытие сессии с прерыванием текущей сессии

Вход:

SID	Не требуется. Допускается использование SID2+"/", SID0+"/", или отсутствие SID.

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	LOGIN	идентификатор функции	

user	NUMBER	1,..,5	номер учетной записи	
pin	PIN_STR	6 символов	PIN-код	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID 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 символа	идентификатор сессии
user	NUMBER	1,..,5	номер учётной записи, открывшей сессию
retcode	ENUM	-32768..32767	код возврата функции

Пример:

```
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
sid2="DD04M5rLhF8q0xJGAeddLmoVvm51BaYGls"&user="1"&retcode="1"
```

Примечания:

- Если при вызове команды LOGIN на устройстве уже открыта другая сессия, то она автоматически прекращается. Существует также команда LOGIN1, полностью совместимая с LOGIN по аргументам, которая в случае обнаружения на устройстве открытой сессии не прекращает её, а возвращает соответствующий код ошибки.

4.1.3. Открытие сессии с сохранением текущей сессии

Вход:

SID	Не требуется. Допускается использование SID2+"/", SID0+"/", или отсутствие SID.

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	LOGIN1	идентификатор функции	
user	NUMBER	1,..,5	номер учетной записи	
pin	PIN_STR	6 символов	PIN-код	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="LOGIN1"&user="1"&pin="123456"
```

Выход:

Параметр	Тип	Ограничение	Назначение
sid2	HANDLE	34 символа	идентификатор сессии
user	NUMBER	1,..,5	номер учётной записи, открывшей сессию
retcode	ENUM	-32768..32767	код возврата функции

Пример:

```
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
sid2="DD04M5rLhF8q0xJGAeddLmoVvm51BaYGls"&user="1"&retcode="1"
```

Примечание:

- Функция полностью совместима по входу и выходу с LOGIN, однако имеет следующее отличие: если при вызове команды на устройстве уже была открыта сессия этого же или другого пользователя, то она не прерывается, а в ответ на команду выдаётся соответствующий код ошибки.

4.1.4. Завершение сессии

Вход:

SID	Не требуется. Допускается использование SID2+"/", SID0+"/", или отсутствие SID.

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	LOGOUT	идентификатор функции	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="LOGOUT"
```

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

Пояснение:

- После выполнения функции теряет силу SID2 и идентификаторы всех объектов, которые запрашивались в течение сессии из хранилища. Закрываются все выполнявшиеся криптооперации, и до открытия следующей сессии запрещается построение TLS-тоннелей.

4.2. Управление сессиями TLS

4.2.1. Получение списка доступных бизнес-систем

Команда выдаёт только «толстые» (недоступные на главной странице веб-интерфейса устройства) бизнес-системы.

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	GET_BS_LIST	идентификатор функции	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID 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	наименование системы
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

4.2.2. Открытие сессии с основной бизнес-системой

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SET_BS_USE	идентификатор функции	
bsid	NUMBER	1 байт	идентификатор системы, полученный от команды "GET_BS_LIST"	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
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	-32768..32767	код возврата функции

Пример:

```
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
sid2="DD04M5rLhF8q0xJGAeddLmoVvm51BaYGls"&retcode="1"
```

4.2.3. Открытие сессии с резервной бизнес-системой

Для резервной бизнес-системы предусмотрен отдельный относительный URI на том же локальном хосте, на котором располагается и основная бизнес-система, и HTTP-API устройства. Резервная бизнес-система в общем случае не связана с основной (ни по сертификатам, ни по адресам TLS-серверов, с которыми будет установлена связь) и работает независимо от основной бизнес-системы.

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	START_NEW_SYSTEM	идентификатор функции	
name	STRING_A	128 символов	Имя хоста бизнес-системы, полученное от команды "GET_BS_LIST" (без названия протокола, слешей и имён подкаталогов)	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="START_NEW_SYSTEM"&name="infocrypt.ru"
```

Выход:

Параметр	Тип	Ограничение	Назначение

Пример:

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

4.2.4. Получить URL для перехода из бизнес-системы

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

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	GET_SID_URL_ID	идентификатор функции	
urlnum	ENUM	0, 1	номер SID в URL	**«0»** - ответ содержит SID0, приводит к завершению сессии (см. команду "LOGOUT") и выводит на страницу выбора учётной записи **«1»** - ответ содержит SID2, не приводит к завершению сессии и выводит на список бизнес-систем на локальной странице администрирования устройства.

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID 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	2048 символов	URL перехода
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

4.3. Администрирование учётных записей

4.3.1. Смена PIN-кода по PUK-коду

Вход:

SID	Допускается использование SID0+"/", или отсутствие SID.

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	CH_PIN_BY_PUK_ID	идентификатор функции	
user	NUMBER	1..5	номер учетной записи	
puk	PIN_STR	12 символов	данные PUK-кода	
pin	PIN_STR	6 символов	данные нового PIN-кода	
pin2	PIN_STR	6 символов	повтор нового PIN-кода	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID 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	-32768..32767	код возврата функции

Пример:

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

Примечания:

- На момент вызова функции сессия не должна быть открыта ни одной учётной записью.
- В случае трёх и более подряд вводов неверного PUK-кода включается "период принудительной задержки" (зависящий от предыдущего количества неправильных вводов PUK подряд - см. Таблицу принудительных задержек для PUK-кодов). В этом режиме при попытке ввести очередной PUK-код менее чем через "период принудительной задержки" от последнего из событий (подключения устройства, ввода неверного PUK-кода или завершения сессии любого пользователя), устройство будет возвращать ошибку "Преждевременная попытка использования кода доступа. (код 832)", и введённый PUK-код к рассмотрению приниматься не будет.
- На протяжении "периода принудительной задержки" можно пытаться передать PUK-код любое количество раз. Ответом будет та же ошибка ("Преждевременное использование кода доступа"), и введённый PUK-код (в т.ч. правильный) к рассмотрению приниматься не будет.
- В случае вызова функции по окончании "периода принудительной задержки" переданный PUK-код будет принят к рассмотрению.
- Если количество неправильных принятых к рассмотрению попыток ввода PUK-кода подряд достигнет 10, данная учётная запись будет заблокирована окончательно, и её дальнейшее восстановление будет невозможно.
- При вводе корректного PUK-кода, принятого к рассмотрению, счётчик предыдущего количества неверных попыток ввода PUK-кода подряд обнуляется.
- Учётная запись может быть заблокирована окончательно также в случае ввода неверного PUK-кода более 300 раз в произвольном порядке без форматирования данной учётной записи (см."FORMAT_PIN_ID").

Таблица принудительных задержек для PUK-кодов

Предыдущее число неверных попыток ввода кода подряд	0	1	2	3	4	5	6	7	8	9
Принудительная задержка до следующей попытки (в минутах)	0	0	0	15	30	60	120	180	1440	1440

4.3.2. Смена PIN-кода по PIN-коду

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	CH_PIN_BY_PIN_ID	идентификатор функции	

pin_old	PIN_STR	6 символов	данные старого PIN-кода	Для форматированных учётных записей (см "FORMAT_PIN_ID") игнорируется. Для остальных - обязательно.
pin_new	PIN_STR	6 символов	данные нового PIN-кода	
pin_new2	PIN_STR	6 символов	повтор нового PIN-кода	
puk	PIN_STR	12 символов	данные нового PUK-кода	Для форматированных учётных записей (см "FORMAT_PIN_ID") обязательно. Для остальных - игнорируется.

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="CH_PIN_BY_PIN_ID"&pin_old="123456"&pin_new="987AbC"&pin_new2="987AbC"
```

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

Примечания:

- При успешном выполнении операции текущий SID2 (полученный ещё при старом PIN-коде) сохраняет актуальность.
- В случае трёхкратного неверного ввода текущего PIN-кода сессия прерывается, SID2 теряет актуальность, а учётная запись меняет состояние на "заблокированная".

4.3.3. Смена PUK-кода по PUK-коду

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	CH_PUK_BY_PUK_ID	идентификатор функции	
puk_old	PIN_STR	12 символов	данные старого PUK-кода	
puk_new	PIN_STR	12 символов	данные нового PUK-кода	
puk_new2	PIN_STR	12 символов	повтор нового PUK-кода	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="CH_PUK_BY_PUK_ID"&puk_old="123456789012"&puk_new="NewPukCode12"&puk_new2="N
ewPukCode12"
```

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

Примечание:

- В случае трёх и более раз подряд ввода неверного текущего PUK-кода сессия прерывается, и далее поведение устройства аналогично вводам неверного PUK-кода при смене PIN-кода по PUK-коду.

4.3.4. Форматирование учётных записей с подтверждением отключением

Функция форматирует заданную учётную запись. А именно, удаляет из памяти устройства все ключевые пары и запросы, сгенерированные данной учётной записью, и все выданные для неё сертификаты. Для записи очищаются PIN- и PUK-коды, а открытие сессий этой учётной записью становится возможным по вводу любого PIN-кода. Однако внутри этих сессий запрещается создание ключевых пар и открытие TLS-тоннелей к бизнес-системам до задания пользователем PIN- и PUK-кодов (т. е. до успешного выполнения команды "CH_PIN_BY_PIN_ID").

Вход:

SID	Допускается использование SID0+"/", или отсутствие SID.

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	FORMAT_PIN_ID	идентификатор функции	
user	NUMBER	1,...,5	номер учетной записи для форматирования	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID 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	-32768..32767	код возврата функции

Пример:

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

Пояснения:

- Команда требует подтверждения в виде отключения устройства от USB-разъёма ПК в течение трёх секунд после подачи. При этом на устройстве часто одновременно мигают оба светодиода. Если в течение этого времени устройство будет отключено от USB, то после подключения заданная учётная запись будет отформатирована.
- Если в течение этого времени устройство не будет отключено, то команда не выполняется, а на ПК возвращается ошибка "Операция отменена пользователем". Повторная подача команды "FORMAT_PIN_ID" в этом случае будет возможна не ранее, чем через 60 секунд.

Примечание:

- На момент вызова функции сессия не должна быть открыта ни одной учётной записью.

4.3.5. Форматирование учётных записей без подтверждения отключением

Функция выполняет те же действия, что и "FORMAT_PIN_ID", с теми же результатами. Отличаются только условия, необходимые для выполнения функции.

Вход:

SID	Допускается использование SID0+"/", или отсутствие SID.

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	FORMAT_PIN_EX_ID	идентификатор функции	
user	NUMBER	1,...,5	номер учетной записи для форматирования	
ctx_handle	HANDLE	8 символов	идентификатор операции	Опционально

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="FORMAT_PIN_EX_ID"&user="1"&ctx_handle="AaBb34yz"
```

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	-32768..32767	код возврата функции
ctx_handle	HANDLE	8 символов	

Пример:

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

Пояснения:

- Функция требует 2 вызова.
  - При первом вызове поле "ctx_handle" должно отсутствовать. В ответ на него клиенту будет возвращён идентификатор операции в поле "ctx_handle".
  - Вторым вызовом клиент должен предъявить этот же идентификатор в поле "ctx_handle". Тогда операция завершится, и учётная запись будет отформатирована.
- Критерием того, что операция завершилась успешно, будет "retcode=1" и отсутствие поля "ctx_handle" в ответе на команду.
- Если в ответе присутствует поле "ctx_handle", значит в запросе был передан неверный "ctx_handle", форматирование записи не произведено, а верным теперь считается "ctx_handle", выданный только что.

4.4. Обновление встроенного ПО устройства

4.4.1. Начать установку частей обновления

Вход:

SID	Не требуется. Допускается использование SID2+"/", SID0+"/", некорректный или отсутствующий SID.

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	INIT_UPD_ID	идентификатор функции	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
```

```
id="INIT_UPD_ID"
```

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

4.4.2. Установить часть обновления

Вход:

SID	Не требуется. Допускается использование SID2+"/", SID0+"/", некорректный или отсутствующий SID.

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SET_UPD_D_ID	идентификатор функции	
blocknum	NUMBER	1,...,2^32 - 1	номер блока	Опционально; если присутствует, то проверяется.
data	BASE64	До 15360 байтов бинарных данных	данные обновления в base64	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="SET_UPD_D_ID"&blocknum="1"&data="RldVSMXPV14icPsOS5EBZBsxgOwnlRTCmBjg0YNqwc
zMdzRc4YAX7F2YdOTx/ax7HCisfw8YBolTZCUsRtUK/2KIdzaAgwnsAllGi4rWW4ihsm3UoEjMv3pZJ
2JeEIPuUgR05DwadwUBVCAAUwAAAAZJA0wAAAAAAAAAAAAAAAAAAAAkAAAgA1ZUJgJCVQMASFQFAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAtu29t9blpbO9mZ8XfAf5xqLRmlIH/uMjCRpVcsFyjl9kpvkFCT9QfR
LWh0Aoj62dUIDhPYfCh31o8Konk3MAx/xksM/AAAAAAAAAAAAf84Ygp8D4G28nrALJiK0hHvX9+c8Qe
+5UyeoTxfXCi/OTtFU8wzWPqryFtH/sdg1AuRi6X0hxM2JjEiRReNXv1PKQi7u7unHiysy9HYKo1k12
jmlgcKWWEIObKszl4YwRHJor8I3Vg1dNU/ex569cs7zEHfUu/JkYbMJ3VdZ9NY/UdPWXjCjxATbspK9
siUo0kADDorfQmEQaL5zulNpWuYiWZefOW5ZoEqp1Tqc49kpOG6PeV7eC1TUKeZskbgxhOs4nON2IM9
KrKu/q7EZ6pBNvFSaEnx0+uPlZV/VHck1KYSh4RGSvyVU3bULHm9edv88o7Gx6olT+6rFCnzPPi3JpH
ThPv16uMSUrFYLzZRMlaWwNoV4y6+dQrOdeJo49/LLMWgpJEnU7AHjspqqRidUBwl/tSZnC8oVkwXgJ
/7qiAXI3dhXnMaize+wA3CbHQjFros3YKSnJasU4mHz+Ia2CwwcJwdUekMKp4E4ZSwqeA8pJ5mHMtd8
MNs8ZavI9uJlhnj2+08VILFTddHNR3u80cgAzsYW/VBteqrL8Sa+YobnK/HfjipswZPGI3Ywl7ryTfa
fxkUwH487nopRX50d3LxkOPvqjUAprgw9BdJOkLnUMRffjQpPDt26lnte9MsGBO4S8SLm3sE+UIxuxx
VcHP0vSMRlLiLA/1mgu1YQBdq7ph0lasij"
```

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

4.4.3. Завершить установку частей обновления

Вход:

SID	Не требуется. Допускается использование SID2+"/", SID0+"/", некорректный или отсутствующий SID.

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	FIN_UPD_ID	идентификатор функции	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="FIN_UPD_ID"
```

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

4.5. Модификация хранилища сертификатов

4.5.1. Генерировать ключевую пару и квалифицированный запрос на сертификат PKCS10

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	CREATE_PAIR_EX_ID	идентификатор функции	
dn	BASE64	4096 байтов бинарных данных	Владелец сертификата (поле subject из запроса и будущего сертификата) в виде собранной ASN.1 структуры	Для req_type == 6 и 7 игнорируется
attr	BASE64	4096 байтов бинарных данных	Атрибуты расширенного запроса на сертификат (для размещения в секции "extensionRequest") в виде фрагмента ASN.1 структуры	Для req_type == 6 и 7 игнорируется
attr2	BASE64	4096 байтов бинарных данных	Прочие атрибуты запроса в виде фрагмента ASN.1 структуры	Опционально; Для req_type == 6 и 7 игнорируется
req_type	ENUM	1,...,7	3 (или наследн. 1) – личный ЭП; 4 (или наследн. 2) – личный TLS; 6 – первичный TLS; 7 – транспортный ЭП	
pk_alg	ENUM	3,4	Формат криптоопераций: 1 - RSA 2 - ГОСТ-2001 3 – ГОСТ-2012-256 4 – ГОСТ-2012-512	Для req_type == 6 и 7 игнорируется
hash_alg	ENUM	2,3	Стандарт подсчёта хеш-кода: 1 – ГОСТ-3411_1994_256 2 – ГОСТ-3411_2012_256 3 – ГОСТ-3411_2012_512	Опционально; по умолчанию: "2"

paramset	STRING	Явно указанные в списке "Поддерживаемые варианты"	Наборы параметров эллиптической кривой. Поддерживаемые варианты: "1" – SET_A "2" – SET_B "3" – SET_C "4" – SET_XchA "5" – SET_XchB "6" – SET_A_TK26_256 "7" – SET_A_TK26_512 "8" – SET_B_TK26_512 "9" – SET_C_TK26_512 "SET_A" "SET_B" "SET_C" "SET_XchA" "SET_XchB" "SET_A_TK26_256" "SET_B_TK26_256" "SET_C_TK26_256" "SET_D_TK26_256" "SET_A_TK26_512" "SET_B_TK26_512" "SET_C_TK26_512"	Опционально; по умолчанию: "SET_B"
sig_alg	ENUM	2	1 – (MD5_RSA 2 –GostR3410_GostR3411	
ow	ENUM	1, 2	1 – Если в хранилище нет места, то автоматически удалить самый старый из прошлых ключей; 2 – Если в хранилище нет места, то вернуть код ошибки	Опционально; по умолчанию: "2"

Пояснения:

- ***dn*** – имя владельца сертификата, на который формируется запрос. Сущность Name – как определяется 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 на сертификат. Этот аргумент проверяется на устройстве исключительно на валидность ASN.1 структуры.
- ***attr*** – произвольное число сущностей типа Attribute, как определено RFC5280, последовательно помещенных в передаваемый массив. Аргумент на устройстве 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 заполняется устройством автоматически)
1.2.643.100.113.X	Класс криптосредства	Должен отсутствовать (значение заполняется устройством автоматически)
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	Серийный номер устройства	Должен отсутствовать (значение заполняется устройством автоматически)
2.5.29.15	KeyUsage	Если присутствует, то должен иметь установленными биты: - для ЭП-сертификатов: digitalSignature(0) и nonRepudiation(1)
2.5.29.37	ExtendedKeyUsage	Если присутствует, то должен содержать:

OID	Назначение OID	Требование в OID
		- для ЭП-сертификатов "OID_ANY_KEY_USAGE" (2.5.29.37.0) и/или "ID_KP_EMAILPROT" (1.3.6.1.5.5.7.3.4). - для TLS-сертификатов "OID_ANY_KEY_USAGE" (2.5.29.37.0) и/или "ID_KP_CLIENTAUTH" (1.3.6.1.5.5.7.3.2). Подробнее: https://tools.ietf.org/html/rfc5280 - для транспортных сертификатов единственный OID "1.2.643.3.123.5.19"

- В дополнение к переданным атрибутам устройством могут быть добавлены дополнительные системные атрибуты.
- **attr2** – произвольное число сущностей типа Attribute как определено RFC5280, последовательно помещенных в передаваемый массив.

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id=CREATE_PAIR_EX_ID&dn=MIIBkjENMAsGA1UEAwwESFNCQzEVMBMGA1UEBAwM0KLRj9C%2F0LrQu
NC9MSowKAYDVQQqDCHQlNC20Y3RhCDQktC10L3QtdC00LjQutGC0L7QstC40YcxCzAJBgNVBAYTAlJV
MRUwEwYDVQQHDAzQntC80YHQutCw0Y8xGDAWBgNVBAgMDzU1INCe0LzRgdC60LDRjzErMCkGA1UECQw
i0J3QsNCx0LXRgNC10LbQvdCw0Y8g0YPQu9C40YbQsCwgMTENMAsGA1UECgwESFNCQzEkMCIGA1UECw
wb0J7RgtC00LXQuyDQvtC%2F0LXRgNCw0YbQuNC5MTAwLgYDVQQMDCfQodGC0LDRgNGI0LjQuSDQvtC
%2F0LXRgNCw0YbQuNC%2B0L3QuNGB0YIxGjAYBggqhQMDgQMBARIMMDA3NzEwMzUzNjA2MRgwFgYFKo
UDZAESDTEwMjc3MzkyMDc0NjIxFjAUBgUqhQNkAxILMTEyMjMzNDQ1OTUxHjAcBgkqhkiG9w0BCQEWD
29AbG9jYWxob3N0LmNvbQ%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=
```

Выход:

Параметр	Тип	Ограничение	Назначение
obj_id	HANDLE	8 символов	идентификатор созданного объекта
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

Примечание:

- При запросе создания ключевой пары транспортного сертификата или первичного подключения (req_type=6 или 7) выдаётся сразу запрос в формате PKCS#10, а не его идентификатор. При повторном вызове этой функции создаётся новая ключевая пара первичного или транспортного сертификата, а предыдущая выгруженная по аналогичному запросу удаляется из устройства.

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

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	DELETE_OBJ_ID	идентификатор функции	
obj_id	HANDLE	8 символов	идентификатор объекта	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID 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	-32768..32767	код возврата функции

Пример:

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

4.5.3. Установить сертификат

Вход:

SID	Для установки общих сертификатов SID не требуется, и допускается использование SID2+"/", SID0+"/", или отсутствие SID. Для установки личных сертификатов обязательно SID2+"/".

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SET_CERT_D_ID	идентификатор функции	
data	PEMDER	До 15360 байтов бинарных данных	данные сертификата	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="SET_CERT_D_ID"&data="MIIBtTCCAWKgAwIBAgII8PcdCNnQFKIwCgYIKoUDBwEBAwIwHDEaMB
gGA1UEAxMRQ3J5cHRvUHJvIFRlc3QgQ0EwHhcNMTgwODMwMTMyMDM5WhcNMTkwMTMwMTMyMDM5WjAdM
RswGQYDVQQDExJUZXN0Q2VydCBUSzI2RCAyNTYwXjAXBggqhQMHAQEBATALBgkqhQMHAQIBAQQDQwAE
QIJee9V+ZaAqYcbM9awbZJzJd5hyy1Ob/xV1nypZM+IStrtxN3MycZ1R0zIP9LB+6fE3Y9YlWTXQEpo
rNdf0I5WjgYcwgYQwHQYDVR0OBBYEFNNqcqIXxOM8eUEnXwH1Ke/oSsJiMAsGA1UdDwQEAwIB/jAJBg
NVHRMEAjAAMEsGA1UdIwREMEKAFG1w7qJtu7T3W7IHpHJCmw0El4N7oSCkHjAcMRowGAYDVQQDExFDc
nlwdG9Qcm8gVGVzdCBDQYIIt/3EWTeRnoIwCgYIKoUDBwEBAwIDQQC0cTexZS6mIHJukqG9pOxVsvaS
6SOvmRwrBggh83JcIylCCh1WZRbdsUo6MDPvSWZN2oRVtVaFVfHLs3RnVdb/"
```

Выход:

Параметр	Тип	Ограничение	Назначение
obj_id	HANDLE	8 символов	идентификатор установленного объекта
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

4.5.4. Установить CRL

Вход:

SID	Не требуется. Допускается использование SID2+"/", SID0+"/", некорректный или отсутствующий SID.

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SET_CRL_D_ID	идентификатор функции	
data	PEMDER	До 15360 байтов бинарных данных	данные CRL в формате PEM/DER/Base64	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
```

```
id="SET_CRL_D_ID"&data="-----BEGIN CERTIFICATE REVOKATION LIST----
-%0D%0AMIICJzCCAdYCAQEwCAYGKoUDAgIDMIIBgjELMAkGA1UEBhMCUlUxMDAuBgNVBAoMJ9CU0L7Q
vyDQ%0D%0AuiDQvdCw0LjQvNC10L3QvtCy0LDQvdC40Y4g0KbQkDE4MDYGA1UECwwv0JTQtdC/0LDRg
NGC0LDQ%0D%0AvNC10L3RgiDQsdC10LfQvtC/0LDRgdC90L7RgdGC0LgxMDAuBgNVBAMMJ9CU0L7Qvy
DQuiDQvdCw%0D%0A0LjQvNC10L3QvtCy0LDQvdC40Y4g0KbQkDEsMCoGA1UEDAwj0JPQu9Cw0LLQvdG
L0Lkg0LfQsNCy%0D%0A0LXRgNC40YLQtdC70YwxFzAVBgNVBAQMDtCj0YbQtdGI0L3QtdCyMSowKAYD
VQQqDCHQktC70LDQ%0D%0AtNC40LzQuNGAINCf0LXRgtGA0L7QstC40YcxFjAUBgNVBAcMDdCc0L7Rg
dC60LLQsC4xIzAhBgNV%0D%0ABAkMGtGD0LsuINCS0LDQstC40LvQvtCy0LAsIDE5MRgwFgYIKoUDA4
EDAQETCjc3MDcwODM4OTMx%0D%0ACzAJBgUqhQNkAxMAFw0xMjA3MTkwODU2NTZaFw0xMjA4MTgwODU
2NTZaoCMwITAfBgNVHSMEGDAW%0D%0AgBQfaMnK+cbpT3HNC54R9/n70bppmTAIBgYqhQMCAgMDQQBE
aTmeqoK2hw3VGbQWMnI9oQzet/E7%0D%0A3+xoBpk6QBXkwg2F/6SYGgnCgqWX4Wo4mIwNW4IatWrCr
HqjfzYGEHJE%0D%0A-----END CERTIFICATE REVOKATION LIST-----"
```

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

4.6. Использование хранилища сертификатов

4.6.1. Получить список объектов (сертификаты и ключевые пары/запросы на сертификаты PKCS10)

Вход:

SID	Для запроса списка общих объектов SID не требуется, допускается использование SID2+"/", SID0+"/", некорректный или отсутствующий SID. Для запроса личных объектов обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	GET_OBJ_LIST_ID	идентификатор функции	
obj_type	ENUM	0,...,7	0 – сертификаты ЭП; 1 – сертификаты TLS; 2 – корневые сертификаты; 3 – запросы ЭП; 4 – запросы TLS; 5 – сертификаты УЦ; 6 – первичные TLS; 7 – транспортные ЭП	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID 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	0..4096 символов	список идентификаторов, сепаратор ";"
retcode	ENUM	-32768..32767	код возврата функции

Пример:

```
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
data="abcdefga;QWER34ui;ZAq1XSw2"&retcode="1"
```

4.6.2. Найти объект по полям

Вход:

SID	Для поиска общих сертификатов SID не требуется, допускается использование SID2+"/", SID0+"/", некорректный или отсутствующий SID. Для запроса личных объектов обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	GET_CERT_ID	идентификатор функции	
serial	BASE64	0..4096 байтов бинарных данных	серийный номер в base64	Опционально
c	STRING	0..4096 символов	поле УЦ [Country]	Опционально
st	STRING	0..4096 символов	поле УЦ [STate]	Опционально
l	STRING	0..4096 символов	поле УЦ [Location]	Опционально
org	STRING	0..4096 символов	поле УЦ [ORGanization]	Опционально
ou	STRING	0..4096 символов	поле УЦ [OrgUnit]	Опционально
cn	STRING	0..4096 символов	поле УЦ [CommonName]	Опционально
ea	STRING	0..4096 символов	поле УЦ [E-mAil]	Опционально
openkey	BASE64	0..4096 байтов бинарных данных	открытый ключ в base64	Опционально
g	STRING	0..4096 символов	поле УЦ []	Опционально
inn	STRING	0..4096 символов	поле УЦ [ИНН]	Опционально
ogrn	STRING	0..4096 символов	поле УЦ [ОГРН]	Опционально
snils	STRING	0..4096 символов	поле УЦ [СНИЛС]	Опционально
t	STRING	0..4096 символов	поле УЦ [должность]	Опционально
obj_type	NUMBER	0,...,2^32-1	Типы объектов, среди которых искать (возможны комбинации по схеме «двоичное или») 16 – корневые сертификаты 32 – сертификаты УЦ 64 – сертификаты ЭП 128 – сертификаты TLS 512 – запросы ЭП 1024 – запросы TLS	Опционально; при отсутствии или значении меньше шестнадцати ищет среди всех типов по возрастанию

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="GET_CERT_ID"&serial="MDBDQTEyMzQ1Ng%3D%3D"&st="Moscow"&org="SilverSparks"
```

Выход:

Параметр	Тип	Ограничение	Назначение
data	HANDLE	8 символов	идентификатор объекта
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

Примечания:

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

- По историческим причинам, не все заявленные в этой таблице поля участвуют в поиске. Для совместимости с существующими решениями некоторые поля при поиске игнорируются.
- Несмотря на либеральные ограничения по длине для текстовых и бинарных полей по отдельности, суммарная длина запроса не должна превышать 15360 байтов.

4.6.3. Получить сертификат X.509 или запрос на сертификат PKCS10

Вход:

SID	Для запроса общих сертификатов SID не требуется, допускается использование SID2+"/", SID0+"/", некорректный или отсутствующий SID. Для запроса личных объектов обязательно SID2+"/"

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

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID 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	данные запроса или сертификата в формате PEM
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

4.7. Подписание данных в формате CMS

Для получения ЭП в формате CMS необходимо выполнить следующие действия:

1. Проинициализировать контекст подписания (INIT_SIGN_H_ID). Устройство вернёт идентификатор криптооперации – последовательность из 8 букв и цифр. Её нужно будет использовать при вызове каждой очередной функции для данной криптооперации.
2. Внести данные для подписи ("SET_SIGN_DATA_H_ID" – 1 или более вызовов).
3. Рассчитать подпись ("CALC_SIGN_H_ID").
4. Проверить статус устройства в контексте ЭП ("GET_CTX_INFO_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 (при использовании устройств с экраном) статус принимает значение REJECTED.
   - При положительном решении (или использовании устройства без кнопки) – статус принимает значение COMPLETE.
   - 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 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 идентификатор операции станет некорректным).
6. При необходимости, ПО клиента может прервать подписание на любом этапе операции. Для этого следует использовать функцию "BREAK_CRYPTOOP_H_ID" и идентификатор данной криптооперации.
7. Одновременно ПО клиента может запустить несколько операций подписания. Устройство вернёт для них независимые идентификаторы.

4.7.1. Инициализация ЭП

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	INIT_SIGN_H_ID	идентификатор функции	
datasize	NUMBER64	0..2^63 - 1	размер данных для ЭП	
hascert	NUMBER	-1..255	максимальное количество сертификатов из цепочки доверия, которое следует вложить в CMS	"-1" – вложить все сертификаты, исключая корневой
hasdata	ENUM	0, 1	"0" – не вкладывать в CMS подписанные данные; "1" – вложить в CMS подписанные данные	
cades	ENUM	0, 1, 2	Формировать ли CAdES в подписываемых атрибутах "0" – не формировать "1" – формировать только для сертификата-подписанта "2" – формировать для всей цепочки, включая корневой	Опционально, по умолчанию: "0"
obj_id	HANDLE	8 байтов	идентификатор сертификата для подписи	Опционально, рекомендуется к использованию
name	BASE64	128 символов UTF8 (размер BASE64 для их кодировки не регламентирован)	Для экранных устройств. Наименование подписываемого документа для отображения на экран	Опционально
signed_attrs	ENUM	0, 1	"0" – не создавать в CMS секцию "signed_attrs" "1" – создавать в CMS секцию "signed_attrs"	Опционально, по умолчанию: "1" Если используется значение "0", должно разрабатываться СКЗИ с проведением тематических исследований

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID 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="aabbccdd"
```

Выход:

Параметр	Тип	Ограничение	Назначение
ctx_handle	HANDLE	8 байтов	идентификатор контекста операции
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

Примечание:

- Для работы по ГОСТ 2012 и выше необходимо указать идентификатор сертификата на этапе вызова команды "INIT_SIGN_H_ID", т. к. параметры криптооперации берутся из сертификата. Если сертификат не будет указан при вызове "INIT_SIGN_H_ID", то это будет воспринято как работа наследного ПО, и сертификат будет необходимо указать в конце. Однако в этом случае подсчёт хеш-кода на данные будет произведён по ГОСТ-3411_1994_256, и, если указанный в конце сертификат содержит другие параметры, то операция будет завершена с ошибкой.

4.7.2. Загрузить порцию данных для ЭП

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SET_SIGN_DATA_H_ID	идентификатор функции	
ctx_handle	HANDLE	8 байтов	идентификатор контекста операции	
blocknum	NUMBER	1..2^32 - 1	номер посылки	Опционально, если присутствует, то проверяется.
data	BASE64	0..16777216 байтов бинарных данных.	данные для подписи в base64	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="SET_SIGN_DATA_H_ID"&ctx_handle="zxcvbnma"&blocknum="8"&data="77u/0JTQvtC/0Y
PRgdGC0LjQvCwg0YLRg9GCINCy0L7RgdC10LzRjNC00LXRgdGP0YIg0LTQtdCy0Y/RgtGMINCx0LDQu
dGC0L7QsiDQtNCw0L3QvdGL0YU=
```

Выход:

Параметр	Тип	Ограничение	Назначение
data_length	NUMBER	0..2^63 - 1	суммарное количество данных, переданных для подписи (включая переданное данным вызовом)
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

4.7.3. Вычисление ЭП

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	CALC_SIGN_H_ID	идентификатор функции	
ctx_handle	HANDLE	8 байт	идентификатор контекста операции	
obj_id	HANDLE	8 байт	идентификатор сертификата для подписи	Наследное, не рекомендуется к использованию (см. Примечание к "INIT_SIGN_H_ID")
direct_diggest	BASE64	1024 байта	явно указанный хеш подписываемых данных	Опционально. Если присутствует, должно разрабатываться СКЗИ с проведением тематических исследований

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="CALC_SIGN_H_ID"&ctx_handle="zxcvbnma"
```

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

Примечания:

- Использование поля "obj_id" может быть необходимым для ПО, работающего также с версиями встроенного ПО устройств версий ниже 500.0. В этом случае универсальным решением будет подача идентификатора сертификата подписанта и при вызове "INIT_SIGN_H_ID", и при вызове "CALC_SIGN_H_ID".
- При использовании аргумента "direct_diggest" подписываемые данные загружать (функцией "SET_SIGN_DATA_H_ID") не обязательно. Если же процессе данной криптооперации вызывалась "SET_SIGN_DATA_H_ID", и передан "direct_diggest", то хеш-код на данные, переданных в "SET_SIGN_DATA_H_ID", должен совпадать со значением, переданным в "direct_diggest".

4.7.4. Получить ЭП в формате CMS

Вход:

SID	Обязательно SID2+"/"

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

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
```

```
Content-Length: xxx\r\n\r\n
id="GET_SIGN_CMS_H_ID"&ctx_handle="zxcvbnma"
```

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	-32768..32767	код возврата функции

Пример:

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

Пояснения:

- Если при инициализации подписания было указано, что выходной CMS должен содержать подписанные данные, то клиент должен сам сформировать выходной CMS-файл. Сначала записать декодированные данные возвращённого этой функцией поля "head", потом сами подписанные данные, потом декодированные данные возвращённого этой функцией поля "suffix".
- Если при инициализации не предполагалось включать подписанные данные в CMS-файл, то следует просто записать сначала декодированные данные поля "head", потом декодированные данные поля "suffix".

Примечания:

- 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" устройство вернёт ошибку, операция подписания будет прервана, и дальнейшее использование её идентификатора будет невозможным.

4.7.5. Информация о контексте

Вход:

SID	Обязательно SID2+"/"

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

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="GET_CTX_INFO_H_ID"&ctx_handle="zxcvbnma"
```

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 рассчитанных за время данной сессии ЭП
retcode	ENUM	-32768..32767	код возврата функции

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"
```

4.7.6. Подписание пакета документов

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SIGN_PACKET	идентификатор функции	
obj_id	HANDLE	8 символов	идентификатор сертификата для подписания	
docs_count	NUMBER	1..127	Количество документов в пакете	
doc1...doc127	BASE64	по суммарной длине 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=&docs_count="10"&doc1="77u/QVRUUklCVV..
."&doc2="..."&doc...
```

Выход:

Параметр	Тип	Ограничение	Назначение
count	NUMBER	не регламентировано	Количество CMS с подписями
cms1...cms127	BASE64	не регламентировано	Последовательность полей «cms...=<...>», содержащих полные подписи загруженных для визуализации документов с теми же номерами, с которыми они были загружены.
retcode	ENUM	-32768..32767	код возврата функции

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"
```

Пояснения:

- Блок документов должен иметь следующий формат: «**doc<номер документа>=<base64-закодированный документ>&doc<номер следующего документа>=<base64-закодированный следующий документ>** и т.д...». Документы в блоке должны быть отсортированы по порядку и не перемежаться с другими параметрами.
- Номера, переданные в именах входных параметров «data...», будут присутствовать 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».
- При формировании CMS-подписей параметры подписания (аналогично "INIT_SIGN_H_ID") соответствуют: hascert=-1&hasdata=0&signed_attrs=1;

4.8. Проверка ЭП в CMS

4.8.1. Инициализация проверки ЭП

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	INIT_CHECK_H_ID	идентификатор функции	
cms_data	PEMDER	15360 байтов бинарных данных	данные CMS ЭП для проверки	Опционально; рекомендуется к использованию

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="INIT_CHECK_H_ID"&cms_data="MIIBDwYJKoZIhvcNAQcCoIIBADCB/QIBATEOMAwGCCqFAwcB
AQICBQAwCwYJKoZIhvcNAQcBMYHaMIHXAgEBMHQwXTEcMBoGCSqGSIb3DQEJARYNYmxhY2tAamFjay5
ydTELMAkGA1UEBhMCUlUxDzANBgNVBAcTBk1vc2NvdzEMMAoGA1UEChMDUG90MREwDwYDVQQDEwhKYW
NrIFBvdAITEgA7O2fntFJUzkMypAABADs7ZzAMBggqhQMHAQECAgUAMAwGCCqFAwcBAQEBBQAEQHbtW
QsvJwNnLwnyeVZTeh0rFxp/rVAkvhcta9lt/jcxxOPJ51fdNqj+cTQSJR7I4WJUkJB0FXDGcJ/jCQNU
2gw="
```

Выход:

Параметр	Тип	Ограничение	Назначение
ctx_handle	HANDLE	8 байт	идентификатор контекста операции
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

Примечание:

- Для работы по ГОСТ 2012 и выше необходимо указать CMS с подписью на этапе вызова команды "INIT_CHECK_H_ID", т.к. параметры криптооперации берутся из сертификата подписанта (а сертификат подписанта определяется по данным из файла подписи). Если сертификат не будет известен при вызове "INIT_CHECK_H_ID", то это будет воспринято как работа наследного ПО, и CMS с подписью будет ожидаться после загрузки данных. Однако в этом случае подсчёт хеш-кода на данные будет произведён по ГОСТ-3411_1994_256. Если указанный в конце сертификат содержит другие параметры, то операция будет завершена с ошибкой.

4.8.2. Данные для проверки ЭП

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SET_CHECK_DATA_H_ID	идентификатор функции	
ctx_handle	HANDLE	8 символов	идентификатор контекста операции	
blocknum	NUMBER	1..2^32 - 1	номер посылки	Опционально, если присутствует, то проверяется.

data	BASE64	0..16777216 байтов бинарных данных.	данные для проверки подписи в base64	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="SET_CHECK_DATA_H_ID"&ctx_handle="zxcvbnma"&blocknum="8"&data="77u/0JTQvtC/0
YPRgdGC0LjQvCwg0YLRg9GCINCy0L7RgdC10LzRjNC00LXRgdGP0YIg0LTQtdCy0Y/RgtGMINCx0LDQ
udGC0L7QsiDQtNCw0L3QvdGL0YU="
```

Выход:

Параметр	Тип	Ограничение	Назначение
data_length	NUMBER64	0..2^63 - 1	суммарное количество данных, переданных для проверки подписи (включает то, что передано данным вызовом)
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

Примечания:

- Если загруженный при инициализации CMS содержал вложенные подписанные данные, то "SET_CHECK_DATA_H_ID" можно не вызывать. Однако, если вызов был произведён, то необходимо передать те же данные, что были вложены в загруженный при инициализации CMS.

4.8.3. Проверка ЭП

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	CHECK_SIGN_H_ID	идентификатор функции	
ctx_handle	HANDLE	8 байт	идентификатор контекста операции	
cms_data	PEMDER	15360 байтов бинарных данных	данные CMS ЭП для проверки	НАСЛЕДНОЕ! Не рекомендуется к использованию (см. Примечание к "INIT_CHECK_H_ID")
direct_diggest	BASE64	1024 байта	явно указанный хеш-код подписанных данных	Опционально. Если присутствует, должно разрабатываться СКЗИ с проведением тематических исследований

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="CHECK_SIGN_H_ID"&ctx_handle="zxcvbnma"
```

Выход:

Параметр	Тип	Ограничение	Назначение
data	ENUM	0, 1	атавизм
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

Примечание:

- При использовании аргумента "direct_diggest" проверяемые данные загружать (функцией "SET_CHECK_DATA_H_ID") не обязательно. Если же процессе данной криптооперации вызывалась "SET_CHECK_DATA_H_ID", и передан "direct_diggest", то хеш-код на данные, переданные в "SET_CHECK_DATA_H_ID", должен совпадать со значением, переданным в "direct_diggest".

Пояснение:

- Код возврата "1" в ответ именно на эту функцию означает верность подписи.

4.9. Шифрование данных в CMS

Для шифрования в формате CMS необходимо выполнить следующие действия:

1. Проинициализировать контекст шифрования ("INIT_ENCIPHER_ID").
2. При необходимости, внести данные сторонних сертификатов-получателей ("ADD_RECIPIENT_Y_ID" – один или более вызовов), получить блок информации для каждого получателя "Recipient_info".
3. При необходимости, внести данные внутренних сертификатов-получателей ("ADD_RECIPIENT_X_ID" – один или более вызовов), получить блок информации для каждого получателя "Recipient_info".
4. Внести данные для шифрования ("ENCIPHER_ID" – один или более вызовов), получить зашифрованные данные.
5. Получить данные начала и окончания CMS ("GET_ENCIPHER_CMS_ID") - "head", "suffix" и "postfix".
6. Сформировать CMS с использованием полученных данных в соответствии со схемой: Зашифрованные_данные_в_формате_CMS = PEM_head + base64 (head + Recipient_info + suffix + зашифрованные данные + postfix) + PEM_tail

4.9.1. Инициализация шифрования

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	INIT_ENCIPHER_ID	идентификатор функции	
datasize	NUMBER64	0..2^63 - 1	размер данных для шифрования	0 - режим неопределённой длины ("infber")

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
```

```
id="INIT_ENCIPHER_ID"&datasize="204896"
```

Выход:

Параметр	Тип	Ограничение	Назначение
ctx_handle	HANDLE	8 символов	идентификатор контекста операции
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

Пояснение:

- При инициализации в режиме неопределённой длины ("infber") шифрование заканчивается вызовом "GET_ENCIPHER_CMS_ID". Если длина была задана явно, то вызов "GET_ENCIPHER_CMS_ID" должен осуществляться только после подачи всей обещанной к зашифрованию длины.

4.9.2. Добавление стороннего сертификата получателя

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	ADD_RECIPIENT_Y_ID	идентификатор функции	
ctx_handle	HANDLE	8 байт	идентификатор контекста операции	
data	PEMDER	15360 байтов бинарных данных	данные стороннего сертификата получателя	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="ADD_RECIPIENT_Y_ID"&ctx_handle="zxcvbnma"&data="<cert_data>"
```

Выход:

Параметр	Тип	Ограничение	Назначение
data	BASE64	2048 байтов бинарных данных	часть данных для формирования CMS содержит информацию о получателе
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

4.9.3. Добавление собственного сертификата получателя

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	ADD_RECIPIENT_X_ID	идентификатор функции	

ctx_handle	HANDLE	8 байт	идентификатор контекста операции	
obj_id	HANDLE	8 байт	идентификатор собственного сертификата	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="ADD_RECIPIENT_X_ID"&ctx_handle="zxcvbnma"&obj_id="abcdefga"
```

Выход:

Параметр	Тип	Ограничение	Назначение
data	BASE64	2048 байтов бинарных данных	часть данных для формирования CMS содержит информацию о получателе
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

4.9.4. Шифрование данных

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	ENCIPHER_ID	идентификатор функции	
ctx_handle	HANDLE	8 байт	идентификатор контекста операции	
data	BASE64	0..16777216 байтов бинарных данных	данные для шифрования;	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="ENCIPHER_ID"&ctx_handle="zxcvbnma"&data="<data_to_encipher>"
```

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	-32768..32767	код возврата функции
enciphered_blob	BASE64	0..16793600 байтов бинарных данных	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>"
```

4.9.5. Получить части для CMS

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	GET_ENCIPHER_CMS_ID	идентификатор функции	

ctx_handle	HANDLE	8 байт	идентификатор контекста операции	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="GET_ENCIPHER_CMS_ID"&ctx_handle="zxcvbnma"
```

Выход:

Параметр	Тип	Ограничение	Назначение
head	BASE64	1..2048 байтов декодированных данных	данные начала CMS в base64
suffix	BASE64	1..2048 байтов декодированных данных	данные суффикса CMS в base64
postfix	BASE64	1..2048 байтов декодированных данных	данные постфикса CMS в base64 (Поле может отсутствовать в ответе. Если присутствует, то следует использовать)
retcode	ENUM	-32768..32767	код возврата функции

Пример:

```
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
head="<head_data>"&suffix="<suffix_data>"&postfix="<postfix_data>"&retco
de="1"
```

4.10. Расшифрование данных в CMS

Для расшифрования в формате CMS необходимо выполнить следующие действия:

1. Проинициализировать контекст расшифрования (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
3. После окончания передачи всех имеющихся данных CMS получить подтверждение успешного расшифрования с помощью вызова "END_DECIPHER_ID". Код возврата "1" будет таким подтверждением. Код, отличный от "1", будет свидетельствовать о повреждении исходного CMS-файла.

4.10.1. Инициализация расшифрования

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	INIT_DECIPHER_ID	идентификатор функции	
head	BASE64	1..8192 байта бинарных данных	данные начала CMS в base64	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="INIT_DECIPHER_ID"&head="<cms_head_containing_blob>"
```

Выход:

Параметр	Тип	Ограничение	Назначение
ctx_handle	HANDLE	8 байт	идентификатор контекста операции
body_displ	NUMBER	2^32 - 1	размер (смещение) начала CMS в чистом виде (до кодирования base64)
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

Пояснение:

- Далее при вызове "DECIPHER_ID" нужно подавать зашифрованные данные, взятые из исходного CMS, начиная с отступа, равного (для двоичных данных) полученному от этой функции "body_displ".

4.10.2. Расшифровывание данных

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	DECIPHER_ID	идентификатор функции	
ctx_handle	HANDLE	8 байт	идентификатор контекста операции	
data	BASE64	0..16777216 байтов бинарных данных	данные для расшифрования	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="DECIPHER_ID"&ctx_handle="zxcvbnma"&data="<data_to_decipher>"
```

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	-32768..32767	код возврата функции
deciphered_blob	BASE64	не превышает размер поданного на вход поля "data"	расшифрованные данные

Пример:

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

4.10.3. Завершение расшифровывания данных

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	END_DECIPHER_ID	идентификатор функции	
ctx_handle	HANDLE	8 байт	идентификатор контекста операции	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
```

```
Content-Length: xxx\r\n\r\n
id="END_DECIPHER_ID"&ctx_handle="zxcvbnma"
```

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

4.11. Вспомогательные функции

4.11.1. Информация о системе

Вход:

SID	Не требуется. Допускается использование SID2+"/", SID0+"/", некорректный или отсутствующий SID.

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	GET_SYS_INFO_ID	идентификатор функции	
recalc_hash	ENUM	0,1	"1" - Пересчитать значения контрольных сумм встроенного ПО и записанных на диск устройства приложений для ПК	Опционально; по умолчанию "0"

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID 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 символов	информация о конфигурации
rnginitdate	STRING_A	19 символов	HH:MM:SS dd.mm.YYYY (Необязательно)
serial	STRING_A	12 символов	серийный номер
buildid	NUMBER	0,...,65535	номер сборки
curuser	NUMBER	-1,1,...,5	текущая учетная запись от 1 до 5; если сессия не открыта, или недоступна по переданному SID, то "-1". Наследное! Вместо этого поля рекомендуется использовать поля "curuser_ex" и "session".
curuser_ex	NUMBER	-1,1,...,5	текущая учетная запись от 1 до 5; если сессия не открыта, или завершилась, то "-1"
session	NUMBER	-32768..32767	Код ошибки, который выдало бы устройство при попытке использовать переданный SID для вызова любой функции, требующей SID2 (может использоваться для определения валидности SID и состояния сессии)
fw_hash_st	STRING_A	64 символа	записанное на устройство контрольное значение встроенного ПО устройства
sw_hash_st	STRING_A	64 символа	записанное на устройство контрольное значение записанных на диск устройства приложений для ПК

fw_hash_calc	STRING_A	64 символа	пересчитанное контрольное значение встроенного ПО устройства (если было заказано аргументом "recalc_hash")
sw_hash_calc	STRING_A	64 символа	пересчитанное контрольное значение записанных на диск устройства приложений для ПК (если было заказано аргументом "recalc_hash")
pin_must_be_changed	ENUM	0, 1	"1" - Пользователю следует изменить PIN-код, иначе будет недоступна часть функционала (эта учётная запись отформатирована - см."FORMAT_PIN_ID") - Опционально.
retcode	ENUM	-32768..32767	код возврата функции

Пример:

```
HTTP/1.0 200 OK\r\n
Content-Length: xxx\r\n\r\n
devcfg="IC0_T32S0000L_C1_VT550HT5BU3"&rnginitdate="10:26:57
16.01.2020"&serial="TST00000037E"&buildid="258"&curuser="-
1"&curuser_ex="1"&session="90"&fw_hash_st="EFE41DE67E158758C521972E3D72293D9721
CD7530C2A914A894DE9703259563"&sw_hash_st="B7DD6BC10AB800153EB083D3D156219A8D635
591DECE9CE83280C72D2F439DB3"&retcode="1"
```

Пояснения:

Строка «devcfg» строится согласно следующему шаблону:

Например: **`devcfg="IC0_A16P0000L_C1_VT550BU03NT05"`**

Состав:

- Идентификатор производителя ("**`IC`**" -- "ИнфоКрипт"),
- Один или несколько символов, не несущие полезной нагрузки для программы-клиента, заканчивающиеся символом "_" (в примере: "**`0_`**"),
- Сведения об аппаратной части устройства, не несущие полезной нагрузки для программы-клиента, заканчивающиеся символом "_" (в примере: "**`A160000L_`**"),
- Ещё один или несколько символов, не несущие полезной нагрузки для программы-клиента, заканчивающиеся символом "_" (в примере: "**`C1_`**"),
- VT<Версия встроенного ПО>. Версия встроенного ПО представляет собой число, состоящее из десятичных цифр без знака в количестве от одной до пяти.
- Далее строка конфигурации может заканчиваться, либо может следовать информация о модулях устройства - дополнительных программно-аппаратных функциях (свойствах) данного устройства, информирующая прикладную программу о возможностях или ограничениях устройства).
    - Каждый блок информации о модуле представляет из себя имя модуля, состоящее из двух латинских букв (с учётом регистра) и от одной до пяти цифр версии (возможно, в этом количестве присутствуют ведущие нули). Цифры заканчиваются окончанием строки конфигурации, либо первой буквой имени очередного модуля.
    - Имена модулей не сортированы.
    - Порядок следования информации о модулях не имеет значения.
    - От версии к версии встроенного ПО не гарантируется сохранение порядка следования информации о модулях одного и того же устройства.
    - На момент написания этой документации в природе существовали устройства, имевшие следующие модули в различных комбинациях:
        - BU – кнопка
        - HT – гипертехнологии (совместимость с криптопровайдерами и с другими ОС, кроме Windows). *Наличие этого модуля включает в себя весь функционал, соответствующий модулю NT версии 05.
        - KA – поддержка кодов аутентификации

- NT – новые технологии (софт-криптофункции, длина HTTP-запросов больше 16 Кб)
- SC – экран

4.11.2. Установить сертификат по умолчанию

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SET_ACT_OBJ_ID	идентификатор функции	
obj_id	HANDLE	8 символов	идентификатор сертификата	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID 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	-32768..32767	код возврата функции

Пример:

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

Пояснения:

- Если id соответствует TLS-сертификату, то этот сертификат будет в первую очередь предъявляться при установке связи с TLS-сервером.
- Если это сертификат ЭП, то данные этого сертификата будут показаны на странице администрирования в правом верхнем углу.
    - В случае последующего удаления сертификата по умолчанию, в качестве текущего используемого будет взят другой личный сертификат, время начала действия которого самое позднее среди всех установленных сертификатов данного типа.
    - При отсутствии личных сертификатов ЭП на странице администрирования будет показана строка: «Личный сертификат ЭП не установлен».
    - Для того чтобы отключить показ текста про личный сертификат ЭП в углу страниц администрирования, нужно подать в качестве "obj_id" строку: "DONT_SHOW_EP_CERT". Чтобы включить показ данных сертификата ЭП, нужно подать в качестве "obj_id" строку: "SHOW_EP_CERT".
- Также поддерживается наследная функциональность: id сертификата по умолчанию располагается на первом месте в списке, выдаваемом функцией "GET_OBJ_LIST_ID". Однако использовать эту функциональность не рекомендуется во избежание конфликтов за первое место в списке между программами, устанавливающими разные личные сертификаты на первое место в списке. Для нахождения нужного сертификата ПО клиента следует выгрузить все сертификаты по идентификаторам, выданным функцией "GET_OBJ_LIST_ID", распарсить их и найти среди них нужный сертификат.

4.11.3. Установка конфигурации бизнес-систем

Вход:

SID	Допускается использование SID2+"/", SID0+"/", или отсутствие SID.

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SET_MC_D_ID	идентификатор функции	
data	PEMDER	15360 байтов бинарных данных	данные конфигурации в CMS-пакете, подписанном сертификатом УЦ, до которого устройство может выстроить цепочку доверия	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="SET_MC_D_ID"&data="MIIB4wYJKoZIhvcNAQcCoIIB1DCCAdACAQExDDAKBgYqhQMCAgkFADCB
2QYJKoZIhvcNAQcBoIHLBIHISUNNU0MNCtLu7erg/yDB0SAxO2h0dHA6Ly9CaXpuZXMtc3lzdGVtLTE
vc3ViZGlyMS9zdWJkaXIyLzsxMS4xMS4xMS4yNTI7MQ0K0e7i8eXsIPLu7erg/yDB0SAyLy/x8vDu6u
Ag8SDq7uzs5e3y4PDo5ew7YnMyO2JzMi5tb3MucnU6NTU1OzENCtLu6/Hy4P8gwdE7aHR0cDovLzEwL
jIxLjEzMi4xNTgvcHJveHkvO0ZTQksuU0JFUkJBTksuUlU7Mg0KDQoxgeAwgd0CAQEwXDBQMSQwIgYD
VQQDDBvQn9C10YfQsNGC0Ywg0KPQpiDQodCRINCg0KQxKDAmBgNVBCEMH9Ci0LXRhdC90LjRh9C10YH
QutC40Lkg0LrQu9GO0YcCCDEyMTIxMjEzMC4GCCqFAwcBAQICMSIEIHV1MT/dVZvY5KtSCiE30lyQme
JgvK5w5ihTXxxZrbLeMAgGBiqFAwICEwRAUEpS2gAfzArWdq53K+UgXx3yGha/wJiCI8SVVUc8Y71Ri
UKo+NSaO40/DdbUic9uAlPjwqm7wCDWqAq5RFXkxQ=="
```

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

Пояснения:

- Количество бизнес-систем в загружаемом CMS-пакете не регламентировано
- Блок с описанием бизнес-систем должен:
    - Представлять собой 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. Для каждой бизнес-системы должны быть записаны следующие параметры.
        - Название бизнес-системы, как его будет видеть пользователь. (Если название содержит последовательность символов "//", то она выводиться пользователю не будет, а всё находящееся после неё и до ";" будет считаться комментарием к имени бизнес-системы и отображаться на странице администрирования устройства соответствующим стилем);
        - Название веб-ресурса, используемое расширением Server Name Indication протокола TLS, транслируемое TLS-сервером.
            - Если это поле содержит символы "://", то всё, что записано до них включительно, считается названием протокола и игнорируется.
            - Если это поле содержит (после игнорирования названия протокола) символ "/", то всё, что записано после него, будет считаться именем поддиректории на сервере бизнес-системы. Имя поддиректории не будут передано на сервер TLS, но будет добавлено в начало всех относительных URI, записанных в заголовках HTTP-запросов, передаваемых на сервер бизнес-системы.

- *Также в заголовках всех HTTP-запросов, передаваемых на сервер бизнес-системы, поле "host" будет заменено с локального адреса на название веб-ресурса. Соответственно, в заголовках ответов название веб-ресурса, будет заменено на локальный HTTP-адрес.

▪ Доменное имя или IP-адрес TLS-сервера в сети, доступной для ПК клиента. Опционально может быть указан номер порта через двоеточие; по умолчанию используется 443.

▪ 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.

o Все параметры являются обязательными, должны следовать в перечисленном порядке и разделяться символом «;».

o Каждая строка-запись о бизнес-системе должна заканчиваться последовательностью CRLF. Весь файл конфигурации должен также заканчиваться дополнительной последовательностью CRLF.

4.11.4. Прерывание криптоопераций

ПО клиента бывает нужно прервать криптооперацию до того, как она закончилась (например, при нажатии клиентом кнопку "Отмена").

Если для прерывания операции ПО клиента просто перестанет обращаться по идентификатору прерываемой операции, "забудет" его, то на устройстве и в приложении "старт" контекст криптооперации всё равно будет оставаться открытым. Если таких "забытых" контекстов накопится критическое количество, то клиент не сможет начать очередную криптооперацию, т.к. будет получать ошибку "CO_NO_FREE_CONTENT".

Все контексты криптоопераций закрываются принудительно только при переподключении устройства, или при завершении сессии учётной записи. Однако, чтобы для закрытия ненужной криптооперации не прибегать к таким мерам, существует команда прерывания конкретной криптооперации.

Она подходит для прерывания любой из перечисленных операций (подписание, проверка подписи, зашифрование, расшифрование) на любом этапе её выполнения. После успешного выполнения функции переданный идентификатор криптооперации становится недействительным, и клиент может безбоязненно забыть о нём.

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	BREAK_CRYPTOOP_H_ID	идентификатор функции	
ctx_handle	HANDLE	8 байт	идентификатор контекста прерываемой операции	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID HTTP/1.1\r\n
Content-Length: xxx\r\n\r\n
id="BREAK_CRYPTOOP_H_ID"&ctx_handle="zxcvbnma"
```

Выход:

Параметр	Тип	Ограничение	Назначение
retcode	ENUM	-32768..32767	код возврата функции

Пример:

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

Примечание:

- Если имеются сомнения, что операция закрыта, то эту функцию можно вызывать неоднократно, в т. ч. с идентификаторами уже закрытых операций. Она будет выдавать "Ок", если гарантирует, что данный идентификатор не принадлежит ни одной из выполняющихся на устройстве криптоопераций.

4.12. Наследные функции (не поддерживаются и не рекомендуются к использованию)

4.12.1. Информация о контексте наследного подписания (не рекомендуется к использованию)

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	GET_CTX_INFO_ID	идентификатор функции	

Пример:

```
POST http://localhost:28016/vpnkeylocal/<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 функции

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”
```

4.12.2. Инициализация наследной ЭП (не рекомендуется к использованию)

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	INIT_SIGN_ID	идентификатор функции	

Пример:

```
POST http://localhost:28016/vpnkeylocal/<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 функции

Пример:

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

4.12.3. Данные для наследной ЭП (не рекомендуется к использованию)

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SET_SIGN_DATA_ID	идентификатор функции	
data	BASE64	15360 байтов бинарных данных	данные для подписи в base64	
blocknum	NUMBER	1..2^32 - 1	номер посылки	Опционально, если присутствует, то проверяется.

Пример:

```
POST http://localhost:28016/vpnkeylocal/<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 функции

Пример:

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

4.12.4. Вычисление наследной ЭП (не рекомендуется к использованию)

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	CALC_SIGN_ID	идентификатор функции	
obj_id	HANDLE	8 байт	идентификатор сертификата для подписи	
ccdata	BASE64	исходные 32 байта контрольной суммы затем кодированные в base64	данные контрольной суммы в base64	Опционально, если присутствует, то проверяется

Пример:

```
POST http://localhost:28016/vpnkeylocal/<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 функции

Пример:

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

4.12.5. Получить наследную ЭП (не рекомендуется к использованию)

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	GET_SIGN_D_ID	идентификатор функции	

Пример:

```
POST http://localhost:28016/vpnkeylocal/<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 функции

Пример:

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

4.12.6. Инициализация наследной проверки ЭП (не рекомендуется к использованию)

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	INIT_CHECK_ID	идентификатор функции	

Пример:

```
POST http://localhost:28016/vpnkeylocal/<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 функции

Пример:

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

4.12.7. Данные для наследной проверки ЭП (не рекомендуется к использованию)

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	SET_CHECK_DATA_ID	идентификатор функции	
data	BASE64	15360 байтов бинарных данных	данные для проверки подписи в base64	
blocknum	NUMBER	1..2^32 - 1	номер посылки	Опционально, если присутствует, то проверяется.

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID 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 функции

Пример:

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

4.12.8. Проверка наследной ЭП (не рекомендуется к использованию)

Вход:

SID	Обязательно SID2+"/"

Параметр	Тип	Ограничение	Назначение	Дополнительно
id	ID_FORMAT	CHECK_SIGN_ID	идентификатор функции	
cert_data	PEMDER	15360 байтов бинарных данных	данные сертификата в base64	
sign_data	BASE64	88 байт	данные ЭП для проверки в base64	

Пример:

```
POST http://localhost:28016/vpnkeylocal/SID 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
retcode	ENUM	2 байт	код возврата функции

Пример:

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

5. Сценарии работы с устройствами «VPN-Key-TLS»

5.1. Установление сессии на устройстве

- [Необязательно] Определить SID0 (входит в состав URL в файле sslgate.url на флеш-диске устройства):

```
[InternetShortcut]
URL=http://localhost:28016/vpnkeylocal/<SID0>/
```

- Вызвать метод API для получения списка доступных учётных записей:

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID0 или SID2>/ 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===

- Вызвать метод API для открытия сессии, указав необходимое имя и PIN-код:

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID0 или SID2>/ 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===

- Полученный идентификатор сессии SID2 необходимо использовать для следующих сценариев.

5.2. Установление сессии на устройстве и канала TLS с поддержкой мультисистемности

- Вход на устройство произведён, SID2 известен. Получаем с устройства список установленных на нём бизнес-систем.

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID2>/ 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===

- Из полученного списка бизнес-систем необходимо выбрать для дальнейшей работы и указать ее идентификатор

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID2>/ 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===

- 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 бизнес-системы.

5.3. Формирование ключевой пары

- При необходимости, выполнить действия по сценарию «Установление сессии на устройстве» (см. пункт 5.1), получить в результате SID2.

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

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

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID2>/ HTTP/1.1
Content-Type: text/html;
Content-Length: xxx

id=CREATE_PAIR_EX_ID&dn=MIIBkjENMAsGA1UEAwwESFNCQzEVMBMGA1UEBAwM0KLRj9C%2F0L
rQuNC9MSowKAYDVQQqDCHQlNC20Y3RhCDQktC10L3QtdC00LjQutGC0L7QstC40YcxCzAJBgNVBA
YTAlJVMRUwEwYDVQQHDAzQntC80YHQutCw0Y8xGDAWBgNVBAgMDzU1INCe0LzRgdC60LDRjzErMC
kGA1UECQwi0J3QsNCx0LXRgNC10LbQvdCw0Y8g0YPQu9C40YbQsCwgMTENMAsGA1UECgwESFNCQz
EkMCIGA1UECwwb0J7RgtC00LXQuyDQvtC%2F0LXRgNCw0YbQuNC5MTAwLgYDVQQMDCfQodGC0LDR
gNGI0LjQuSDQvtC%2F0LXRgNCw0YbQuNC%2B0L3QuNGB0YIxGjAYBggqhQMDgQMBARIMMDA3NzEw
MzUzNjA2MRgwFgYFKoUDZAESDTEwMjc3MzkyMDc0NjIxFjAUBgUqhQNkAxILMTEyMjMzNDQ1OTUx
HjAcBgkqhkiG9w0BCQEWD29AbG9jYWxob3N0LmNvbQ%3D%3D&req_type=4&pk_alg=3&hash_al
g=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===

- [Необязательно] Используя полученный идентификатор ("хендл") запроса выгрузить запрос из устройства

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID2>/ 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===

- Выгруженные данные запроса можно передать в УЦ для выпуска сертификата.

- Следует помнить, что все идентификаторы ("хендлы") объектов, выданные устройством, актуальны только в течение данной сессии. После завершения данной сессии и открытии следующей сессии все идентификаторы теряют актуальность. Поэтому, чтобы получить идентификатор 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).

5.4. Установка сертификата к уже сформированной ключевой паре

- 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.

- При необходимости, выполнить действия по сценарию «Установление сессии на устройстве», получить в результате SID2.

- Вызвать метод API для установки выданного в УЦ сертификата на устройство:

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID2>/ HTTP/1.1
Content-Type: text/html;
Content-Length: xxx

id=SET_CERT_D_ID&data=-----BEGIN                                CERTIFICATE----
-%0d%0aMIID4jCCA5GgAwIBAgIIMDBDQTg2NjIwCAYGKoUDAgIDMFAxJDAiBgNVBAMMG9Cf%0d%0
a0LXRh9Cw0YLRjCDQo9CmINCh0JEg0KDQpDEoMCYGA1UEIQwf0KLQtdGF0L3QuNGH%0d%0a0LXRg
dC60LjQuSDQutC70Y7RhzAeFw0xMzAxMjgwMDAwMDBaFw0xNzAxMjgwMDAw%0d%0aMDBaMIIBAjE
LMAkGA1UEBhMCUlUxRTBDBgNVBAMMPNCf0LDQvdC60YDQsNGC0L7Q%0d%0asiDQkNGA0YHQtdC90
LjQuSDQkNC70LXQutGB0LDQvdC00YDQvtCy0LjRhzEbMBkG%0d%0aA1UEBAwS0J/QsNC90LrRgNC
w0YLQvtCyMUUwQwYDVQQqDDzQn9Cw0L3QutGA0LDR%0d%0agtC+0LIg0JDRgNGB0LXQvdC40Lkg0
JDQu9C10LrRgdCw0L3QtNGA0L7QstC40Ycx%0d%0aFjAUBgNVBAcMDdCc0L7RgdC60LLQsC4xIzA
hBgNVBAkMGtGD0LsuINCS0LDQstC4%0d%0a0LvQvtCy0LAsIDE5MQswCQYFKoUDZAMTADBjMBwGB
iqFAwICEzASBgcqhQMCAiMC%0d%0aBgcqhQMCAh4BA0MABECdhi5+fSybOCVCj6mw9zwYTrhuIBB
O9rruCthuTT6eL2EI%0d%0aV1uxHb5hkAzzEUwGjpyExesGZ6Fj/Mf83QsaCAJUo4IBlzCCAZMwH
QYDVR0OBBYE%0d%0aFLK1qLk8DBIaQqTiLeubs2x/ATAgMA8GA1UdEwEB/wQFMAMBAf8wGAYDVR0
lBBEw%0d%0aDwYEVR0lAAYHKoUDA3sFATAOBgNVHQ8BAf8EBAMCAcYwMwYDVR0fBCwwKjAooCag%
0d%0aJIYiaHR0cDovL3d3dy5zYnJmLnJ1L2NhLzAwMDB4NTA5LmNybDBBBgcqhQMDewMB%0d%0aB
```

DYMNDAwQ0E4NjYyadCi0LXQutGD0YnQsNGPINGC0LXRgdGC0L7QstCw0Y8g0JDQ%0d%0akdChINC
h0JEwIwYFKoUDZG8EGgwY0JHQuNC60YDQuNC/0YIt0JrQodCRLdChMB0G%0d%0aA1UdIAQWMBQwC
AYGKoUDZHEBMAgGBiqFA2RxAjBEBgUqhQNkcAQ7MDkMAAwY0JHQ%0d%0auNC60YDQuNC/0YIt0Jr
QodCRLdChDAAMGdCk0KHQkSDQoNCkINCh0KQvMTI0LTEw%0d%0aNTEwFAYHKoUDA3sDBAQJBgcqh
QMDewUHMB8GA1UdIwQYMBaAFFep7BWqnjB7CAzR%0d%0a99V6SDFMCkOhMAgGBiqFAwICAwNBAFL
M8HwhTnIzrKS/BBHZ4X5ETrpAhj4WM5Ni%0d%0abTwAHitgMUAJqxrKbGevkOmVzHqFQS1D9aorx
VWwBk6mrxzxJkI=%0d%0a-----END CERTIFICATE-----
===пример запроса; окончание===

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

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

5.5. Поиск сертификата

- При необходимости, выполнить действия по сценарию «Установление сессии на устройстве» (см. пункт 5.1), получить в результате SID2.

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

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID2>/ HTTP/1.1
Content-Type: text/html;
Content-Length: xxx

id=GET_CERT_ID&serial=dZI3swuXnzKS8w==
```
===пример запроса; окончание===

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

- 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.

5.6. Формирование ЭП в CMS

- На устройстве должен быть установлен хотя бы один личный сертификат ЭП.

- При необходимости, выполнить действия по сценарию «Установление сессии на устройстве» (см. пункт 5.1), получить в результате SID2.

- Выполнить действия по сценарию «Поиск сертификата» и узнать идентификатор личного сертификата ЭП.

- Вызвать метод API для инициализации расчета ЭП:

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID2>/ 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===

- 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/<SID2>/ HTTP/1.1
Content-Type: text/html;
Content-Length: xxx

id=SET_SIGN_DATA_H_ID&blocknum=12&ctx_handle=abcdefgh&data=S0FLfC98RS1UTyBUQ
U0gREFISGJsRS4uLg==
```
===пример запроса; окончание===

В ответ возвращается суммарный объем уже переданных на расчет данных:

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

- Вызвать метод API для расчета ЭП:

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID2>/ 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===

- 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/<SID2>/ 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===

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

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID2>/ HTTP/1.1
Content-Type: text/html;
Content-Length: xxx

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

В ответ возвращаются данные ЭП в base64:

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

- 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 записан 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-----".

5.7. Проверка ЭП в CMS

- При необходимости, выполнить действия по сценарию «Установление сессии на устройстве» (см. пункт 5.1), получить в результате SID2.

- Вызвать метод API для инициализации проверки ЭП:

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID2>/ HTTP/1.1
Content-Type: text/html;
Content-Length: xxx

id=INIT_CHECK_H_ID&cms_data=MIIFvAYJKoZIhvcNAQcCoIIFrTCCBakCAQExDjAMBggqhQMH
AQECAgUAMAsG
```
===пример запроса; окончание===

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

- 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/<SID2>/ HTTP/1.1
Content-Type: text/html;
Content-Length: xxx

id=SET_CHECK_DATA_H_ID&blocknum=12&ctx_handle=abcdefgh&data=S0FLfC98RS1UTyBU
QU0gREFISGJsRS4uLg==
```
===пример запроса; окончание===

В ответ возвращается суммарный объем уже переданных на проверку данных

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

- Вызвать метод API для проверки ЭП (код возврата "1" в ответ именно на эту функцию означает верность подписи):

===пример запроса; начало===
```
POST http://localhost:28016/vpnkeylocal/<SID2>/ 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===

6. Аргументы командной строки для приложения "Старт"

Существует возможность ограничить функциональность приложения "Старт" с помощью ключей, передаваемых ему при запуске из командной строки. Эта возможность применима в случае, если приложение "Старт" запускается для работы внешнего ПО, "толстого клиента", дублирующего часть функционала приложения "Старт", либо в случае наличия ограничений аппаратно-программной конфигурации ПК клиента, не позволяющих исполнять на ПК весь задуманный функционал приложения "Старт", либо для решения конфликтных ситуаций.

Ключи командной строки могут следовать в произвольном порядке и должны быть разделены пробелами.

Пример:

```
start.exe /silent /nores
```

Перечисление и назначение ключей:

- "**/silent**" - Запретить действия, связанные с интерфейсом ОС, предпринимаемые по инициативе приложения "Старт", такие как:
  - Открытие страницы администрирования в браузере по умолчанию
  - Вывод сообщений ОС в случае ошибок при запуске или необходимости сообщения пользователю причин возможного ограничения функциональности
- "**/ignore_ccid**" - Запретить использование протокола CCID при поиске устройства на ПК
- "**/ignore_libusb**" - Запретить использование протокола LibUsb при поиске устройства на ПК
- "**/nores**" - Не использовать продвинутые графические ресурсы интерфейса администрирования в браузере и пользовательского интерфейса ОС (если браузер клиента имеет ограничения по функционалу, либо ПК клиента -- по количеству оперативной памяти)
- "**/CMD**" - Вызов Командного Интерпретатора (см. 7)

7. Командный Интерпретатор

Существует возможность доступа к высокоуровневым функциям устройства, используя консоль ОС (командную строку). Для этого служит модуль Командного Интерпретатора (КИ), встроенный в приложения "Старт" для всех поддерживаемых ОС, записанных на внешний носитель Устройства. Для вызова КИ служит ключ командной строки "/CMD".

Интерфейс:

```
start.exe [/опции] /CMD <команда_и_аргументы> [>имя_файла_для_выдачи_результата]
```

Пример:

```
start.exe /CMD "LOGIN&user=1&PIN=123456" >answer.txt
```

Ответ:

```
SID2:     aq12wsDE34RFgt56yhJU78iKlo90mnBVcx
RESULT:   1
```

Пояснения:

- О входных аргументах:
  - Ключи командной строки приложения "Старт", имена команд КИ и их аргументов нечувствительны к регистру.
  - Значения аргументов могут быть чувствительны.
  - Сразу за именем аргумента без разделителей должен следовать символ "=" и значение аргумента.
  - Пары "аргумент=значение" разделяются символом "&".
  - Если значение какого-либо аргумента должно содержать символ "&" (используемый в интерфейсе КИ для разделения аргументов), то для его передачи используется 2 символа "&", следующие подряд.
  - Весь блок аргументов команды (в примере: "LOGIN&user=1&PIN=123456") должен быть заключён в одни общие двойные кавычки (код символа: 0x22). (это сделано из-за ограничений парсера командной строки Windows)
  - Если значение какого-либо аргумента должно содержать символ кавычки ("), то для его передачи используется 2 символа кавычки, следующие подряд.
  - Порядок входных и выходных аргументов недетерминирован.
  - На MacOs некоторые символы требуют дополнительного экранирования символом обратного слеша (код символа: 0x5C):
    - **!** (код: 0x21)
    - **"** (код: 0x22)
    - **&** (код: 0x26)
    - **'** (код: 0x27)
    - **(** (код: 0x28)
    - **)** (код: 0x29)
    - ***** (код: 0x2A)
    - **;** (код: 0x3B)
    - **<** (код: 0x3C)
    - **>** (код: 0x3E)
    - **[** (код: 0x5B)
    - **]** (код: 0x5D)
    - **`** (код: 0x60)
    - **|** (код: 0x7C)

    Таким образом, например, вызов КИ, на Windows имеющий вид:

    "`start.exe /CMD "LOGIN&user=1&PIN=!~Az'}" >answer.txt`"

на MacOs будет выглядеть как:

```
"/Volumes/CDROM/Start.app/Contents/MacOS/Start                    /CMD
\"LOGIN\&USER=1\&PIN=\!~Az\'}\" >answer.txt"
```

- О результатах выполнения команд:
  - Имена выдаваемых полей и их значения форматируются в виде текстовой таблицы для удобства просмотра.
  - В последней строке ответа КИ выдаёт код возврата (RESULT). Он может иметь один из следующих вариантов:

1	Операция выполнена успешно
-2	Ошибка при подготовке среды выполнения КИ
-3	Команда не распознана (в т. ч. из-за ошибки в синтаксисе при записи аргументов, например, отсутствия одной, или обеих кавычек)
-4	Ошибка при разборе/подготовке аргументов команды
-5	Ошибка при выполнении команды (подробнее смотрите логи приложений "Старт": выполняющего команду КИ и принимающего от него команды HTTP-API)

  - Если поле "RESULT" отсутствует в ответе, или его значение не равно "1", то наличие и валидность остальных полей не гарантируется. Если причина прерывания КИ - ошибочный код возврата, пришедший от API, то в ответ будет добавлено поле "API_retcode" с тем кодом. Коды возврата от АПИ описаны в "VPNKEY-httpinterface.docx", глава "Коды возврата".
- О перенаправлении вывода:
  - Если в конце командной строки передана предваряемая символом ">" текстовая подстрока (в примере это "answer.txt"), то эта подстрока будет трактована как имя файла, в который КИ запишет результат выполнения команды. Если имя файла не передано, то результат будет выведен в консоль.
  - На Windows приложение "Старт" не является консольным, поэтому сразу же после запуска возвращает управление ОС. Это исключает возможность запускающему приложению (ЗП) получить в свою консоль результат выполнения КИ. Поэтому для автоматического получения от КИ результата ЗП может использовать только вывод в файл, описанный в предыдущем абзаце. Узнать об окончании выполнения КИ ЗП сможет, как только в ответ на попытку открыть файл выдачи на запись (для "fopen" - ключ "r+b") ОС вернёт не код 32 ("**ERROR_SHARING_VIOLATION**"), а 0 ("**ERROR_SUCCESS**").
  - КИ может выводить, кроме результатов выполнения команд, дополнительную человеко-читаемую текстовую информацию, а также ожидать нажатия кнопки для продолжения. При использовании (дополнительно к "/CMD") ключа "/SILENT" человеко-читаемые комментарии не будут выводиться, а "любая кнопка" будет считаться нажатой.
  - Некоторые команды, могущие выдавать человеко-нечитаемые данные (PEM или бинарные), имеют возможность делать это в отдельный "файл выдачи", имя которого задаётся опциональным параметром "OUTFILE". При отсутствии этого параметра вывод человеко-нечитаемых данных будет произведён в консоль КИ.
- Прочее:
  - Для выполнения функционала КИ необходимо наличие в ОС работающего приложения "Старт". Если при запуске КИ оно не было запущено, то КИ запускает другим потоком "Старт" как сервис, выполняет команду и завершает весь процесс. На это тратится значительное время (до нескольких

секунд), что может на порядки замедлить выполнение КИ. Поэтому в целях ускорения работы рекомендуется запускать КИ при запущенном заранее приложении "Старт".

- Запущенное приложение "Старт" завершается автоматически при отключении Устройства от ПК, либо при выполнении КИ-команды "KILL_ICS".

Управление сессией пользователя

Получение списка доступных учетных записей

Команда:

PIN_LIST

Входные параметры:

Отсутствуют

Выходные параметры:

Список учётных записей с пояснениями о состоянии каждой.

Пример:

```
start.exe /CMD "PIN_LIST"
```

Ответ:

```
PIN1:     Ok
PIN2:     Blocked
PIN3:     Purged
PIN5:     Ok
RESULT:   1
```

Открытие сессии

Команда:

LOGIN

Входные параметры:

USER	Номер пользователя для открытия сессии
PIN	PIN-код для открытия сессии

Выходные параметры:

SID2	Идентификатор открытой сессии

Пример:

```
start.exe /CMD "LOGIN&user=1&pin=123456"
```

Ответ:

```
SID2:     aq12wsDE34RFgt56yhJU78iKlo90mnBVcx
RESULT:   1
```

Завершение сессии

Команда:

LOGOUT

Входные параметры:

Отсутствуют

Выходные параметры:

Отсутствуют

Пример:

```
start.exe /CMD "LOGOUT"
```

Ответ:

```
RESULT:   1
```

Пояснения:

После завершения сессии теряет актуальность идентификатор сессии (полученный как "SID2" от команды "LOGIN"), идентификаторы (хендлы) всех объектов, выданные в течение этой сессии, а также ТЛС-тоннель, возможно, открытый командой "BS_USE".

Управление сессией TLS

Получение списка доступных бизнес-систем

Команда:

BS_LIST

Входные параметры:

SID Идентификатор сессии, полученный как "**SID2**" от команды "LOGIN"

Выходные параметры:

Список доступных бизнес-систем.

Пример:

```
start.exe /CMD "BS_LIST&SID=aq12wsDE34RFgt56yhJU78iKlo90mnBVcx"
```

Ответ:

```
bs:       5. "CryptoPRO 2012_256_au"
bs:       6. "Amicon FPSU"
bs:       19. "Local:443"
RESULT:   1
```

Пояснения:

Каждая доступная бизнес-система описывается одной строкой, начинающейся на "bs:". Далее (возможно, через несколько пробелов) следует её номер, уникальный в пределах ответа, заканчивающийся точкой. Он предназначен для использования как аргумент для команды "BS_USE" (см.ниже).
Далее - человекочитаемое имя БС.

Открытие сессии с бизнес-системой и отправка на неё проверочного запроса GET

Команда:

BS_USE

Входные параметры:

SID	Идентификатор сессии (полученный как "**SID2**" от команды "**LOGIN**")
BS_ID	Номер открываемой бизнес-системы (полученный от команды "**BS_LIST**")
[URL]	Относительный адрес для запроса у сервера открытой бизнес-системы. (при отсутствии запрашивается "/index.htm")
[OUTFILE]	Имя "файла выдачи" для записи ответа от сервера бизнес-системы. (при отсутствии этого поля ответ выводится непосредственно в консоль)

Выходные параметры:

data	Ответ, присланный сервером бизнес-системы (включая HTTP-заголовок).

Пример:

```
start.exe /CMD "BS_USE&SID=aq12wsDE34RFgt56yhJU78iKlo90mnBVcx&BS_ID=5"
```

Ответ:

```
data:

HTTP/1.1 200 OK
Content-Length: 361
Content-Type: text/html

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<html><head><meta http-equiv="Content-Type" content="text/html; charset=UTF-
8"><title>The Last Page</title></head>
<body><h1>The Last Page</h1><h3>You have reached The End of internet</h3>
There are no more links.<br><br>
You must now turn off your computer and go to something useful.
</body></html>
RESULT:   1
```

Пояснения:

1. На другой стороне открываемого TLS-тоннеля должен находиться сервер, поддерживающий протокол TLS версии 1.2 (RFC 5246), криптоалгоритмы по ГОСТ Р 34.10-2012 / ГОСТ Р 34.11-2012, и имеющий сертификат шифрования с цепочкой доверия, начинающейся корневым сертификатом, установленным на устройстве. Если настройки сервера включают в себя аутентификацию клиента, то сервер должен иметь у себя корневой сертификат от цепочки доверия, которой принадлежит личный TLS-сертификат клиента.
2. После успешного выполнения этой команды и до завершения приложения "Старт" или сессии все HTTP-запросы, направляемые на адрес http://localhost:localport/ (кроме http://localhost:localport/vpnkeylocal/) будут перенаправляться через ТЛС-тоннель на сервер бизнес-системы, с которой эта команда открыла сессию.

Администрирование учётных записей

Смена PIN-кода по PUK-коду (восстановление)

Команда:

```
CH_PIN_BY_PUK
```

Входные параметры:

USER	Номер учётной записи для восстановления PIN
PUK	Действующий код восстановления PUK

NEW_PIN Новый PIN-код

Выходные параметры:

Отсутствуют

Пример:

```
start.exe /CMD "CH_PIN_BY_PUK&USER=3&PUK=123456789012&NEW_PIN=123123"
```

Ответ:

```
RESULT:   1
```

Комментарий:

Выполнение команды возможно только если на устройстве не открыта сессия ни одного пользователя.

Смена PIN-кода по PIN-коду

Команда:

CH_PIN_BY_PIN

Входные параметры:

SID	Идентификатор сессии (полученный как "**SID2**" от команды "**LOGIN**")
PIN	Действующий PIN-код открывшей сессию учётной записи
NEW_PIN	Новый PIN-код
[PUK]	Для форматированных УЗ необходимо задать PUK. Для смены уже изменявшегося пользователем PIN-кода PUK не требуется.

Выходные параметры:

Отсутствуют

Пример:

```
start.exe /CMD
"CH_PIN_BY_PIN&SID=aq12wsDE34RFgt56yhJU78iKlo90mnBVcx&PIN=123456&NEW_PIN=123123"
```

Ответ:

```
RESULT:   1
```

Смена PUK-кода по PUK-коду

Команда:

CH_PUK_BY_PUK

Входные параметры:

SID	Идентификатор сессии (полученный как "**SID2**" от команды "**LOGIN**")
PUK	Действующий PUK-код открывшей сессию учётной записи
NEW_PUK	Новый PUK-код

Выходные параметры:

Отсутствуют

Пример:

```
start.exe /CMD
"CH_PUK_BY_PUK&SID=aq12wsDE34RFgt56yhJU78iKlo90mnBVcx&PUK=123456789012&NEW_PUK=12
3123123123"
```

Ответ:

```
RESULT:   1
```

Форматирование учётной записи

Команда:

```
FORMAT_PIN
```

Входные параметры:

`USER` Номер учётной записи для форматирования

Выходные параметры:

Отсутствуют

Пример:

```
start.exe /CMD "FORMAT_PIN&USER=3"
```

Ответ:

```
RESULT:   1
```

Комментарий:

Выполнение команды возможно только если на устройстве не открыта сессия ни одного пользователя.

Обновление встроенного ПО устройства

Команда:

```
SET_UPDATE
```

Входные параметры:

`INFILE` Имя файла с обновлением

Выходные параметры:

Отсутствуют

Пример:

```
start.exe /CMD "SET_UPDATE&INFILE=c:\users\Vasya\Desktop\token_update_560_716.fw"
```

Ответ:

```
RESULT:   1
```

Использование хранилища сертификатов

Просмотр объектов хранилища

Команда:

```
OBJ_LIST
```

Входные параметры:

TYPE Тип объектов для просмотра:
0 – сертификаты ЭП;
1 – сертификаты TLS;
2 – корневые сертификаты;
3 – запросы ЭП;
4 – запросы TLS;
5 – сертификаты УЦ;
6 – первичные TLS;
7 – транспортные ЭП

[SID] Идентификатор сессии (полученный как "**SID2**" от команды "**LOGIN**")
Обязателен только для типов: 0,1,3,4.

Выходные параметры:

Предпросмотр выгруженных объектов запрошенного типа.

Пример:

```
start.exe /CMD "OBJ_LIST&TYPE=2"
```

Ответ:

```
1.
Серийный номер  : 4e 6d 47 8b 26 f2 7d 65 7f 76 8e 02 5c e3 d3 93
Алгоритм        : ГОСТ Р 34.10-2012 и ГОСТ Р 34.11-2012 (256)
Срок действия   : 2018.07.06-15:18:06 - 2036.07.01-15:18:06
Субъект(CN)     : Минкомсвязь России
Открытый ключ   : 75 39 2a 45 a7 b9 a2 95 7d f7 10 fd 22 92 07 ba
                  1d b6 5a 71 8a 7d 7d 58 fc b1 46 b9 45 61 57 ac
                  1d bb 48 a5 f9 4a fb 48 19 ea 6a 29 eb fa f5 14
                  98 78 71 ca 47 e8 d3 f5 85 f6 36 e4 8a f7 03 8d
Хендл           : JmF50jGB
================================================================================
2.
Серийный номер  : 4e cf 85 c9 32 2a 8e 90 43 77 ee 78 ec d2 82 b9
Алгоритм        : ГОСТ Р 34.10-2001 и ГОСТ Р 34.11-1994 (256)
Срок действия   : 2013.02.11-17:30:28 - 2043.02.11-17:40:28
Субъект(CN)     : УЦ КРИПТО-ПРО
Открытый ключ   : f6 9f d5 76 12 57 ef 14 7f 93 ff 47 cb 49 bb 13
                  46 93 f5 1b 96 50 75 d4 4d 08 db 4d 69 10 a8 b8
                  f7 2f bd 73 e1 e7 26 2c 6a eb ce 5f d5 76 df c4
                  ba e5 81 ef 4d 98 5d 16 b3 c3 55 0a 4e 3f 9a 62
Хендл           : b4ydH2nf
================================================================================
3.
Серийный номер  : 76 87 b9 ad 79 9c 32 e5 b9 b9
Алгоритм        : ГОСТ Р 34.10-2012 и ГОСТ Р 34.11-2012 (256)
Срок действия   : 2018.08.15-03:00:00 - 2026.08.15-03:00:00
Субъект(CN)     : Корневой сертификат АРМ-И: 17:26:53 15.08.2018
Открытый ключ   : bd 81 c6 7a 7b 41 7b c7 2e 5d 50 b3 89 f5 90 2c
                  39 53 79 ce 12 e9 16 a4 78 2c 53 1d 44 0f 30 8f
                  34 af 3c a6 80 57 b5 3c cc f1 23 f4 c7 01 65 b5
                  e1 31 d4 fc 6b ed 46 e1 a6 35 62 f8 76 f6 5e 8f
Хендл           : XAswX9kz
================================================================================
RESULT:   1
```

Пояснения:

"Предпросмотр" представляет собой таблицу с данными объектов - форматированный блок текста, содержащий про каждый объект запрошенного типа следующие строки, (каждая из которых заканчивается символом конца строки):

- порядковый номер в списке (десятичное число, заканчивающееся точкой)

- серийный номер (если есть)
- криптографический алгоритм
- дата создания (и дата окончания действия, если есть)
- данные издателя (если есть)
- данные субъекта (если есть)
- открытый ключ (если есть)
- идентификатор криптопровайдера (если есть)
- внутренний идентификатор объекта "хендл"
- строка-разделитель, состоящая из не менее чем 64 символов "=" подряд.

Получение объекта из хранилища

Команда:

GET_OBJ

Входные параметры:

OBJ_ID	Хендл объекта (полученный из списка от команды "**OBJ_LIST**")
[SID]	Идентификатор сессии (полученный как "**SID2**" от команды "**LOGIN**") *Обязателен только для объектов типов: 0,1,3,4 (см."**OBJ_LIST**").*
[OUTFILE]	Имя "файла выдачи" для записи выгруженных данных

Выходные параметры:

data	Выгруженный объект (в формате PEM)

Пример:

```
start.exe /CMD "GET_OBJ&OBJ_ID=XAswX9kz&OUTFILE=my_cert.cer"
```

Ответ:

```
data:     <written to "my_cert.cer">
RESULT:   1
```

Модификация хранилища сертификатов

Генерирование ключевой пары и квалифицированного запроса на сертификат PKCS10

Команда:

CREATE_PAIR

Входные параметры:

gost	Алгоритм генерируемой пары: "2" - ГОСТ-2012-256, "3" - ГОСТ-2012-512
usage	Назначение: "0" - ТЛС, "1" - ЭП, "2" - шифрование, "3" - ЭП и шифрование
DN*	Начало тройки параметров, задающих OID для размещения в секции "DN": Например: **DN_OID*=2.5.4.3 // OID "CommonName"** **DN_TAG*=12 // Кодировка: ASN1_TAG_UTF8_STRING** **DN_VAL*=Ivan%20Ivanov // Значение параметра** *Вместо "*" д.б. число, общее для всей тройки и уникальное на весь запрос. **Байты в полях "VAL", содержащие значения, не являющиеся латинскими буквами или цифрами в кодировке ASCII (в т. ч. знаки препинания и пробелы) должны быть закодированы последовательностью: "%XX", где "XX" - 16-ричный код байта.

***Номера троек должны идти подряд и начинаться с "0"

A* Начало тройки параметров, задающих OID для секции "ATTR" (опц.)

A2* Начало тройки параметров, задающих OID для секции "ATTR2" (опц.)

SID Идентификатор сессии (полученный как "**SID2**" от команды "**LOGIN**")

Выходные параметры:

obj_id Идентификатор (хендл) созданного объекта

Пример:

```
start.exe /CMD "CREATE_PAIR
&SID=aq12wsDE34RFgt56yhJU78iKlo90mnBVcx&gost=2&usage=3
&DN_OID0=2.5.4.3
&DN_TAG0=12
&DN_VAL0=%22Silver%20Sparks%22Ltd
&DN_OID1=2.5.4.6
&DN_TAG1=19
&DN_VAL1=RU
&DN_OID2=2.5.4.8
&DN_TAG2=12
&DN_VAL2=K123PA95
&DN_OID3=1.2.643.100.4
&DN_TAG3=18
&DN_VAL3=777666555
&DN_OID4=2.5.4.4
&DN_TAG4=12
&DN_VAL4=IvanovII"
```

Ответ:

```
obj_id:   uZvMKHsw
RESULT:   1
```

Удаление ключевой пары и соответствующего ей сертификата или только личного сертификата

Команда:

DEL_OBJ

Входные параметры:

obj_id Идентификатор (хендл) удаляемого объекта, (полученный из списка от команды "**OBJ_LIST**")

SID Идентификатор сессии (полученный как "**SID2**" от команды "**LOGIN**")

Выходные параметры:

Отсутствуют

Пример:

```
start.exe /CMD "DEL_OBJ&SID=aq12wsDE34RFgt56yhJU78iKlo90mnBVcx&obj_id=5fabYm19"
```

Ответ:

```
RESULT:   1
```

Установка сертификата

Команда:

SET_CERT

Входные параметры:

INFILE Имя файла с устанавливаемым сертификатом (поддерживаемые кодировки: PEM, DER, Base64)

[SID] Идентификатор сессии (полученный как "**SID2**" от команды "**LOGIN**")
Обязателен только для установки личных сертификатов.

Выходные параметры:

obj_id Идентификатор (хендл) установленного сертификата

Пример:

```
start.exe /CMD "SET_CERT&SID=aq12wsDE34RFgt56yhJU78iKlo90mnBVcx&INFILE=cert.cer"
```

Ответ:

```
obj_id:   uZvMKHsw
RESULT:   1
```

Установка CRL

Команда:

```
SET_CRL
```

Входные параметры:

INFILE Имя файла с устанавливаемым списком отзыва (поддерживаемые кодировки: PEM, DER, Base64)

Выходные параметры:

Отсутствуют

Пример:

```
start.exe /CMD "SET_CRL&INFILE=crl.crl"
```

Ответ:

```
RESULT:   1
```

Установка файла конфигурации бизнес-систем

Команда:

```
SET_MC
```

Входные параметры:

INFILE Имя файла с устанавливаемой конфигурацией бизнес-систем (поддерживаемые кодировки: PEM, DER, Base64)

Выходные параметры:

Отсутствуют

Пример:

```
start.exe /CMD "SET_MC&INFILE=mc.p7s"
```

Ответ:

```
RESULT:   1
```

Криптооперации

Подписание данных в формате CMS

Команда:

SIGN_DATA

Входные параметры:

SID	Идентификатор сессии (полученный как "**SID2**" от команды "**LOGIN**")
INFILE	Имя файла для подписания
OBJ_ID	Идентификатор (хендл) сертификата для подписания, (полученный из списка от команды "**OBJ_LIST**")
[HASCERT]	Длина выгружаемой цепочки доверия, начиная от сертификата-подписанта. "-1" - все до корневого, исключая корневой. "0" - не вкладывать. По умолчанию "1" (вложить только сертификат-подписант).
[OUTFILE]	Имя файла для записи выгруженной подписи (формат: PEM)

Выходные параметры:

data	Электронная подпись (формат: PEM)

Пример:

```
start.exe /CMD
"SIGN_DATA&SID=aq12wsDE34RFgt56yhJU78iKlo90mnBVcx&INFILE=test_data.txt&HASCERT=0&
obj_id=5fabYm19&OUTFILE=sign.p7s"
```

Ответ:

```
data:     <written to "sign.p7s">
RESULT:   1
```

Проверка ЭП в формате CMS

Команда:

CHECK_SIGN

Входные параметры:

SID	Идентификатор сессии (полученный как "**SID2**" от команды "**LOGIN**")
INFILE	Имя файла данных для проверки
SIGNFILE	Имя файла данных с подписью

Выходные параметры:

Отсутствуют.
(Завершение команды без ошибок свидетельствует об успешной проверке подписи).

Пример:

```
start.exe /CMD
"SIGN_DATA&SID=aq12wsDE34RFgt56yhJU78iKlo90mnBVcx&INFILE=test_data.txt&SIGNFILE=s
ign.p7s"
```

Ответ:

```
RESULT:   1
```

Шифрование данных в формате CMS

Команда:

ENCIPHER

Входные параметры:

SID	Идентификатор сессии (полученный как "**SID2**" от команды "**LOGIN**")
INFILE	Имя файла данных для зашифрования
RECIPIENT*	Имя файла сертификата получателя зашифрованных данных. Вместо звёздочки должна быть указана цифра от 0 до 9, уникальная в пределах одного запроса. КИ поддерживает от 1 до 10 получателей.
[OUTFILE]	Имя файла для записи результата зашифрования (формат PEM).

Выходные параметры:

Файл с зашифрованными данными

Пример:

```
start.exe /CMD
"ENCIPHER&SID=aq12wsDE34RFgt56yhJU78iKlo90mnBVcx&INFILE=test_data.txt&RECIPIENT1=
myself1.cer&RECIPIENT2=kontragent.cer&RECIPIENT3=tov_major.cer&OUTFILE=cipher.p7e
"
```

Ответ:

```
data:     <written to "cipher.p7e">
RESULT:   1
```

Расшифрование данных в формате CMS

Команда:

DECIPHER

Входные параметры:

SID	Идентификатор сессии (полученный как "**SID2**" от команды "**LOGIN**")
INFILE	Имя файла данных для расшифрования
[OUTFILE]	Имя файла для записи результата расшифрования.

Выходные параметры:

Файл с расшифрованными данными

Пример:

```
start.exe /CMD
"DECIPHER&SID=aq12wsDE34RFgt56yhJU78iKlo90mnBVcx&INFILE=code.p7e&OUTFILE=res.txt"
```

Ответ:

```
data:     <written to "res.txt">
RESULT:   1
```

Вспомогательные функции

Просмотр информации об устройстве

Команда:

GET_SYS_INFO

Входные параметры:

[SID] Идентификатор сессии (полученный как "**SID2**" от команды "**LOGIN**")
Опционально, может понадобиться только для проверки, актуальна ли на момент запроса сессия, при открытии которой он был выдан.

Выходные параметры:

Таблица с данными про устройство, включающая:
- статусную строку Устройства (STATUS:)
- дату и время инициализации POSIX UTM (INIT_TIME:)
- дату и время на устройстве POSIX UTM (TOKEN_TIME:)
- его серийный номер (SERIAL:)
- номер прошивки (FW_VERSION:)
- номер активного пользователя (CUR_USER:); если сессия не открыта, то -1
- оценка переданного SID (SESSION)
- хеш микрокода устройства (FW_HASH)
- хеш приложения для ПК (PC_HASH)

Пример:

```
start.exe /CMD "GET_SYS_INFO&SID=aq12wsDE34RFgt56yhJU78iKlo90mnBVcx"
```

Ответ:

```
STATUS:     ICc_T32S0000L_C1_VT550HT5BU3
INIT_TIME:  12:00:07 06.12.2021
SERIAL:     TST00000018C
FW_VERSION: 550.259
CUR_USER:   1
SESSION:    96
FW_HASH:    ed 56 d2 83 05 80 47 6d db 6f bf ef 61 ce 5b 02
            70 8f 4f 13 c3 3c b2 82 6e fb 71 93 fc c1 3c b0
PC_HASH:    2E 24 7E E4 AD 9C 7F 4A 9C BB 42 63 73 07 88 6B
            92 68 81 03 1B 85 5C D9 F9 3D 32 70 56 89 BF 22
RESULT:     1
```

Завершение "старта.exe", выполняющегося как сервис

Команда:

```
KILL_ICS
```

Входные параметры:

Отсутствуют

Выходные параметры:

Отсутствуют

Пример:

```
start.exe /CMD "KILL_ICS"
```

Ответ:

```
RESULT:   1
```