ViPNet SIES Unit

Руководство разработчика

infotecs

© АО «ИнфоТеКС», 2021

ФРКЕ.00237-01 33 02

Версия продукта 2.3

Этот документ входит в комплект поставки ViPNet SIES Unit, и на него распространяются все условия лицензионного соглашения.

Ни одна из частей этого документа не может быть воспроизведена, опубликована, сохранена в электронной базе данных или передана в любой форме или любыми средствами, такими как электронные, механические, записывающие или иначе, для любой цели без предварительного письменного разрешения АО «ИнфоТеКС».

ViPNet® является зарегистрированным товарным знаком АО «ИнфоТеКС».

В продукте использовано изобретение, защищенное патентом РФ № 2706176.

Все названия компаний и продуктов, которые являются товарными знаками или зарегистрированными товарными знаками, принадлежат соответствующим владельцам.

АО «ИнфоТеКС»

127083, Москва, улица Мишина, д. 56, стр. 2, этаж 2, помещение IX, комната 29

Телефон: +7 (495) 737-6192, 8-800-250-0260 — бесплатный звонок из России (кроме Москвы)

Веб-сайт: infotecs.ru

Служба поддержки: hotline@infotecs.ru

Содержание

Введение ... **5**
О документе ... 6
Для кого предназначен документ ... 6
Связанные документы ... 6
Соглашения документа ... 6
Обратная связь ... 8

Глава 1. Общие сведения ... **9**
Об интерфейсе ViPNet SIES Unit RESTful API ... 10
Синтаксис HTTP-сообщения запроса ... 10
Синтаксис HTTP-сообщения ответа ... 11
Назначения прикладных связей для методов ViPNet SIES Unit RESTful API ... 13
Коды результата выполнения функций ... 15
Применение ViPNet SIES Unit RESTful API ... 16

Глава 2. Методы ViPNet SIES Unit RESTful API ... **17**
Методы работы с CRISP-сообщениями ... 18
Создание CRISP-сообщения с зашифрованными данными ... 19
Обработка CRISP-сообщения с зашифрованными данными ... 20
Создание CRISP-сообщения с имитовставкой ... 22
Обработка CRISP-сообщения с имитовставкой ... 23
Методы для поточной работы с CRISP-сообщениями ... 24
Создание фрагмента CRISP-сообщения с зашифрованными данными ... 25
Обработка фрагмента CRISP-сообщения с зашифрованными данными ... 27
Создание фрагмента CRISP-сообщения с имитовставкой ... 29
Обработка фрагмента CRISP-сообщения с имитовставкой ... 31
Методы работы с CMS-контейнерами ... 34
Зашифрование в CMS-контейнер с прикрепленными данными ... 35
Расшифрование из CMS-контейнера с прикрепленными данными ... 36
Зашифрование в CMS-контейнер с открепленными данными ... 37
Расшифрование из CMS-контейнера с открепленными данными ... 39
Создание электронной подписи в CMS-контейнере с прикрепленными данными ... 40
Проверка электронной подписи из CMS-контейнера с прикрепленными данными ... 42
Создание электронной подписи в CMS-контейнере с открепленными данными ... 43

Проверка электронной подписи из CMS-контейнера с открепленными данными ...44
Другие криптографические методы 47
Вычисление хэш-кода без указания получателя 47
Проверка хэш-кода без указания отправителя 48
Вычисление хэш-кода с указанием получателя 50
Проверка хэш-кода с указанием отправителя 51
Инициирование аутентификации пользователя 52
Завершение аутентификации пользователя в формате CMS 53
Информационно-управляющие методы 56
Получение сведений о ViPNet SIES Unit 56
Тест связи 57
Обработка защищенного конверта ViPNet SIES MC 58

Приложение А. Глоссарий 61

Введение

О документе 6

Обратная связь 8

О документе

В данном документе вы найдете информацию по использованию ViPNet® SIES Unit RESTful API для криптографической обработки данных защищаемого устройства.

Для кого предназначен документ

Документ предназначен для разработчиков, ведущих проекты по внедрению ПО взаимодействия защищаемого устройства с ViPNet SIES Unit. Предполагается, что разработчик обладает опытом разработки программного обеспечения с использованием RESTful API на любом языке программирования и базовым представлением о криптографической защите информации.

Связанные документы

В таблице ниже перечислены документы, с которыми мы рекомендуем вам также ознакомиться.

Таблица 1. Связанные документы

Документ	Содержание
ViPNet SIES Unit. Руководство администратора	Методика установки и настройки программного обеспечения ViPNet SIES Unit, входящего в состав комплекса ViPNet SIES и предназначенного для защиты информации, обрабатываемой устройствами верхнего уровня индустриальной и IIoT-системы
ViPNet SIES. Сценарии работы	Основные сценарии работы комплекса ViPNet SIES в индустриальной и IIoT-системе

Соглашения документа

Ниже перечислены соглашения, принятые в этом документе для выделения информации.

Таблица 2. Обозначения, используемые в примечаниях

Обозначение	Описание
	Внимание! Указывает на обязательное для исполнения или следования действие или информацию.
	Примечание. Указывает на необязательное, но желательное для исполнения или следования действие или информацию.

Обозначение	Описание
	Совет. Содержит дополнительную информацию общего характера.

Таблица 3. Обозначения, используемые для выделения информации в тексте

Обозначение	Описание
Название	Название элемента интерфейса. Например, заголовок окна, название поля, кнопки или клавиши.
Клавиша+Клавиша	Сочетание клавиш. Чтобы использовать сочетание клавиш, следует нажать первую клавишу и, не отпуская ее, нажать вторую клавишу.
Меню > **Подменю** > **Команда**	Иерархическая последовательность элементов. Например, пункты меню или разделы на панели навигации.
`Код`	Имя файла, путь, фрагмент текстового файла (кода) или команда, выполняемая из командной строки.

Обратная связь

Дополнительная информация

Сведения о продуктах и решениях ViPNet, распространенные вопросы и другая полезная информация собраны на сайте ИнфоТеКС:

- Информация о продуктах ViPNet.
- Информация о решениях ViPNet.
- Часто задаваемые вопросы.
- Форум пользователей продуктов ViPNet.

Контактная информация

Если у вас есть вопросы, свяжитесь со специалистами ИнфоТеКС:

- Единый многоканальный телефон:

  +7 (495) 737-6192,

  8-800-250-0-260 — бесплатный звонок из России (кроме Москвы).
- Служба поддержки: hotline@infotecs.ru.

  Форма для обращения в службу поддержки через сайт.

  Телефон для клиентов с расширенной поддержкой: +7 (495) 737-6196.
- Отдел продаж: soft@infotecs.ru.

Если вы обнаружили уязвимости в продуктах компании, сообщите о них по адресу security-notifications@infotecs.ru. Распространение информации об уязвимостях продуктов компании ИнфоТеКС регулируется политикой ответственного разглашения.

1

Общие сведения

Об интерфейсе ViPNet SIES Unit RESTful API 10

Назначения прикладных связей для методов ViPNet SIES Unit RESTful API 13

Коды результата выполнения функций 15

Применение ViPNet SIES Unit RESTful API 16

Об интерфейсе ViPNet SIES Unit RESTful API

Интерфейс ViPNet SIES Unit RESTful API предназначен для интеграции ViPNet SIES Unit с защищаемыми устройствами (см. глоссарий, стр. 63) индустриальных (см. глоссарий, стр. 63) или IIoT-систем (см. глоссарий, стр. 62) на прикладном уровне при реализации сценариев защиты данных и взаимодействия с системой управления ViPNet SIES MC (см. глоссарий, стр. 62).

Взаимодействие с ViPNet SIES Unit RESTful API организовано по протоколу HTTP/1.1. Служба ViPNet SIES Unit выступает пассивной стороной взаимодействия, отвечая на запросы со стороны активного клиента — ПО защищаемого устройства, выступающего инициатором взаимодействия.

В ходе взаимодействия прикладная служба ViPNet SIES Unit отвечает на HTTP-запросы, посылаемые на сетевой порт `9876` для адреса `localhost` устройства с установленным ViPNet SIES Unit. Номер сетевого порта вы можете изменить. Подробнее см. документ «ViPNet SIES Unit. Руководство администратора».

Протокол взаимодействия ПО защищаемого устройства и ViPNet SIES Unit через ViPNet SIES Unit RESTful API состоит из HTTP-сообщения с методом POST или GET и HTTP-сообщения ответа на запрос. Аутентификация (см. глоссарий, стр. 63) пользователя в запросах не требуется — стороны взаимодействуют:

- при установке ViPNet SIES Unit на защищаемом устройстве — в замкнутой программной среде;
- при установке ViPNet SIES Unit на отдельной аппаратной платформе — в единой контролируемой зоне.

Для передачи входных и выходных параметров в HTTP-сообщениях запросов и ответов используются JSON-структуры (см. глоссарий, стр. 62).

Примечание. В 32-разрядных версиях ViPNet SIES Unit гарантируется корректная криптографическая обработка JSON-структур с данными объемом не более 150 Мбайт.

Синтаксис HTTP-сообщения запроса

```
{POST | GET} http://<адрес>:<порт>/api/v2/<идентификатор операции>
[Connection: keep-alive]
[Content-Type: application/json]
(пустая строка)
[<JSON-структура>]
```

Описание заголовков и параметров HTTP-сообщения запроса

- Метод HTTP (`POST` или `GET`) зависит от вызываемого метода ViPNet SIES Unit RESTful API. Используется `POST`, если в описании метода ViPNet SIES Unit RESTful API не указано, что используется `GET`.
- `<адрес>` — 127.0.0.1 для локального подключения к ViPNet SIES Unit или IP-адрес аппаратной платформы с установленным ViPNet SIES Unit.
- `<порт>` — номер сетевого порта ViPNet SIES Unit RESTful API, заданный в настройках ViPNet SIES Unit. По умолчанию 9876.
- `<идентификатор операции>` — идентификатор операции ViPNet SIES Unit (на стр. 17), выполнение которой запрашивается через ViPNet SIES Unit RESTful API.
- `Connection` — необязательный заголовок. Позволяет поддерживать постоянное HTTP-соединение с ViPNet SIES Unit RESTful API. При добавлении заголовка в HTTP-запрос использование лексемы `keep-alive` обязательно.
- `<JSON-структура>` — данные защищаемого устройства в JSON-формате, структура которых зависит от запрашиваемой прикладной функции ViPNet SIES Unit. Может присутствовать в HTTP-сообщении только совместно с заголовком `Content-Type`.

Синтаксис HTTP-сообщения ответа

```
HTTP/1.1 <HTTP-код>
Server: <наименование и версия HTTP-сервера>
Date: <дата выработки ответа>
Content-Type: application/json
Content-Length: <размер данных>
Connection: [keep-alive | close]
(пустая строка)
[<JSON-структура>]
```

Описание заголовков и параметров HTTP-сообщения ответа

- `<HTTP-код>` — код состояния протокола HTTP, см. RFC 7231. Может принимать одно из следующих значений:
  - `200` — параметр `<JSON-структура>` содержит результат выполнения прикладной функции ViPNet SIES Unit;
  - не равен `200` — возникла ошибка в работе ViPNet SIES Unit RESTful API.

    При возникновении ошибки обработки HTTP-сообщения запроса формируется HTTP-сообщение ответа с указанием кода ошибки (`4xx` или `5xx`), кратким и полным описанием в кодировке по стандарту UTF-8, см. RFC 3629 и ISO/IEC 10646 Annex D.

    В протоколе обмена HTTP-сообщениями приняты следующие коды ошибок стороны клиента:

    - `400` — ошибка синтаксиса HTTP-сообщения, указанный URI (Uniform Resource Identifier) не поддерживается;
    - `503` — запрос не может быть обработан, ViPNet SIES Unit готовится к выключению.

- `Server` — стандартный обязательный заголовок, содержащий список названий и версий HTTP-сервера ViPNet SIES Unit и компонентов HTTP-сервера ViPNet SIES Unit с комментариями.
- `Date` — стандартный обязательный заголовок, содержащий дату и время выработки HTTP-сообщения ответа (формат по RFC 7231, подмножество формата по RFC 5322).
- `Content-Type` — стандартный обязательный заголовок, содержащий характеристики содержимого HTTP-сообщения.
- `Content-Length` — обязательный заголовок. В параметре `<размер данных>` содержит длину в байтах возвращаемой JSON-структуры.
- `Connection` — стандартный обязательный заголовок, содержащий лексему соединения `close` или `keep-alive`.

  Лексема соединения `close` означает, что данный запрос — последний в текущем соединении с ViPNet SIES Unit RESTful API и соединение закрыто.

  Лексема соединения `keep-alive` подтверждает возможность повторного использования HTTP-соединения с ViPNet SIES Unit RESTful API (использование одного TCP-соединения для отправки и получения многократных HTTP-запросов и ответов вместо открытия нового соединения для каждой пары запрос-ответ).
- `<JSON-структура>` — ответ ViPNet SIES Unit RESTful API с результатом обработки запроса прикладной функции ViPNet SIES Unit или с сообщением об ошибке. Коды и соответствующие им описания ошибок приведены в разделах с описанием методов интерфейса ViPNet SIES Unit RESTful API.

Назначения прикладных связей для методов ViPNet SIES Unit RESTful API

Возможность обмена защищенными данными между защищаемыми устройствами индустриальной системы организуется с помощью прикладных связей (см. глоссарий, стр. 64), заданных между защищаемыми устройствами в ViPNet SIES MC. Подробнее см. документ «ViPNet SIES MC. Руководство администратора».

Перед использованием отдельных методов ViPNet SIES Unit RESTful API с помощью ViPNet SIES MC добавьте прикладную связь между защищаемым устройством с установленным ViPNet SIES Unit и защищаемыми устройствами — адресатами криптографически обработанных данных. Затем синхронизируйте связь на SIES-узлах (см. глоссарий, стр. 62) всех защищаемых устройств — участников связи. Назначения прикладных связей в зависимости от функций приведены в таблице ниже.

Таблица 4. Методы ViPNet SIES Unit RESTful API и требуемые назначения прикладных связей

Функция	Назначение прикладной связи
Получение сведений о ViPNet SIES Unit (на стр. 56)	—
Тест связи (на стр. 57)	—
Обработка защищенного конверта ViPNet SIES MC (на стр. 58)	—
Создание CRISP-сообщения с зашифрованными данными (на стр. 19)	Шифрование в режиме реального времени
Создание фрагмента CRISP-сообщения с зашифрованными данными (на стр. 25)	Шифрование в режиме реального времени
Создание CRISP-сообщения с имитовставкой (на стр. 22)	Вычисление и проверка имитовставки
Создание фрагмента CRISP-сообщения с имитовставкой (на стр. 29)	Вычисление и проверка имитовставки
Обработка CRISP-сообщения с зашифрованными данными (на стр. 20)	Шифрование в режиме реального времени
Обработка фрагмента CRISP-сообщения с зашифрованными данными (на стр. 27)	Шифрование в режиме реального времени
Обработка CRISP-сообщения с имитовставкой (на стр. 23)	Вычисление и проверка имитовставки

Функция	Назначение прикладной связи
Обработка фрагмента CRISP-сообщения с имитовставкой (на стр. 31)	Вычисление и проверка имитовставки
Зашифрование в CMS-контейнер с прикрепленными данными (на стр. 35)	Шифрование и проверка технологической электронной подписи
Расшифрование из CMS-контейнера с прикрепленными данными (на стр. 36)	Шифрование и проверка технологической электронной подписи
Зашифрование в CMS-контейнер с открепленными данными (на стр. 37)	Шифрование и проверка технологической электронной подписи
Расшифрование из CMS-контейнера с открепленными данными (на стр. 39)	Шифрование и проверка технологической электронной подписи
Создание электронной подписи в CMS-контейнере с прикрепленными данными (на стр. 40)	Шифрование и проверка технологической электронной подписи
Проверка электронной подписи из CMS-контейнера с прикрепленными данными (на стр. 42)	Шифрование и проверка технологической электронной подписи
Создание электронной подписи в CMS-контейнере с открепленными данными (на стр. 43)	Шифрование и проверка технологической электронной подписи
Проверка электронной подписи из CMS-контейнера с открепленными данными (на стр. 44)	Шифрование и проверка технологической электронной подписи
Вычисление хэш-кода без указания получателя (на стр. 47)	Вычисление и проверка хэш-суммы
Проверка хэш-кода без указания отправителя (на стр. 48)	Вычисление и проверка хэш-суммы
Вычисление хэш-кода с указанием получателя (на стр. 50)	Вычисление и проверка хэш-суммы
Проверка хэш-кода с указанием отправителя (на стр. 51)	Вычисление и проверка хэш-суммы
Инициирование аутентификации пользователя (на стр. 52)	—
Завершение аутентификации пользователя в формате CMS (на стр. 53)	Шифрование и проверка технологической электронной подписи

Выбор и применение тех или иных функций ViPNet SIES Unit определяется реализуемым вами сценарием защиты информации. Подробнее см. документ «ViPNet SIES. Сценарии работы».

Коды результата выполнения функций

В таблице ниже приведены коды результата выполнения функций, возвращаемые ViPNet SIES Unit.

Таблица 5. Список значений кодов, возвращаемых ViPNet SIES Unit

Символьное представление кода результата	Значение	Описание
`Success`	0	Команда выполнена успешно
`InternalError`	-503	Внутренняя ошибка
`ValidationError`	-504	Ошибка проверки достоверности сообщения
`NotAllowed`	-506	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit
`NodeNotFound`	-508	Указанный получатель или отправитель не найден
`UnknownAlgorithm`	-509	Неизвестный алгоритм
`InvalidStreamId`	-511	Некорректный идентификатор объекта
`ErrorInInputData`	-513	Входные данные имеют некорректный формат
`InputDataNotMcCommand`	-514	Входные данные не содержат команды ViPNet SIES MC
`PermissionDenied`	-515	Отсутствует разрешение на выполнение операции
`InputDataOverflow`	-516	Превышен допустимый размер прикладных данных

Применение ViPNet SIES Unit RESTful API

Чтобы использовать ViPNet SIES Unit RESTful API для взаимодействия с ViPNet SIES Unit:

1. На программно-аппаратной платформе, предназначенной для разработки модуля взаимодействия ПО защищаемого устройства с ViPNet SIES Unit, в любой среде разработки создайте проект программного модуля.
2. Реализуйте выполнение запросов прикладных функций ViPNet SIES Unit через методы ViPNet SIES Unit RESTful API в соответствии с реализуемыми сценариями защиты данных. Подробнее см. документ «ViPNet SIES. Сценарии работы».
3. Если ViPNet SIES Unit удаленно управляется из ViPNet SIES MC, реализуйте обработку защищенного конверта ViPNet SIES MC (на стр. 58).
4. Реализуйте анализ результатов выполнения прикладных функций, полученных от ViPNet SIES Unit RESTful API.
5. Проведите отладку проекта.
6. Установите готовый программный модуль на защищаемое устройство.

2

Методы ViPNet SIES Unit RESTful API

Методы работы с CRISP-сообщениями 18

Методы работы с CMS-контейнерами 34

Другие криптографические методы 47

Информационно-управляющие методы 56

Методы работы с CRISP-сообщениями

ViPNet SIES Unit позволяет организовать обмен защищенными данными между защищаемыми устройствами на прикладном уровне по протоколу CRISP (см. глоссарий, стр. 61) с помощью CRISP-сообщений (см. глоссарий, стр. 61).

ViPNet SIES Unit ведет список порядковых номеров успешно обработанных входящих CRISP-сообщений. Верхней границей списка служит максимальный номер принятого CRISP-сообщения. Нижняя граница списка равна максимальному номеру принятого CRISP-сообщения, уменьшенному на 31.

Входящие CRISP-сообщения могут обрабатываться получателем не в порядке создания, а в порядке поступления. В этом случае не могут быть обработаны устаревшие или повторные сообщения. Соответствующий код результата `-504` возвращается, если любая из следующих проверок завершается положительно:

- номер обрабатываемого сообщения меньше нижней границы списка (устаревшее сообщение);
- номер обрабатываемого сообщения уже входит в список (повторное сообщение).

После успешной обработки входящего сообщения список принятых CRISP-сообщений обновляется с учетом номера обработанного CRISP-сообщения:

- номер лежит в границах списка — обработанное CRISP-сообщение помечается как принятое;
- номер лежит за верхней границей списка — обработанное CRISP-сообщение помечается как принятое, номер становится новой верхней границей и весь список «сдвигается» в сторону новой верхней границы.

При вызове методов данного раздела требуется указывать адресную информацию об отправителе или получателе криптографически обработанных данных. В зависимости от реализуемого сценария защиты данных в качестве адресной информации об отправителе или получателе может выступать:

- адрес защищаемого устройства (см. глоссарий, стр. 63), с которым ведется двустороннее взаимодействие (парная прикладная связь);
- мультивещательный адрес (см. глоссарий, стр. 63) множества защищаемых устройств в случае массовой рассылки (мультивещательная прикладная связь);
- адрес резервированного защищаемого устройства, с которым ведется двустороннее взаимодействие по единственному адресу;
- любой из адресов резервированного защищаемого устройства, с которым ведется двустороннее взаимодействие по множеству адресов.

Создание CRISP-сообщения с зашифрованными данными

Метод создает CRISP-сообщение с зашифрованными данными для указанного получателя. После шифрования данные защищаются имитовставкой (см. глоссарий, стр. 63).

Режимы работы ViPNet SIES Unit

«Штатный».

Идентификатор операции

```
crisp-encrypt
```

JSON-структура HTTP-сообщения запроса

```
{
"deviceId": "<строка>",
"data": "<строка (base64)>"
}
```

Таблица 6. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`deviceId`	обязательный	Адресная информация получателя CRISP-сообщения	строка
`data`	обязательный	Данные для создания CRISP-сообщения	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа при успешной обработке запроса

```
{
"errorCode": 0,
"crisp": "<строка>"
}
```

Таблица 7. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`errorCode`	обязательный	Код результата выполнения функции (`0` — успешно)	целое число
`crisp`	обязательный	Созданное CRISP-сообщение с зашифрованными данными	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа в случае возникновения ошибки при обработке запроса

```
{
"errorCode": <целое число>
}
```

Таблица 8. Возможные ошибки

Значение ключа `errorCode`	Описание
`-513`	Входные данные имеют некорректный формат
`-503`	Внутренняя ошибка
`-508`	Указанный получатель не найден. Выдается, если прикладная связь не синхронизирована на данном SIES-узле с помощью ViPNet SIES MC
`-506`	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit
`-516`	Превышен допустимый размер прикладных данных
`-515`	Отсутствует разрешение на выполнение операции

Обработка CRISP-сообщения с зашифрованными данными

Метод обрабатывает CRISP-сообщение с зашифрованными данными, полученное от отправителя с указанным адресом. Целостность зашифрованного сообщения контролируется проверкой имитовставки.

Режимы работы ViPNet SIES Unit

«Штатный».

Идентификатор операции

```
crisp-decrypt
```

JSON-структура HTTP-сообщения запроса

```
{
"deviceId": "<строка>",
"crisp": "<строка (base64)>"
}
```

Таблица 9. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`deviceId`	обязательный	Адресная информация отправителя	строка

Ключ	Вид	Описание	Тип данных
		CRISP-сообщения	
crisp	обязательный	CRISP-сообщение с зашифрованными данными для обработки	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа при успешной обработке запроса

```
{
"errorCode": 0,
"decryptedData": "<строка (base64)>"
}
```

Таблица 10. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
errorCode	обязательный	Код результата успешного выполнения функции (0 — успешно)	целое число
decryptedData	обязательный	Расшифрованные данные	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа в случае возникновения ошибки при обработке запроса

```
{
"errorCode": <целое число>
}
```

Таблица 11. Возможные ошибки

Значение ключа errorCode	Описание
-513	Входные данные имеют некорректный формат
-503	Внутренняя ошибка
-504	Ошибка проверки достоверности сообщения
-508	Указанный отправитель не найден. Выдается, если прикладная связь не синхронизирована на данном SIES-узле с помощью ViPNet SIES MC
-506	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit
-516	Превышен допустимый размер прикладных данных
-515	Отсутствует разрешение на выполнение операции

Создание CRISP-сообщения с имитовставкой

Метод создает CRISP-сообщение с имитовставкой для указанного получателя. Защищаемые данные не входят в CRISP-сообщение и передаются отдельно.

Режимы работы ViPNet SIES Unit

«Штатный».

Идентификатор операции

```
crisp-imit
```

JSON-структура HTTP-сообщения запроса

```
{
"deviceId": "<строка>",
"data": "<строка (base64)>"
}
```

Таблица 12. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`deviceId`	обязательный	Адресная информация получателя CRISP-сообщения	строка
`data`	обязательный	Данные для создания CRISP-сообщения	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа при успешной обработке запроса

```
{
"errorCode": 0,
"crisp": "<строка>"
}
```

Таблица 13. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`errorCode`	обязательный	Код результата выполнения функции (`0` — успешно)	целое число
`crisp`	обязательный	Созданное CRISP-сообщение с имитовставкой	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа в случае возникновения ошибки при обработке запроса

```
{
```

```
"errorCode": <целое число>
}
```

Таблица 14. Возможные ошибки

Значение ключа `errorCode`	Описание
`-513`	Входные данные имеют некорректный формат
`-503`	Внутренняя ошибка
`-508`	Указанный получатель не найден. Выдается, если прикладная связь не синхронизирована на данном SIES-узле с помощью ViPNet SIES MC
`-506`	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit
`-516`	Превышен допустимый размер прикладных данных
`-515`	Отсутствует разрешение на выполнение операции

Обработка CRISP-сообщения с имитовставкой

Метод обрабатывает CRISP-сообщение с имитовставкой, полученное от указанного отправителя. Защищаемые данные не входят в CRISP-сообщение и передаются отдельно.

Режимы работы ViPNet SIES Unit

«Штатный».

Идентификатор операции

```
crisp-verify-imit
```

JSON-структура HTTP-сообщения запроса

```
{
"deviceId": "<строка>",
"crisp": "<строка (base64)>",
"data": "<строка (base64)>"
}
```

Таблица 15. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`deviceId`	обязательный	Адресная информация отправителя CRISP-сообщения	строка
`crisp`	обязательный	CRISP-сообщение с имитовставкой для обработки	строка в кодировке Base64

Ключ	Вид	Описание	Тип данных
`data`	обязательный	Данные, полученные от отправителя для проверки имитовставки	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа

```
{
"errorCode": <целое число>
}
```

Таблица 16. Возможные ошибки

Значение ключа `errorCode`	Описание
`0`	Имитовставка верна
`-513`	Входные данные имеют некорректный формат
`-503`	Внутренняя ошибка
`-504`	Ошибка проверки достоверности сообщения
`-508`	Указанный отправитель не найден. Выдается, если прикладная связь не синхронизирована на данном SIES-узле с помощью ViPNet SIES MC
`-506`	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit
`-516`	Превышен допустимый размер прикладных данных
`-515`	Отсутствует разрешение на выполнение операции

Методы для поточной работы с CRISP-сообщениями

Общий сценарий поточного создания и обработки CRISP-сообщений:

1. Отправитель делит данные для передачи на фрагменты не менее 2049 байт или на фрагменты определенной длины (рекомендуемые значения приведены в описаниях соответствующих методов).
2. Отправитель с помощью методов ViPNet SIES Unit создает фрагменты CRISP-сообщения.
3. Отправитель последовательно соединяет фрагменты в единое CRISP-сообщение.
4. Отправитель передает CRISP-сообщение получателю.
5. Получатель получает CRISP-сообщение.

6 Получатель самостоятельно делит полученное CRISP-сообщение на фрагменты определенной длины.

7 Получатель с помощью методов ViPNet SIES Unit по фрагментам передает CRISP-сообщение на обработку.

8 Получатель последовательно соединяет в результат обработки фрагменты, возвращаемые ViPNet SIES Unit.

Создание фрагмента CRISP-сообщения с зашифрованными данными

Метод создает фрагмент CRISP-сообщения с зашифрованными данными для указанного получателя. Общая длина данных в бинарном виде для поточной обработки должна быть не менее 4098 байт.

Режимы работы ViPNet SIES Unit

«Штатный».

Идентификатор операции

```
crisp-encrypt
```

JSON-структура HTTP-сообщения запроса для создания потока

```
{
"deviceId": "<строка>",
"data": "<строка (base64)>",
"totalDataSize": <целое число>
}
```

Таблица 17. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`deviceId`	обязательный	Адресная информация получателя CRISP-сообщения	строка
`data`	обязательный	Начальный фрагмент данных для создания CRISP-сообщения. Длина — не менее 2049 байт	строка в кодировке Base64
`totalDataSize`	необязательный	Общий размер данных в байтах. Если ключ опускается, поток не создается и выполняется создание нефрагментированного CRISP-сообщения (на стр. 19)	целое число

JSON-структура HTTP-сообщения запроса с очередным фрагментом данных

```
{
"deviceId": "<строка>",
```

```
"data": "<строка (base64)>",
"streamId": <целое число>
}
```

Таблица 18. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`deviceId`	обязательный	Адресная информация получателя CRISP-сообщения	строка
`data`	обязательный	Очередной фрагмент данных для создания CRISP-сообщения с зашифрованными данными. Длина фрагмента — любая, но не менее 2049 байт	строка в кодировке Base64
`streamId`	необязательный	Условный идентификатор потока фрагментированной обработки	целое число

JSON-структура HTTP-сообщения ответа при успешной обработке запроса

```
{
"errorCode": 0,
"crisp": "<строка>",
"streamId": <целое число>
}
```

Таблица 19. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`errorCode`	обязательный	Код результата выполнения функции (`0` — успешно)	целое число
`crisp`	обязательный	Фрагмент создаваемого CRISP-сообщения с зашифрованными данными	строка в кодировке Base64
`streamId`	необязательный	Условный идентификатор потока фрагментированной обработки	целое число

JSON-структура HTTP-сообщения ответа в случае возникновения ошибки при обработке запроса

```
{
"errorCode": <целое число>,
"streamId": <целое число>
}
```

Таблица 20. Возможные ошибки

Значение ключа `errorCode`	Описание
`-513`	Входные данные имеют некорректный формат

Значение ключа `errorCode`	Описание
`-503`	Внутренняя ошибка
`-508`	Указанный получатель не найден. Выдается, если прикладная связь не синхронизирована на данном SIES-узле с помощью ViPNet SIES MC
`-506`	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit
`-511`	Не найден поток фрагментированной обработки, соответствующий указанному идентификатору
`-515`	Отсутствует разрешение на выполнение операции
`-516`	Превышен допустимый размер прикладных данных

Обработка фрагмента CRISP-сообщения с зашифрованными данными

Метод обрабатывает фрагмент CRISP-сообщения с зашифрованными данными, полученного от отправителя с указанным адресом.

Режимы работы ViPNet SIES Unit

«Штатный».

Идентификатор операции

```
crisp-decrypt
```

JSON-структура HTTP-сообщения запроса для создания потока

```
{
"deviceId": "<строка>",
"crisp": "<строка (base64)>",
"totalDataSize": <целое число>
}
```

Таблица 21. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`deviceId`	обязательный	Адресная информация отправителя CRISP-сообщения	строка
`crisp`	обязательный	Начальный фрагмент CRISP-сообщения с зашифрованными данными для обработки. Длина — не менее 2049 байт	строка в кодировке Base64
`totalDataSize`	необязательный	Общий размер CRISP-сообщения в байтах	целое число

JSON-структура HTTP-сообщения запроса с очередным фрагментом данных

```
{
"deviceId": "<строка>",
"crisp": "<строка (base64)>",
"streamId": <целое число>
}
```

Таблица 22. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`deviceId`	обязательный	Адресная информация отправителя CRISP-сообщения	строка
`crisp`	обязательный	Очередной фрагмент CRISP-сообщения с зашифрованными данными для обработки. Длина фрагмента — любая, но не менее 2049 байт	строка в кодировке Base64
`streamId`	необязательный	Условный идентификатор потока фрагментированной обработки	целое число

JSON-структура HTTP-сообщения ответа при успешной обработке запроса

```
{
"errorCode": 0,
"decryptedData": "<строка (base64)>",
"streamId": <целое число>
}
```

Таблица 23. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`errorCode`	обязательный	Код результата успешного выполнения функции (`0` — успешно)	целое число
`decryptedData`	обязательный	Расшифрованные данные из фрагмента CRISP-сообщения	строка в кодировке Base64
`streamId`	необязательный	Условный идентификатор потока фрагментированной обработки	целое число

JSON-структура HTTP-сообщения ответа в случае возникновения ошибки при обработке запроса

```
{
"errorCode": <целое число>,
"streamId": <целое число>
}
```

Таблица 24. Возможные ошибки

Значение ключа `errorCode`	Описание
`-513`	Входные данные имеют некорректный формат
`-503`	Внутренняя ошибка
`-504`	Ошибка проверки достоверности сообщения
`-508`	Указанный отправитель не найден. Выдается, если прикладная связь не синхронизирована на данном SIES-узле с помощью ViPNet SIES MC
`-506`	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit
`-516`	Превышен допустимый размер прикладных данных
`-515`	Отсутствует разрешение на выполнение операции
`-511`	Не найден поток фрагментированной обработки, соответствующий указанному идентификатору

Создание фрагмента CRISP-сообщения с имитовставкой

Метод создает фрагмент CRISP-сообщения с имитовставкой для указанного получателя. Защищаемые данные не входят в CRISP-сообщение и передаются отдельно.

Режимы работы ViPNet SIES Unit

«Штатный».

Идентификатор операции

```
crisp-imit
```

JSON-структура HTTP-сообщения запроса для создания потока

```
{
"deviceId": "<строка>",
"data": "<строка (base64)>",
"totalDataSize": <целое число>
}
```

Таблица 25. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`deviceId`	обязательный	Адресная информация получателя CRISP-сообщения	строка
`data`	обязательный	Начальный фрагмент данных для создания CRISP-сообщения. Длина — не менее 2049 байт	строка в кодировке Base64

Ключ	Вид	Описание	Тип данных
totalDataSize	необязательный	Общий размер данных в байтах. Если ключ опускается, поток не создается и выполняется создание нефрагментированного CRISP-сообщения (на стр. 22)	целое число

JSON-структура HTTP-сообщения запроса с очередным фрагментом данных

```
{
"deviceId": "<строка>",
"data": "<строка (base64)>",
"streamId": <целое число>
}
```

Таблица 26. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
deviceId	обязательный	Адресная информация получателя CRISP-сообщения	строка
data	обязательный	Очередной фрагмент данных для создания CRISP-сообщения. Длина фрагмента — любая, но не менее 2049 байт	строка в кодировке Base64
streamId	необязательный	Условный идентификатор потока фрагментированной обработки	целое число

JSON-структура HTTP-сообщения ответа при успешной обработке запроса

```
{
"errorCode": 0,
"crisp": "<строка>",
"streamId": <целое число>
}
```

Таблица 27. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
errorCode	обязательный	Код результата выполнения функции (0 — успешно)	целое число
crisp	обязательный	Фрагмент создаваемого CRISP-сообщения с имитовставкой	строка в кодировке Base64
streamId	необязательный	Условный идентификатор потока фрагментированной обработки	целое число

JSON-структура HTTP-сообщения ответа в случае возникновения ошибки при обработке запроса

```
{
```

```
"errorCode": <целое число>,
"streamId": <целое число>
}
```

Таблица 28. Возможные ошибки

Значение ключа `errorCode`	Описание
`-513`	Входные данные имеют некорректный формат
`-503`	Внутренняя ошибка
`-508`	Указанный получатель не найден. Выдается, если прикладная связь не синхронизирована на данном SIES-узле с помощью ViPNet SIES MC
`-506`	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit
`-516`	Превышен допустимый размер прикладных данных
`-515`	Отсутствует разрешение на выполнение операции
`-511`	Не найден поток фрагментированной обработки, соответствующий указанному идентификатору

Обработка фрагмента CRISP-сообщения с имитовставкой

Метод обрабатывает фрагмент CRISP-сообщения с имитовставкой, полученного от указанного отправителя. Защищаемые данные не входят в CRISP-сообщение и передаются отдельно.

Режимы работы ViPNet SIES Unit

«Штатный».

Идентификатор операции

```
crisp-verify-imit
```

JSON-структура HTTP-сообщения запроса для создания потока

```
{
"deviceId": "<строка>",
"crisp": "<строка (base64)>",
"data": "<строка (base64)>",
"totalDataSize": <целое число>
}
```

Таблица 29. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`deviceId`	обязательный	Адресная информация отправителя CRISP-сообщения	строка

Ключ	Вид	Описание	Тип данных
`crisp`	обязательный	CRISP-сообщение целиком	строка в кодировке Base64
`data`	обязательный	Начальный фрагмент данных, полученных от отправителя для проверки имитовставки. Длина — не менее 2049 байт	строка в кодировке Base64
`totalDataSize`	необязательный	Общий размер в байтах данных, полученных от отправителя для проверки имитовставки	целое число

JSON-структура HTTP-сообщения запроса с очередным фрагментом данных

```
{
"deviceId": "<строка>",
"data": "<строка (base64)>",
"streamId": <целое число>
}
```

Таблица 30. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`deviceId`	обязательный	Адресная информация отправителя CRISP-сообщения	строка
`data`	обязательный	Очередной фрагмент данных, полученных от отправителя для проверки имитовставки. Длина фрагмента — любая, но не менее 2049 байт	строка в кодировке Base64
`streamId`	необязательный	Условный идентификатор потока фрагментированной обработки	целое число

JSON-структура HTTP-сообщения ответа

```
{
"errorCode": <целое число>,
"streamId": <целое число>
}
```

Таблица 31. Возможные ошибки

Значение ключа `errorCode`	Описание
`0`	Имитовставка верна
`-513`	Входные данные имеют некорректный формат
`-503`	Внутренняя ошибка
`-504`	Ошибка проверки достоверности сообщения

Значение ключа `errorCode`	Описание
`-508`	Указанный отправитель не найден. Выдается, если прикладная связь не синхронизирована на данном SIES-узле с помощью ViPNet SIES MC
`-506`	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit
`-516`	Превышен допустимый размер прикладных данных
`-515`	Отсутствует разрешение на выполнение операции
`-511`	Не найден поток фрагментированной обработки, соответствующий указанному идентификатору

Методы работы с CMS-контейнерами

Методы предназначены для обработки криптографических сообщений в формате CMS-контейнеров (см. глоссарий, стр. 61).

CMS-контейнер содержит сведения для корректного восстановления или проверки данных, например:

- информацию о примененном алгоритме хэширования (см. глоссарий, стр. 64);
- информацию о примененном алгоритме электронной подписи (см. глоссарий, стр. 64);
- дату и время формирования электронной подписи;
- серийный номер сертификата ключа проверки электронной подписи (см. глоссарий, стр. 64).

ViPNet SIES Unit поддерживает обработку CMS-контейнеров двух типов:

1. данные с электронной подписью (signed data) открепленной (см. глоссарий, стр. 64) или прикрепленной (см. глоссарий, стр. 64);
2. шифрованные данные (enveloped data) открепленные или прикрепленные.

Синтаксис криптографических сообщений в CMS-контейнерах, обрабатываемых ViPNet SIES Unit, соответствует стандарту RFC 5652. Применение CMS рекомендовано Техническим комитетом Росстандарта (ТК 026: «Криптографическая защита информации»).

Структуры данных в CMS-контейнерах кодируются по правилам BER (Basic Encoding Rules).

При вызове методов данного раздела требуется указывать адресную информацию об отправителе или получателе криптографически обработанных данных. В зависимости от реализуемого сценария защиты данных в качестве адресной информации об отправителе или получателе может выступать:

- адрес защищаемого устройства, с которым ведется двустороннее взаимодействие (парная прикладная связь);
- адрес резервированного защищаемого устройства, с которым ведется двустороннее взаимодействие по единственному адресу;
- любой из адресов резервированного защищаемого устройства, с которым ведется двустороннее взаимодействие по множеству адресов.

Зашифрование в CMS-контейнер с прикрепленными данными

Метод зашифровывает данные для указанного получателя и возвращает сформированный CMS-контейнер типа Enveloped Data с зашифрованными данными.

Режимы работы ViPNet SIES Unit

«Штатный».

Идентификатор операции

```
pki-encrypt-attached
```

JSON-структура HTTP-сообщения запроса

```
{
"deviceId": "<строка>",
"data": "<строка (base64)>"
}
```

Таблица 32. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`deviceId`	обязательный	Адресная информация получателя зашифрованных данных	строка
`data`	обязательный	Данные для зашифрования	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа при успешной обработке запроса

```
{
"errorCode": 0,
"cms": "<строка (base64)>"
}
```

Таблица 33. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`errorCode`	обязательный	Код результата успешного выполнения функции (`0` — успешно)	целое число
`cms`	обязательный	Созданный CMS-контейнер с зашифрованными данными	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа в случае возникновения ошибки при обработке запроса

```
{
"errorCode": <целое число>
}
```

Таблица 34. Возможные ошибки

Значение ключа `errorCode`	Описание
`-513`	Входные данные имеют некорректный формат
`-516`	Превышен допустимый размер прикладных данных
`-508`	Указанный получатель не найден. Выдается в отсутствие действительного прикладного сертификата SIES-узла (если прикладная связь не синхронизирована на данном SIES-узле с помощью ViPNet SIES MC)
`-515`	Отсутствует разрешение на выполнение операции
`-503`	Внутренняя ошибка
`-506`	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit

Расшифрование из CMS-контейнера с прикрепленными данными

Метод расшифровывает данные из CMS-контейнера типа Enveloped Data.

Режимы работы ViPNet SIES Unit

«Штатный».

Идентификатор операции

```
pki-decrypt-attached
```

JSON-структура HTTP-сообщения запроса

```
{
"cms": "<строка (base64)>"
}
```

Таблица 35. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`cms`	обязательный	CMS-контейнер с зашифрованными данными	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа при успешной обработке запроса

```
{
"errorCode": 0,
"decryptedData": "<строка (base64)>"
}
```

Таблица 36. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`errorCode`	обязательный	Код результата успешного выполнения функции (`0` — успешно)	целое число
`decryptedData`	обязательный	Расшифрованные данные	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа в случае возникновения ошибки при обработке запроса

```
{
"errorCode": <целое число>
}
```

Таблица 37. Возможные ошибки

Значение ключа `errorCode`	Описание
`-513`	Входные данные имеют некорректный формат
`-516`	Превышен допустимый размер прикладных данных
`-508`	Получатель не найден. Выдается, если CMS-контейнер не предназначен для ViPNet SIES Unit или прикладная связь не синхронизирована на ViPNet SIES Unit с помощью ViPNet SIES MC
`-503`	Внутренняя ошибка
`-506`	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit

Зашифрование в CMS-контейнер с открепленными данными

Метод зашифровывает данные для указанного получателя и возвращает сформированный CMS-контейнер типа Enveloped Data и зашифрованные данные.

Режимы работы ViPNet SIES Unit

«Штатный».

Идентификатор операции

```
pki-encrypt-detached
```

JSON-структура HTTP-сообщения запроса

```
{
"deviceId": "<строка>",
"data": "<строка (base64)>"
}
```

Таблица 38. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`deviceId`	обязательный	Адресная информация получателя зашифрованных данных	строка
`data`	обязательный	Данные для зашифрования	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа при успешной обработке запроса

```
{
"errorCode": 0,
"cms": "<строка (base64)>",
"encryptedData": "<строка (base64)>"
}
```

Таблица 39. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`errorCode`	обязательный	Код результата успешного выполнения функции (`0` — успешно)	целое число
`cms`	обязательный	Созданный CMS-контейнер	строка в кодировке Base64
`encryptedData`	обязательный	Зашифрованные данные	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа в случае возникновения ошибки при обработке запроса

```
{
"errorCode": <целое число>
}
```

Таблица 40. Возможные ошибки

Значение ключа `errorCode`	Описание
`-513`	Входные данные имеют некорректный формат
`-516`	Превышен допустимый размер прикладных данных
`-508`	Указанный получатель не найден. Выдается в отсутствие действительного прикладного сертификата SIES-узла (если прикладная связь не синхронизирована на данном SIES-узле с помощью ViPNet SIES MC)
`-515`	Отсутствует разрешение на выполнение операции
`-503`	Внутренняя ошибка
`-506`	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit

Расшифрование из CMS-контейнера с открепленными данными

Метод расшифровывает данные с помощью содержимого CMS-контейнера типа Enveloped Data.

Режимы работы ViPNet SIES Unit

«Штатный».

Идентификатор операции

`pki-decrypt-detached`

JSON-структура HTTP-сообщения запроса

```
{
"cms": "<строка (base64)>",
"encryptedData": "<строка (base64)>"
}
```

Таблица 41. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`cms`	обязательный	CMS-контейнер	строка в кодировке Base64
`encryptedData`	обязательный	Зашифрованные данные	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа при успешной обработке запроса

```
{
"errorCode": 0,
"decryptedData": "<строка (base64)>"
}
```

Таблица 42. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`errorCode`	обязательный	Код результата успешного выполнения функции (`0` — успешно)	целое число
`decryptedData`	обязательный	Расшифрованные данные	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа в случае возникновения ошибки при обработке запроса

```
{
"errorCode": <целое число>
}
```

Таблица 43. Возможные ошибки

Значение ключа `errorCode`	Описание
`-513`	Входные данные имеют некорректный формат
`-516`	Превышен допустимый размер прикладных данных
`-508`	Получатель не найден. Выдается, если CMS-контейнер не предназначен для ViPNet SIES Unit или прикладная связь не синхронизирована на ViPNet SIES Unit с помощью ViPNet SIES MC
`-503`	Внутренняя ошибка
`-506`	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit

Создание электронной подписи в CMS-контейнере с прикрепленными данными

Метод формирует CMS-контейнер типа Signed Data с электронной подписью и подписанными данными.

Режимы работы ViPNet SIES Unit

«Штатный».

Идентификатор операции

```
pki-sign-attached
```

JSON-структура HTTP-сообщения запроса

```
{
"data": "<строка (base64)>"
}
```

Таблица 44. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`data`	обязательный	Данные для заверения электронной подписью	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа при успешной обработке запроса

```
{
"errorCode": 0,
"cms": "<строка (base64)>"
}
```

Таблица 45. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`errorCode`	обязательный	Код результата успешного выполнения функции (`0` — успешно)	целое число
`cms`	обязательный	CMS-контейнер с электронной подписью и подписанными данными	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа в случае возникновения ошибки при обработке запроса

```
{
"errorCode": <целое число>
}
```

Таблица 46. Возможные ошибки

Значение ключа `errorCode`	Описание
`-513`	Входные данные имеют некорректный формат
`-516`	Превышен допустимый размер прикладных данных
`-515`	Отсутствует разрешение на выполнение операции
`-503`	Внутренняя ошибка
`-506`	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit

Проверка электронной подписи из CMS-контейнера с прикрепленными данными

Метод проверяет электронную подпись из CMS-контейнера типа Signed Data для данных.

Значение параметра `deviceId` можно использовать для проверки соответствия идентификатора отправителя данных ожидаемому значению.

Режимы работы ViPNet SIES Unit

«Штатный».

Идентификатор операции

```
pki-verify-signature-attached
```

JSON-структура HTTP-сообщения запроса

```
{
"cms": "<строка (base64)>"
}
```

Таблица 47. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`cms`	обязательный	CMS-контейнер с электронной подписью и подписанными данными	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа при успешной обработке запроса

```
{
"errorCode": 0,
"deviceId": "<строка>",
"data": "<строка (base64)>"
}
```

Таблица 48. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`errorCode`	обязательный	Код результата выполнения функции (`0` — электронная подпись верна)	целое число
`deviceId`	обязательный	Адресная информация защищаемого устройства, сформировавшего электронную подпись	строка
`data`	обязательный	Подписанные данные отправителя	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа в случае возникновения ошибки при обработке запроса

```
{
"errorCode": <целое число>
}
```

Таблица 49. Возможные ошибки

Значение ключа `errorCode`	Описание
`-513`	Входные данные имеют некорректный формат
`-516`	Превышен допустимый размер прикладных данных
`-508`	Отправитель не найден. Выдается в отсутствие действительного прикладного сертификата SIES-узла (если прикладная связь не синхронизирована на данном SIES-узле с помощью ViPNet SIES MC)
`-515`	Отсутствует разрешение на выполнение операции
`-504`	Ошибка проверки достоверности сообщения
`-503`	Внутренняя ошибка

Создание электронной подписи в CMS-контейнере с открепленными данными

Метод создает CMS-контейнер типа Signed Data с электронной подписью без данных.

Режимы работы ViPNet SIES Unit

«Штатный».

Идентификатор операции

```
pki-sign-detached
```

JSON-структура HTTP-сообщения запроса

```
{
"data": "<строка (base64)>"
}
```

Таблица 50. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`data`	обязательный	Данные для заверения электронной подписью	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа при успешной обработке запроса

```
{
"errorCode": 0,
"cms": "<строка (base64)>"
}
```

Таблица 51. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`errorCode`	обязательный	Код результата успешного выполнения функции (`0` — успешно)	целое число
`cms`	обязательный	CMS-контейнер с электронной подписью без подписанных данных	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа в случае возникновения ошибки при обработке запроса

```
{
"errorCode": <целое число>
}
```

Таблица 52. Возможные ошибки

Значение ключа `errorCode`	Описание
`-513`	Входные данные имеют некорректный формат
`-516`	Превышен допустимый размер прикладных данных
`-515`	Отсутствует разрешение на выполнение операции
`-503`	Внутренняя ошибка
`-506`	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit

Проверка электронной подписи из CMS-контейнера с открепленными данными

Метод проверяет электронную подпись из CMS-контейнера типа Signed Data для данных.

Значение параметра `deviceId` можно использовать для проверки соответствия идентификатора отправителя данных ожидаемому значению.

Режимы работы ViPNet SIES Unit

«Штатный».

Идентификатор операции

```
pki-verify-signature-detached
```

JSON-структура HTTP-сообщения запроса

```
{
"cms": "<строка (base64)>",
"data": "<строка (base64)>"
}
```

Таблица 53. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`cms`	обязательный	CMS-контейнер без данных	строка в кодировке Base64
`data`	обязательный	Подписанные данные для проверки электронной подписи в CMS-контейнере	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа при успешной обработке запроса

```
{
"errorCode": 0,
"deviceId": "<строка>"
}
```

Таблица 54. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`errorCode`	обязательный	Код результата выполнения функции (`0` — электронная подпись верна)	целое число
`deviceId`	обязательный	Адресная информация защищаемого устройства, сформировавшего электронную подпись	строка

JSON-структура HTTP-сообщения ответа в случае возникновения ошибки при обработке запроса

```
{
"errorCode": <целое число>
}
```

Таблица 55. Возможные ошибки

Значение ключа `errorCode`	Описание
`-513`	Входные данные имеют некорректный формат
`-516`	Превышен допустимый размер прикладных данных

Значение ключа `errorCode`	Описание
`-508`	Отправитель не найден. Выдается в отсутствие действительного прикладного сертификата SIES-узла (если прикладная связь не синхронизирована на данном SIES-узле с помощью ViPNet SIES MC)
`-515`	Отсутствует разрешение на выполнение операции
`-504`	Ошибка проверки достоверности сообщения
`-503`	Внутренняя ошибка
`-506`	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit

Другие криптографические методы

При вызове отдельных методов данного раздела требуется указывать адресную информацию об отправителе или получателе криптографически обработанных данных. В зависимости от реализуемого сценария защиты данных в качестве адресной информации об отправителе или получателе может выступать:

- адрес защищаемого устройства, с которым ведется двустороннее взаимодействие (парная прикладная связь);
- адрес резервированного защищаемого устройства, с которым ведется двустороннее взаимодействие по единственному адресу;
- любой из адресов резервированного защищаемого устройства, с которым ведется двустороннее взаимодействие по множеству адресов.

Вычисление хэш-кода без указания получателя

Метод вычисляет хэш-код для данных по алгоритму ГОСТ Р 34.11-2012 с длиной хэш-кода 256 бит. Значение выходного параметра `hashCtx` позволяет проверить хэш-код на SIES-узле любого типа вне зависимости от наличия и назначения прикладных связей.

Режимы работы ViPNet SIES Unit

«Штатный».

Идентификатор операции

```
hash
```

JSON-структура HTTP-сообщения запроса

```
{
"data": "<строка (base64)>"
}
```

Таблица 56. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`data`	обязательный	Данные для хэширования	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа при успешной обработке запроса

```
{
"errorCode": 0,
```

```
"hash": "<строка (base64)>",
"hashCtx": "<строка (base64)>"
}
```

Таблица 57. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`errorCode`	обязательный	Код результата успешного выполнения функции (`0` — успешно)	целое число
`hash`	обязательный	Хэш-код для данных	строка в кодировке Base64
`hashCtx`	обязательный	Контекст хэширования для проверки хэш-кода	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа в случае возникновения ошибки при обработке запроса

```
{
"errorCode": <целое число>
}
```

Таблица 58. Возможные ошибки

Значение ключа `errorCode`	Описание
`-513`	Входные данные имеют некорректный формат
`-516`	Превышен допустимый размер прикладных данных
`-503`	Внутренняя ошибка
`-506`	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit

Проверка хэш-кода без указания отправителя

Метод проверяет соответствие хэш-кода данным. Алгоритм хэширования выбирается по следующим правилам:

- задан контекст хэширования — применяется алгоритм, заданный в контексте хэширования;
- не задан контекст хэширования — применяется алгоритм ГОСТ Р 34.11-2012 с длиной хэш-кода 256 бит.

Режимы работы ViPNet SIES Unit

«Штатный».

Идентификатор операции

```
hash-verify
```

JSON-структура HTTP-сообщения запроса

```
{
"hashCtx": "<строка (base64)>",
"data": "<строка (base64)>",
"hash": "<строка (base64)>"
}
```

Таблица 59. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`hashCtx`	необязательный	Контекст хэширования для проверки хэш-кода	строка в кодировке Base64
`data`	обязательный	Данные для проверки хэш-кода	строка в кодировке Base64
`hash`	обязательный	Хэш-код для данных	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа

```
{
"errorCode": <целое число>
}
```

Таблица 60. Возможные ошибки

Значение ключа `errorCode`	Описание
`0`	Хэш-код верный
`-513`	Входные данные имеют некорректный формат
`-516`	Превышен допустимый размер прикладных данных
`-504`	Ошибка проверки достоверности сообщения
`-503`	Внутренняя ошибка
`-506`	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit
`-509`	Неизвестный алгоритм

Вычисление хэш-кода с указанием получателя

Метод вычисляет хэш-код для данных по алгоритму, заданному в настройках прикладной связи с получателем. Значение выходного параметра `hashCtx` позволяет проверить хэш-код на SIES-узле любого типа вне зависимости от наличия и назначения прикладных связей.

Режимы работы ViPNet SIES Unit

«Штатный».

Идентификатор операции

```
hash-address
```

JSON-структура HTTP-сообщения запроса

```
{
"deviceId": "<строка>",
"data": "<строка (base64)>"
}
```

Таблица 61. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`deviceId`	обязательный	Адресная информация получателя хэшируемых данных	строка
`data`	обязательный	Данные для хэширования	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа при успешной обработке запроса

```
{
"errorCode": 0,
"hash": "<строка (base64)>",
"hashCtx": "<строка (base64)>"
}
```

Таблица 62. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`errorCode`	обязательный	Код результата успешного выполнения функции (`0` — успешно)	целое число
`hash`	обязательный	Хэш-код для данных защищаемого устройства	строка в кодировке Base64
`hashCtx`	обязательный	Контекст хэширования для проверки хэш-кода	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа в случае возникновения ошибки при обработке запроса

```
{
"errorCode": <целое число>
}
```

Таблица 63. Возможные ошибки

Значение ключа `errorCode`	Описание
`-513`	Входные данные имеют некорректный формат
`-516`	Превышен допустимый размер прикладных данных
`-508`	Указанный получатель не найден. Выдается в отсутствие действительного прикладного сертификата SIES-узла (если прикладная связь не синхронизирована на данном SIES-узле с помощью ViPNet SIES MC)
`-515`	Отсутствует разрешение на выполнение операции
`-509`	Неизвестный алгоритм
`-503`	Внутренняя ошибка
`-506`	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit

Проверка хэш-кода с указанием отправителя

Метод проверяет соответствие хэш-кода данным. Применяется алгоритм, заданный в настройках прикладной связи.

Режимы работы ViPNet SIES Unit

«Штатный».

Идентификатор операции

```
hash-verify-address
```

JSON-структура HTTP-сообщения запроса

```
{
"deviceId": "<строка>",
"data": "<строка (base64)>",
"hash": "<строка (base64)>"
}
```

Таблица 64. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`deviceId`	обязательный	Адресная информация отправителя хэшированных	строка

Ключ	Вид	Описание	Тип данных
		данных	
data	обязательный	Данные для проверки хэш-кода	строка в кодировке Base64
hash	обязательный	Хэш-код для данных	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа

```
{
"errorCode": <целое число>
}
```

Таблица 65. Возможные ошибки

Значение ключа errorCode	Описание
0	Хэш-код верный
-513	Входные данные имеют некорректный формат
-516	Превышен допустимый размер прикладных данных
-508	Указанный отправитель не найден. Выдается, если прикладная связь не синхронизирована на данном SIES-узле с помощью ViPNet SIES MC
-515	Отсутствует разрешение на выполнение операции
-504	Ошибка проверки достоверности сообщения
-503	Внутренняя ошибка
-506	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit
-509	Неизвестный алгоритм

Инициирование аутентификации пользователя

Метод инициирует процесс аутентификации (см. глоссарий, стр. 63) пользователя защищаемого устройства. В методе используется HTTP-сообщение запроса типа GET.

Режимы работы ViPNet SIES Unit

«Штатный».

Идентификатор операции

```
start-auth
```

JSON-структура HTTP-сообщения запроса

Нет.

JSON-структура HTTP-сообщения ответа при успешной обработке запроса

```
{
"errorCode": 0,
"authData": "<строка (base64)>"
}
```

Таблица 66. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`errorCode`	обязательный	Код результата выполнения функции (`0` — успешно)	целое число
`authData`	обязательный	Случайная последовательность байт, используемая для аутентификации	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа в случае возникновения ошибки при обработке запроса

```
{
"errorCode": <целое число>
}
```

Таблица 67. Возможные ошибки

Значение ключа `errorCode`	Описание
`-503`	Внутренняя ошибка
`-506`	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit

Завершение аутентификации пользователя в формате CMS

Метод завершает процесс аутентификации пользователя защищаемого устройства в формате CMS без данных пользователя с электронной подписью по ГОСТ Р 34.10–2012 с длиной ключа проверки подписи 512 бит.

Режимы работы ViPNet SIES Unit

«Штатный».

Идентификатор операции

```
finish-auth-cms
```

JSON-структура HTTP-сообщения запроса

```
{
"authData": "<строка (base64)>",
"cms": "<строка (base64)>"
}
```

Таблица 68. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`authData`	обязательный	Случайная последовательность байт, используемая для аутентификации	строка в кодировке Base64
`cms`	обязательный	CMS-контейнер типа Signed Data без данных пользователя с электронной подписью. Электронную подпись нужно сформировать для случайной последовательности байт `authData` из результата выполнения метода `start-auth` по ГОСТ Р 34.10–2012 с длиной ключа проверки подписи 512 бит	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа при успешной обработке запроса

```
{
"errorCode": 0,
"userInfo": "<строка>"
}
```

Таблица 69. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`errorCode`	обязательный	Код результата выполнения функции (`0` — успешно)	целое число
`userInfo`	обязательный	Информация о пользователе защищаемого устройства, проходящем аутентификацию (значение параметра **Адрес защищаемого устройства**, заданное в ViPNet SIES MC)	строка

JSON-структура HTTP-сообщения ответа в случае возникновения ошибки при обработке запроса

```
{
"errorCode": <целое число>
}
```

Таблица 70. Возможные ошибки

Значение ключа `errorCode`	**Описание**
`-513`	Входные данные имеют некорректный формат
`-508`	Указанный пользователь не найден. Выдается, если прикладная связь между ViPNet SIES Core и SIES-узлом типа **Пользователь** не задана или не синхронизирована на данном SIES-узле с помощью ViPNet SIES MC
`-511`	Не найден сеанс аутентификации
`-504`	Ошибка проверки достоверности сообщения
`-503`	Внутренняя ошибка
`-506`	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit
`-516`	Превышен допустимый размер прикладных данных

Информационно-управляющие методы

Получение сведений о ViPNet SIES Unit

Метод возвращает идентификатор и текущий режим работы ViPNet SIES Unit. В методе используется HTTP-сообщение запроса типа GET.

Режимы работы ViPNet SIES Unit

Все, кроме режима «Системная блокировка».

Идентификатор операции

```
sies-info
```

JSON-структура HTTP-сообщения запроса

Отсутствует.

JSON-структура HTTP-сообщения ответа при успешной обработке запроса

```
{
"errorCode": 0,
"node_id": "<строка>",
"node_type": <целое число>,
"version": "<строка>",
"work_mode": <целое число>
}
```

Таблица 71. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`errorCode`	обязательный	Код результата успешного выполнения функции (`0` — успешно)	целое число
`nodeId`	обязательный	Идентификатор ViPNet SIES Unit. До завершения первой инициализации выдается значение `000000000000`	строка
`nodeType`	обязательный	Тип SIES-узла (`5` — ViPNet SIES Unit)	целое число
`version`	обязательный	Версия ПО ViPNet SIES Unit	строка

Ключ	Вид	Описание	Тип данных
`workMode`	обязательный	Режим работы ViPNet SIES Unit: • `3` — «Инициализация», выполняются только запросы на вызов информационно-управляющих методов; • `4` — «Конфигурирование», выполняются только запросы на вызов информационно-управляющих методов. • `7` — «Штатный», выполняются запросы на криптографическую обработку данных защищаемого устройства и на вызов информационно-управляющих методов. • `9` — «Очистка», выполняются только запросы на вызов данного метода и метода Тест связи (на стр. 57). Подробнее см. документ «ViPNet SIES Unit. Руководство администратора».	целое число

JSON-структура HTTP-сообщения ответа в случае возникновения ошибки при обработке запроса

```
{
"errorCode": <целое число>
}
```

Таблица 72. Возможные ошибки

Значение ключа `errorCode`	Описание
`-503`	Внутренняя ошибка

Тест связи

Функция «Эхо». Метод предназначен для проверки доступности ViPNet SIES Unit. Без изменения возвращает JSON-структуру, переданную на вход. В методе используется HTTP-сообщение запроса типа GET.

Режимы работы ViPNet SIES Unit

Все, кроме режима «Системная блокировка».

Идентификатор операции

```
echo
```

JSON-структура HTTP-сообщения запроса

```
{
"data": "<строка>"
}
```

Таблица 73. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`data`	обязательный	Строка с любым текстом	строка

JSON-структура HTTP-сообщения ответа при успешной обработке запроса

```
{
"errorCode": 0,
"data": "<строка>"
}
```

Таблица 74. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
`errorCode`	обязательный	Код результата успешного выполнения функции (`0` — успешно)	целое число
`data`	обязательный	Строка с текстом, переданная на вход	строка

JSON-структура HTTP-сообщения ответа в случае возникновения ошибки при обработке запроса

```
{
"errorCode": <целое число>
}
```

Таблица 75. Возможные ошибки

Значение ключа `errorCode`	Описание
`-503`	Внутренняя ошибка

Обработка защищенного конверта ViPNet SIES MC

Метод выполняет обработку защищенного конверта (см. глоссарий, стр. 63) ViPNet SIES MC, полученного защищаемым устройством. В случае успешного завершения вызова функции значение выходного параметра `answerToMC` следует вернуть ViPNet SIES MC. В случае получения сообщения об ошибке никаких действий не требуется (например, данный факт может фиксироваться в журнале работы защищаемого устройства).

Примечание. Задача идентификации защищенного конверта ViPNet SIES MC при передаче по каналам связи индустриальной системы возлагается на индустриальную систему и защищаемое устройство.

Режимы работы ViPNet SIES Unit

Все, кроме режимов «Системная блокировка» и «Очистка».

Идентификатор операции

```
mccommand
```

JSON-структура HTTP-сообщения запроса

```
{
"command": "<строка (base64)>"
}
```

Таблица 76. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
command	обязательный	Защищенный конверт от ViPNet SIES MC	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа при успешной обработке запроса

```
{
"errorCode": 0,
"answerToMC": "<строка (base64)>"
}
```

Таблица 77. Ключи JSON-структуры

Ключ	Вид	Описание	Тип данных
errorCode	обязательный	Код результата успешного выполнения функции (0 — успешно)	целое число
answerToMC	обязательный	Результат обработки защищенного конверта от ViPNet SIES MC, который следует передать в ViPNet SIES MC	строка в кодировке Base64

JSON-структура HTTP-сообщения ответа в случае возникновения ошибки при обработке запроса

```
{
"errorCode": <целое число>
}
```

Таблица 78. Возможные ошибки

Значение ключа `errorCode`	**Описание**
`-503`	Внутренняя ошибка
`-506`	Выполнение команды запрещено в текущем режиме работы ViPNet SIES Unit
`-513`	Входные данные имеют некорректный формат
`-514`	Входные данные не содержат команды ViPNet SIES MC

A

Глоссарий

CMS (Cryptographic Message Syntax)

Стандарт, описывающий структуру криптографических сообщений (контейнеров), содержащих сведения для их корректного открытия или использования. Защищаемые данные могут передаваться как в составе CMS-сообщений так и отдельно. С помощью CMS-сообщений можно организовать защищенный обмен данными между защищаемыми устройствами — участниками парной или резервированной парной прикладной связи. Создание и обработку CMS-сообщений выполняют SIES-узлы.

CMS-контейнер

Криптографическое сообщение, оформленное по стандарту CMS. Применение CMS-контейнеров в обмене криптографическими сообщениями рекомендовано Техническим комитетом Росстандарта (ТК 026: «Криптографическая защита информации»).

CRISP (Cryptographic Industrial Security Protocol)

Неинтерактивный протокол прикладного уровня для защищенной передачи данных в индустриальных системах. Обеспечивает конфиденциальность и целостность сообщений и защищает от навязывания повторных сообщений. Применяется в соответствии с утвержденными рекомендациями по стандартизации Р 1323565.1.029-2019 «Информационная технология. Криптографическая защита информации. Протокол защищенного обмена для индустриальных систем». Адресация, маршрутизация и доставка сообщения выполняются средствами индустриальной системы.

CRISP-сообщение

Блок данных, защищенный с помощью протокола CRISP. Создание и обработку CRISP-сообщений выполняют SIES-узлы.

IIoT-система

IoT-система отраслевого применения, объединяющая промышленные (производственные) объекты, оснащенные встроенными технологиями для взаимодействия друг с другом или с внешней средой и с возможностью удаленного контроля и управления в автоматическом режиме без участия человека. Строится по принципам индустриального интернета вещей (IIoT).

JSON (JavaScript Object Notation)

Текстовый формат обмена данными, основанный на JavaScript — мультипарадигменном языке программирования. Формат считается независимым от JavaScript, легко читается людьми и может использоваться совместно с любым языком программирования.

RESTful API

REST (Representational State Transfer) — архитектурный стиль взаимодействия компонентов распределенного приложения в сети. Это согласованный набор ограничений (веб-API), учитываемых при проектировании распределенной гипермедиа-системы. Удаленный вызов процедуры — HTTP-запрос (обычно «GET» или «POST» — REST-запрос). Данные передаются в параметрах запроса.

SIES-узел

Компонент комплекса ViPNet SIES, развернутый в индустриальной системе, выполняющий прикладные криптографические операции над данными индустриальной системы и централизованно управляемый с помощью ViPNet SIES MC.

ViPNet SIES (Security for Industrial and Embedded Solutions)

Комплекс продуктов ИнфоТеКСа для защиты информации в индустриальных системах (Патент РФ № 2706176).

ViPNet SIES MC

Система управления SIES-узлами. Отправляет SIES-узлам защищенные команды управления и мониторинга по каналам связи индустриальной системы. Выполняет роль центра управления ключевой системой ViPNet SIES.

ViPNet SIES Unit

Компонент комплекса ViPNet SIES. Программно-аппаратный или программный комплекс, размещенный на операторском уровне индустриальной системы. Интегрируется с устройствами операторского уровня индустриальной системы (SCADA, OPC и другими) и выполняет функции SIES-узла.

Адрес защищаемого устройства

Уникальное обозначение защищаемого устройства в индустриальной системе, с помощью которого другие защищаемые устройства взаимодействуют с данным защищаемым устройством. Используется в парных и резервированных парных прикладных связях.

Аутентификация

Процедура проверки подлинности на основе сравнения данных, предоставленных субъектом и данных, хранящихся на объекте.

Защищаемое устройство

Техническое средство обработки информации индустриальной системы, интегрированное с SIES-узлом.

Защищенный конверт

Команда управления или мониторинга ViPNet SIES MC или ответ SIES-узла, упакованный в криптографический контейнер, защищенный от прочтения и подмены данных.

Имитовставка

Специальный набор символов, добавляемый к сообщению и предназначенный для обеспечения контроля целостности сообщения и аутентификации источника данных сообщения.

Индустриальная система

Комплекс средств, обеспечивающий полный цикл функционирования производства или отдельного технологического процесса в различных областях экономики.

Мультивещательная прикладная связь

Прикладная связь для криптографической обработки данных во взаимодействии между несколькими защищаемыми устройствами.

Мультивещательный адрес

Уникальное обозначение мультивещательной прикладной связи между защищаемыми устройствами, на которое участники связи публикуют и с которого прочитывают защищенные сообщения.

Парная прикладная связь

Прикладная связь для криптографической обработки данных во взаимодействии между двумя защищаемыми устройствами.

Прикладная связь

Набор криптографических параметров и ключевой информации, с помощью которого защищаемое устройство может выполнять криптографические операции над данными. Прикладная связь задается с помощью ViPNet SIES MC и хранится на SIES-узлах взаимодействующих защищаемых устройств.

Резервированная парная прикладная связь

Прикладная связь для криптографической обработки данных во взаимодействии защищаемых устройств в индустриальных системах с применением резервирования.

Сертификат ключа проверки электронной подписи

Электронный документ или документ на бумажном носителе, выданный удостоверяющим центром либо доверенным лицом удостоверяющего центра и подтверждающий принадлежность ключа проверки электронной подписи владельцу сертификата ключа проверки электронной подписи.

Хэширование

Преобразование массива данных произвольной длины в битовую строку фиксированной длины, выполняемое определенным алгоритмом. Результат применения хэш-функции (хэш-код или хэш-сумма) однозначно определяется первоначальным массивом данных. Повторное применение хэш-функции к массиву данных и сравнение повторного хэш-кода с первоначальным хэш-кодом позволяет проверить целостность массива данных.

Целостность

Отсутствие изменений в передаваемой или хранимой информации по сравнению с ее исходной записью.

Электронная подпись (ЭП)

Информация в электронной форме, которая присоединена к другой информации в электронной форме (подписываемой информации) или иным образом связана с такой информацией и которая используется для определения лица, подписывающего информацию.

Электронная подпись открепленная

При использовании открепленной подписи электронная подпись помещается в CMS-контейнер, а заверяемые электронной подписью данные в CMS-контейнер не помещаются. Заверяемые данные передаются адресату отдельно. Чтобы проверить электронную подпись требуется CMS-контейнер с открепленной подписью и данные, заверяемые электронной подписью.

Электронная подпись прикрепленная

При использовании прикрепленной подписи электронная подпись и заверяемые электронной подписью данные совместно помещаются в CMS-контейнер.