КОД безопасности

Программа доверенной визуализации и подписи

Jinn-Client

Версия 2

Руководство программиста

RU.АМБС.58.29.12.010 33

Почтовый адрес:	**115127, Россия, Москва, а/я 66 ООО "Код Безопасности"**
Телефон:	**8 495 982-30-20**
E-mail:	**info@securitycode.ru**
Web:	**https://www.securitycode.ru**

Оглавление

Введение

Данное руководство предназначено для программистов, использующих изделие "Программа доверенной визуализации и подписи Jinn-Client. Версия 2" RU.АМБС.58.29.12.010 (далее — Jinn-Client, изделие). В нем содержатся сведения об использовании функций API (application programming interface) Jinn-Client.

Сайт в интернете. Информация о продуктах компании "Код Безопасности" представлена на сайте https://www.securitycode.ru.

Служба технической поддержки. Связаться со службой технической поддержки можно по телефону 8 800 505-30-20 или по электронной почте support@securitycode.ru.

Учебные курсы. Освоить аппаратные и программные продукты компании "Код Безопасности" можно в авторизованных учебных центрах. Список учебных центров и условия обучения представлены на сайте компании https://www.securitycode.ru/company/education/training-courses/.

Глава 1
Общие сведения

Назначение и основные функции Jinn-Client

Jinn-Client предназначен для формирования ЭП электронного документа, расположенного в оперативной памяти компьютера в виде XML-документа, текстового или бинарного файла. Формирование ЭП осуществляется в соответствии с положениями ст. 12 Федерального закона РФ "Об электронной подписи" от 06.04.2011 №63-ФЗ.

Jinn-Client реализует следующие основные функции:

- формирование ЭП в соответствии с криптографическим алгоритмом ГОСТ Р 34.10–2012 с функцией хэширования по ГОСТ Р 34.11–2012;
- формирование ЭП в форматах XMLDSig, XAdES-BES, CMS, CAdES-BES;
- формирование ЭП в среде операционной системы семейства Linux в графическом режиме;
- формирование ЭП серии документов — одна подпись под каждым документом;
- проверка корректности сформированной ЭП;
- доверенная визуализация документов в форматах txt, xml, pdf, bin средствами Jinn-Client;
- генерация ключей ЭП в криптографических контейнерах формата PKCS#15 и формирование запросов формата PKCS#10 на выпуск сертификатов ключей проверки ЭП в соответствии с методическими рекомендациями ТК 26;
- чтение ключей, сгенерированных средством криптографической защиты информации "КриптоПро CSP" версий 4, 5;
- фильтрация ключей для формирования ЭП;
- контроль целостности установленного ПО Jinn-Client.

Взаимодействие Jinn-Client с веб-порталом

Подписание документов может осуществляться локально после вызова Jinn-Client или на веб-портале.

Для подписания документов на веб-портале необходимо обеспечить его взаимодействие с Jinn-Client, установленным на компьютере пользователя, подписывающего документы. С этой целью на компьютер пользователя должен быть установлен плагин (расширение для веб-браузера).

Установку плагина выполняет администратор. Описание установки плагина Jinn Sign Extension приведено в [**3**].

Взаимодействие Jinn-Client с веб-порталом осуществляется в форме "запрос — ответ" с использованием интерфейса Window.postMessage:

- веб-портал формирует запрос и отправляет его в Jinn-Client;
- Jinn-Client обрабатывает запрос, формирует ответ и отправляет его веб-порталу;
- веб-портал принимает ответ и обрабатывает его.

Чтобы обеспечить взаимодействие Jinn-Client с веб-порталом, программист должен реализовать работу с API плагина Jinn-Client в соответствии с инструкциями, приведенными в следующей главе.

Внимание!

- Перед первым обращением к веб-порталу необходимо, чтобы после установки на компьютер Jinn-Client был запущен хотя бы один раз. В противном

случае на портале должно выводиться сообщение пользователю, поясняющее причину произошедшей ошибки.

- Если во время подписания на веб-портале пользователь обновил страницу, сформированная подпись не будет возвращена пользователю и будет утеряна.

Системные требования

Компьютеры, на которых предполагается использовать Jinn-Client, должны соответствовать следующим аппаратным и программным требованиям:

Операционная система x64	• Astra Linux Common Edition 2.12 "Орел" (kernel version 4.15.3); • Astra Linux Special Edition 1.5 (kernel version 4.2.0); • Astra Linux Special Edition 1.6 "Смоленск"; • CentOS 7.2 (kernel version 3.10); • RHEL 7.5 Desktop x64 (3.10 kernel version); • Ubuntu 16.04 LTS Desktop (версия ядра linux 4.4); • Ubuntu 18.04 LTS Desktop (версия ядра linux 4.15); • Альт Линукс СПТ 7.0.5 (ядро Linux 3.14); • Альт Линукс 8.2 (ядро Linux 4.9); • РЕД ОС 7.1 МУРОМ (Linux Kernel 4.9); • ROSA Enterprise Desktop (RED) X3 (Linux Kernel 4.9.41)
Операционная система x32	• Альт Линукс 8.2; • CentOS 7.2; • Ubuntu 16.04 LTS Desktop; • ROSA Enterprise Desktop (RED) X3
Процессор (тактовая частота)	Не менее 1,2 ГГц
Оперативная память	Не менее 4 Гбайт
Жесткий диск (свободное место)	Не менее 300 Мбайт
Привод	Привод DVD/CD-ROM
Интерфейсы	2 x USB 2.0; 1 x PCI-E — для исполнения 2; 1 x PCI/PCI-E/Mini PCI-E/Mini PCI-E Half — для исполнения 2
Дополнительное ПО	ПАК "Соболь" — для исполнения 2

Глава 2
Интерфейс взаимодействия плагина Jinn Sign Extension с веб-порталом

Запрос от веб-портала к плагину

Для взаимодействия веб- портала с плагином необходимо использовать window.postMessage(messageFromPortal, "*").

messageFromPortal — это json структура, имеющая следующее содержание:

- source: 'page', строка;
- secret_key: 'JinnSignExtensionSecretKey', строка;
- message, структура.

Для того чтобы обеспечить обработку сообщений внутри плагина только от портала, необходимо в структуру message включать

secret_key = JinnSignExtensionSecretKey.

Структура message

Message имеет следующую структуру:

- 'id': 'xxxxx', строка;
- 'context': 'yyyyy', строка;
- 'method': operation, одна строка из набора;
- 'data': data || {} , структура с данными или {}, если структура с данными не определена.

- id — уникальный идентификатор запроса, ответ вернется с тем же идентификатором, который состоит из произвольного набора символов. Например, строка, состоящая из HEX символов: XXXXXXXX- XXXX- XXXX- XXXX-XXXXXXXXXXXX.
- context — уникальный идентификатор контекста, единый для серии запросов внутри одного контекста, который состоит из произвольного набора символов. Например, строка, состоящая из HEX символов: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX.
- method — набор операций, значения:

Название операции	Назначение
CreateBinaryBase64CryptoContext	Открыть контекст для работы с подписанием в формате CMS/CAdES-BES/XMLDSig/XAdES-BES
SignDocument	Подписать документ CMS/CAdES-BES. Выполняется проверка криптографической части подписи. Если подписи не проходят проверку, процедура подписания завершается с отрицательным результатом
CheckSignature	Проверить подпись CMS/CAdES-BE/XMLDSig/XAdES-BES (относительно криптографии). Проверка выполняется с использованием отдельного интерфейса проверки подписи
SignHashDocument	Подписать хэш, CMS/CAdES-BES
CheckHashSignature	Проверить подпись хэша, CMS/CAdES-BES (относительно криптографии). Проверка выполняется с использованием отдельного интерфейса проверки подписи

CloseCryptoContext	Закрыть контекст по работе с CMS/CAdES-BES/XMLDSig/XAdES-BES
CreateXmlCryptoContext	Открыть контекст по работе с XMLDSig/XAdES-BES, DEPRECATED (устарел)
SignDocumentWithTransform	Подписать документ XMLDSig/XAdES-BES с применением дополнительного xsl-трансформа. Выполняется проверка криптографической части подписи. Если подписи не проходят проверку, процедура подписания завершается с отрицательным результатом

- data — содержание структуры data имеет зависимость от поля method.

  CreateBinaryBase64CryptoContext

  Поле data пустое.

  SignDocument

  Поле data содержит следующие элементы:

  - doc — данные для подписания в формате Base64, обязательный параметр;
  - options — structureOptions, структура, содержащая необязательные параметры:
    - needVisualize, bool — отображение документа перед подписанием;
    - showResult, bool — показать диалог об успешности операции;
    - mimeType - DEPRECATED, string — MIME тип ("формат") документа на подпись;
    - advanced, bool — тип подписи: CMS - false, CAdES-BES - true;
    - includeDocumentToCMS - DEPRECATED - bool — включить документ в подпись;
    - includeCertificateToCMS - DEPRECATED - bool — включить сертификат в подпись;
    - includeDocument, bool — его значение перезапишет параметр includeDocumentToCMS;
    - includeCertificate, bool — его значение перезапишет параметр includeCertificateToCMS;
    - signedAttributes, string особого формата (base64 asn-1 представление передаваемого параметра) — подписываемые атрибуты, включаемые в подпись;
    - unsignedAttributes, string особого формата — неподписываемые атрибуты, включаемые в подпись;
    - documentTitle, string — имя документа (при отображении в окне перед подписанием);
    - ownerCertificateName, string — имя владельца сертификата, ключом подписи которого выполняется подписание (при отображении в окне перед подписанием);
    - ownerCertificatePosition, string — должность владельца сертификата, ключом подписи которого выполняется подписание;
    - filter, string особого формата — фильтр для выбора сертификатов автора подписи в Jinn-Client.

  CheckSignature

  Поле data содержит следующие элементы:

  - doc — подписанный документ в формате BASE64, обязательный параметр, если проверяется отсоединенная подпись;
  - options — structureOptions, структура параметров. Содержание для CMS/CAdES-BES/XMLDSig/XAdES-BES:
    - showResult, bool — показать диалог об успешности операции;

- certificate, string — сертификат в формате BASE64. Обязательный параметр, если сертификат не включен в подпись;
- signature, string, подпись в формате BASE64. Обязательный параметр.

SignHashDocument

Поле data содержит следующие элементы:

- hash — данные для подписания в формате Base64, обязательный параметр;
- options — structureOptions, структура, содержащая необязательные параметры:
  - needVisualize, bool — отображение документа перед подписанием;
  - showResult, bool — показать диалог об успешности операции;
  - mimeType - DEPRECATED, string — MIME тип ("формат") документа на подпись;
  - advanced, bool — тип подписи: CMS - false, CAdES-BES - true;
  - hashOid, string — Oid алгоритма, на котором сформирован хэш;
  - includeCertificateToCMS, bool — включить сертификат в подпись;
  - includeCertificate, bool — его значение перезапишет параметр includeCertificateToCMS);
  - signedAttributes, string особого формата (base64 asn-1 представление передаваемого параметра) — подписываемые атрибуты, включаемые в подпись;
  - unsignedAttributes, string особого формата (base64 asn-1 представление передаваемого параметра) — неподписываемые атрибуты, включаемые в подпись;
  - documentTitle, string — имя документа при отображении;
  - ownerCertificateName, string — имя владельца сертификата, ключом подписи которого выполняется подписание;
  - ownerCertificatePosition, string — должность владельца сертификата, ключом подписи которого выполняется подписание;
  - filter, string особого формата — фильтр для выбора сертификатов автора подписи в Jinn-Client.

CheckHashSignature

Поле data содержит следующие элементы:

- hash, string — данные для проверки в формате Base64, обязательный параметр;
- options — structureOptions, структура параметров. Содержание (параметры signature и certificate обязательные):
  - showResult, bool — показать диалог об успешности операции;
  - certificate, string — сертификат в формате BASE64, обязательный параметр, если сертификат не включен в подпись;
  - signature, string — подпись в формате BASE64, обязательный параметр.

CloseCryptoContext

Поле data пустое.

CreateXmlCryptoContext

Поле data пустое.

SignDocumentWithTransform

Поле data содержит следующие элементы:

- doc — данные для подписания в формате Base64, обязательный параметр;
- options — structureOptions, структура, содержащая необязательные параметры:

- needVisualize, bool — отображение документа перед подписанием;
- showResult, bool — показать диалог об успешности операции;
- mimeType - DEPRECATED, string — MIME тип ("формат") документа на подпись;
- advanced, bool — тип подписи: XMLDSig - false, XAdES-BES - true;
- includeDocument, bool — включить документ в подпись;
- signedProperties, string особого формата (base64 xmldsig ноды передаваемого параметра) — подписываемые xml параметры, включаемые в подпись;
- unsignedProperties, string особого формата (base64 xmldsig ноды передаваемого параметра) — неподписываемые xml параметры, включаемые в подпись;
- propertiesNamespaceRef, string — ссылка на пространство имен для xml элементов signedProperties и unsignedProperties; параметр задается только совместно с заданием параметра propertiesNamespacePrefix (см. ниже);
- propertiesNamespacePrefix, string — префикс имен xml элементов signedProperties и unsignedProperties; параметр задается только совместно с заданием параметра propertiesNamespaceRef (см. выше);
- documentTitle, string — имя документа при отображении;
- ownerCertificateName, string — имя владельца сертификата, ключом подписи которого выполняется подписание (отображается при визуализации);
- ownerCertificatePosition, string — должность владельца сертификата, ключом подписи которого выполняется подписание (отображается при визуализации);
- xsltUri, string — ссылка на xslt-преобразование;
- elementUri , string — подписать только указанный данной строкой xml-параметр;
- xslt, string — xslt-преобразование в формате BASE64;
- filter, string особого формата — фильтр для выбора сертификатов автора подписи в Jinn-Client.

Взаимозависимости параметров для (XMLDSig/XAdES-BES) подписи:

- если XsltUri задано, то данная ссылка на xslt-преобразование должна быть доступна на чтение;
- propertiesNamespaceRef, propertiesNamespacePrefix оказывают влияние на конечный подписанный документ только в случае, если заданы signedProperties или unsignedProperties;
- порядок наложения преобразований подписываемого xml-документа согласно передаваемым параметрам:
  i. elementUri
  ii. xsltUri
  iii. xslt

Ответ от плагина веб-порталу

Для того чтобы принять ответ от плагина, необходимо использовать:

window.addEventListener(‘message’, function(event) { //processing }, false).

message — это json структура, имеющая следующее содержание:

- source: 'content_script', строка;
- secret_key: 'JinnSignExtensionSecretKey', строка;
- message: msg, структура.

msg имеет следующую структуру:

- 'id': 'xxxxx', строка;
- 'context': 'yyyyy', строка;
- 'data': data, структура.

Содержание структуры data

- **error**, строка, сообщение-статус об ошибке или успешности операции, необязательный параметр.
  - "Ошибка парсинга JSON запроса".
  - "Контекст подписания открыт".
  - "Контекст подписания закрыт".
  - "Не удалось определить тип операции".
  - "Ошибка внутри контекста".
  - "Лицензия недействительна".
  - "Проверка подписи завершилась успешно".
  - "Подпись не прошла проверку".
  - "Контекст отменен пользователем".
  - "Операция отменена пользователем".
  - "Документ успешно подписан".
  - "Операция завершилась ошибкой".
  - "Достигнут лимит в 1000 запросов на контекст".
  - "Достигнут лимит ожидания обработки запроса".
  - "Соединение прервано".
- **result**, код ошибки — 0 в случае, если операция прошла успешно, иначе коды ошибок по операциям, обязательный параметр.
  - 0, операция успешна.
  - -1, соединение прервано.
  - -2, ошибка разбора (парсинга) JSON запроса.
  - -3, неверно задан метод.
  - -4, лицензия недействительна.
  - -5, операция завершилась ошибкой.
  - -6, не удалось сохранить сформированную подпись.
  - -7, ошибка преобразования подписи в base64.
  - -8, ошибка проверки подписи.
  - -9, операция отменена.
  - -10, контекст отменен.
  - -11, внутри данного контекста подписания была обнаружена ошибка или данный контекст был отменен.
  - -12, достигнут лимит ожидания на открытие нового контекста.
- **signature**, строка, подпись в формате base64, необязательный параметр.
- **tall**, **verify** – зарезервированы для использования в следующих версиях.

Общий сценарий взаимодействия плагина и веб-портала

Интерфейс вызовов плагина, состоит из 3 типов сообщений (запросов):

1. Открытие контекста
   (CreateBinaryBase64CryptoContext, CreateXmlCryptoContext- Deprecated).
2. n операций k, n = { 0,1,2,3,.. }, k = { SignDocument, CheckSignature, SignHashDocument, CheckHashSignature, SignDocumentWithTransform }.
3. Закрытие контекста
   (CloseCryptoContext).

Для каждого из отправленных запросов плагин отсылает ответ. Соответственно добавляется еще три типа сообщения (ответа):

- ответ на открытие контекста;
- ответ (с результатом) операции;
- ответ на закрытие контекста.

Для корректной реализации сценариев подписи или проверки подписи одного документа необходимо соблюсти следующую последовательность отсылки запросов и обработки ответов для одного контекста подписания:

1. Открытие контекста.
2. Обработка ответа на открытие контекста.
3. Запрос на операцию подписи или проверки подписи.
4. Обработка ответа (с результатом) операции.
5. Закрытие контекста.
6. Обработка ответа на закрытие контекста подписания.

Сначала отправляется запрос на открытие контекста (сообщение первого типа), он может высылаться для одного контекста только один раз. При подписании с нескольких порталов, страниц или вкладок для каждого запроса должно высылаться свое сообщение на открытие контекста со своим идентификатором.

После получения от приложения (плагина) ответа на открытие контекста (сообщение второго типа) следует отправлять запрос на подписание для открытого контекста (сообщение третьего типа).

После получения ответа на подписание (сообщение четвертого типа) в данном контексте можно выслать новый запрос на подписание (сообщение третьего типа) или запрос на закрытие контекста (сообщение пятого типа).

На каждый запрос с закрытием контекста (сообщение пятого типа) ожидается ответ на закрытие контекста (сообщение шестого типа). После чего работу с данным контекстом можно считать законченной.

Отправленные запросы с разными контекстами могут обрабатываться параллельно, однако сам процесс подписания документа, начиная с выбора контейнера и заканчивая результатом операции подписания, становится в очередь.

В случае, если запрос на открытие контекста завершился с ошибкой (например, "соединение прервано"), дальнейшие запросы в данном сценарии отправлять нет необходимости, однако, если контекст уже был открыт, то запрос на закрытие контекста должен быть отправлен вне зависимости от результатов операций внутри контекста.

Пример формирования подписи CAdES-BES

ToPlugin 1(Create context) json-структура запроса на открытие контекста подписания

{"id":"afdc06ec- c053- 4edd- 88f0- 89845a82c77e","context":"bbe99e31- 0e23- 4d26-83ee-dbb9247df606","method":"CreateBinaryBase64CryptoContext","data": {}}

id — уникальный идентификатор запроса. Ответ требуется с тем же идентификатором, который состоит из произвольного набора символов. Например, строка, состоящая из HEX символов: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX.

context — уникальный идентификатор контекста, единый для цепочки из трех типов запросов, который состоит из произвольного набора символов. Например, строка, состоящая из HEX символов: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX.

method — Тип запроса ("CreateBinaryBase64CryptoContext" - открытие контекста).

FromPlugin 1(Create context) json-структура ответа на открытие контекста подписания

{"context":"bbe99e31-0e23-4d26-83ee-dbb9247df606","data":{"error":" Контекст подписания открыт ","result":0,"tall":0,"verify":0},"id":"afdc06ec-c053-4edd-88f0-89845a82c77e"}

data — поле с содержимым запроса или ответа;

error — статус-сообщение запроса;

result — результат запроса;

tall, verify — зарезервированы(не используются).

ToPlugin 2(Sign) json-структура запроса на подписание

{"id":"cfcad461-f4e4-45ae-b5b3-2131b69a11b4","context":"bbe99e31-0e23-4d26-83ee-dbb9247df606","method":"SignDocument","data":
{"doc":"MDEyMzQ1Njc4OTEwYWJjZGVmZ2hpamtsbW5vcHFyc3R1dnd4eXoK","options":
{"needVisualize":false,"showResult":true,"includeDocument":false,"includeCertificate":true,"advanced":true}}}

method — тип запроса ("SignDocument" — подпись документа);

doc — подписываемые данные в формате base64;

options — опции подписания;

needVisualize — опция-флаг визуализации перед подписанием;

showResult — показать диалог об успешности операции;

includeDocument — опция-флаг включения документа в подпись;

includeCertificate — опция-флаг включения сертификата в подпись;

advanced — опция-флаг усиленной подписи, в данном случае CAdES-BES (XAdES-BES для XMLDSig).

FromPlugin 2(Sign) json-структура ответа на подписание

{"context":"bbe99e31-0e23-4d26-83ee-dbb9247df606","data": {"error":" Документ успешно подписан ","result":0,"signature":"MIIF9gYJKoZIhvcNAQcCoIIF5zCCBeMCAQExDjAMBggqhQMHAQ ","tall":0,"verify":0},"id":"cfcad461-f4e4-45ae-b5b3-2131b69a11b4"}

signature — сформированная подпись в Base64.

ToPlugin 3(Close context) json-структура запроса на закрытие контекста подписания

{"id":"7599e952-b786-4dbe-99fc-065e12faaa88","context":"bbe99e31-0e23-4d26-83ee-dbb9247df606","method":"CloseCryptoContext","data":{}}

method — тип запроса ("CloseCryptoContext" — закрытие контекста)

FromPlugin 3(Close context) json-структура ответа на закрытие контекста подписания

{"context":"bbe99e31-0e23-4d26-83ee-dbb9247df606","data": {"error":" Контекст подписания закрыт ","result":0,"tall":0,"verify":0},"id":"7599e952-b786-4dbe-99fc-065e12faaa88"}

Глава 3
Фильтр для выбора сертификатов автора подписи

Данный фильтр предназначен для того, чтобы в сценарии подписи документов с использованием бруазерного плагина отфильтровать сертификаты по признаку или признакам, заданным в фильтре.

Фильтр представляет собой строковое выражение, состоящее из нескольких выражений (условий).

Ограничение на передаваемую строку — 4111 символов.

Сертификат будет считаться соответствующим фильтру, только если он соответствует всем условиям, составляющим фильтр. Однако при выборе сертификата в сценарии подписания, имеется возможность убрать фильтр, установив флаг “Отобразить все ключи”. В таком случае, сертификаты, которые не соответствуют фильтру, будут отображаться шрифтом серого цвета, тогда как соответствующие фильтру сертификаты будут отображаться шрифтом черного цвета.

Фильтр сертификатов не запрещает использовать не соответствующие сертификаты при подписании.

Выражения в фильтре разделяются строкой "AND". Выражение может состоять из одной или двух частей. Рекомендованное количество выражений — не более 2 на фильтр.

Левая часть — элемент в ASN.1-структуре. Необязательная правая часть содержит проверочное значение.

Левая часть указывает на элемент в ASN.1-структуре, т.е. содержит описание пути к проверяемому элементу.

Проверяемый элемент должен иметь примитивный ASN.1-тип (один из строковых, INTEGER, BOOLEAN, OBJECT IDENTIFIER…). В случае если путь указывает на элемент составного типа, для сравнения будет выбран первый найденный в нем элемент примитивного типа.

Описание пути к элементу в ASN.1-структуре состоит из последовательности имен ASN.1-типов или элементов, разделенных символом ':'. Данная последовательность имен обрабатывается без учета регистра. Допустимые имена типов и элементов составлены в соответствии с RFC5280 (X.509) и приказом ФСБ№795 "О требованиях к форме и составу квалифицированного сертификата".

Имена ASN.1-типов, рекомендованных к использованию:

[CertificateSerialNumber];

[PublicKeyAlgorithm];

[CommonName];

[OrganizationName];

[organizationalUnitName];

[Title];

[EmailAddress];

[Surname];

[GivenName];

[LocalityName];

[Snils].

В дополнение к именам, описанным в указанных документах, возможно использование зарезервированного имени ‘any’, которое соответствует любому числу элементов любого типа.

Поиск элементов по заданному пути всегда выполняется так, как если бы путь начинался с ‘any’, т.е. нет необходимости прописывать полный путь начиная с "Certificate, TBSCertificate, …. ".

Необязательная правая часть содержит проверочное значение, с которым необходимо сравнить значение элемента, найденного в ASN.1-структуре сертификата.

В случае отсутствия правой части условие считается выполненным, если элемент по указанному пути будет найден. Иначе выполняется проверка на точное совпадение значений.

В связи с тем, что строковые данные могут содержать различные символы, требующие экранирования, проверочное значение должно быть представлено в виде base64-строки.

Исходным значением для формирования base64-строки всегда должно быть UTF-8-представление проверочного значения – вне зависимости от проверяемого ASN.1-типа (BMPString, UTF8String, BOOLEAN, OBJECT IDENTIFIER …).

Способы задания фильтра

1. Поиск по наличию данного поля в сертификате [commonName].
2. Поиск по прямому значению данного поля [commonName]=BASE64.
3. Поиск по структуре в x509 [Subject:any:commonName]=BASE64 (any - термин подразумевает любой тип).
4. Комбинации пар из 1-3 с помощью AND.

Способ, указанный в п. **3**, не рекомендован к использованию.

Порядок проверки

1. Проверочное значение извлекается из base64.
2. Значение проверяемого элемента переводится в строковое представление в кодировке UTF-8.
3. Выполняется поиск заданной ASN.1 структуры в сертификате.
4. Выполняется проверка на точное совпадение значений, найденного ASN.1-типа, если выражение фильтра было задано со значением. Выполняется проверка на точное совпадение строк, полученных выше.

Примеры фильтров

Пример 1

[commonName]="aWQyMDAxX0NfQw=="

Данному фильтру будут соответствовать только сертификаты, в которых будет найдено поле commonName “id2001_C_C”.

“id2001_C_C” ---Base64---> "aWQyMDAxX0NfQw=="

Пример 2

[subject:any:snils]="MTIzNDEyMzQxMjM0Cg==" AND [authoritykeyidentifier, any, authoritycertissuer]

Данному фильтру будут соответствовать только сертификаты, владелец которых имеет СНИЛС “123412341234”, а также содержащие расширение AuthorityKeyIdentifier в варианте с полем AuthorityCertIssuer.

Приложение

Пример формирования строки особого формата

Ниже приведен пример формирования содержания в запросе и формирования подписываемых и неподписываемых свойств в XAdES-BES подписи.

"signedProperties":"Attribute1:wPLw6OHz8iAx&Attribute2:wPLw6OHz8iAy"

"unsignedProperties":"Attribute3:wPLw6OHz8iAz"

"Attribute1:wPLw6OHz8iAx&Attribute2:wPLw6OHz8iAy"

Attribute1, Attribute2 — название подписываемого свойства;

: — управляющий символ, означающий, что за ним следует содержание данного свойства;

wPLw6OHz8iAx — значение свойства в base64;

& — управляющий символ, позволяющий задавать несколько подписываемых свойств.

Пример запроса CAdES-BES подписи

Перед запросом требуется отправить запрос на открытие контекста, а после него — на закрытие контекста.

{"context":"32e459e5-59f8-486b-bb1a-e132e378557f", "data":{"options":{"needVisualize":true "showResult":true "mimeType":"text" "advanced":true "includeDocumentToCMS":false "includeCertificateToCMS":true "includeDocument":false "includeCertificate":true "signedAttributes":"2.12.3.23576.11:DA7Qn9GA0LjQvNC10YAgMQ==&2.12.3.23576.12:DA7Qn9GA0LjQvNC10YAgMg==" "unsignedAttributes":"2.12.3.23576.13:DA7Qn9GA0LjQvNC10YAgMw==" "documentTitle":"Тестовый документ" "ownerCertificateName":"Иванов Иван Иванович" "ownerCertificatePosition":"Главный бухгалтер" "filter":"[commonName]=aWQyMDAxX0NfQw=="}, "doc":"MDEyMzQ1Njc4OTEwYWJjZGVmZ2hpamtsbW5vcHFyc3R1dnd4eXoK"}, "id":"6365b504-b770-42ac-af82-9c75e2c5694d", "method":"SignDocument"}

Пример запроса XAdES-BES подписи

Перед запросом требуется отправить запрос на открытие контекста, а после него — на закрытие контекста.

{"context":"32e459e5-59f8-486b-bb1a-e132e378557f", "data":{"options":{"needVisualize":true "showResult":true "mimeType":"text" "advanced":true "includeDocument":true "includeCertificate":true "signedProperties":"Attribute1:wPLw6OHz8iAx&Attribute2:wPLw6OHz8iAy" "unsignedProperties":"Attribute3:wPLw6OHz8iAz" "xsltUri":"http://www.aaa-xslt.ru" "documentTitle":"Тестовый документ" "ownerCertificateName":"Иванов Иван Иванович" "ownerCertificatePosition":"Главный бухгалтер" "elementUri":"Example1" "propertiesNamespaceRef":"http://uri.etsi.org/01903/v1.4.1#" "propertiesNamespacePrefix":"xades" "filter":"[commonName]=aWQyMDAxX0NfQw=="},
"xslt":"PHhzbDpzdHlsZXNoZWV0IHZlcnNpb249IjEuMCIgeG1sbnM6eHNsPSJodHRwOi8vd3d3LnczLm9y
Zy8xOTk5L1hTTC9UcmFuc2Zvcm0iPg0KICA8eHNsOnRlbXBsYXRlIG1hdGNoPSJAKnxub2RlKCki
Pg0KICAgIDx4c2w6Y29weT4NCiAgICAgICA8xc2w6YXBwbHktdGVtcGxhdGVzIHNlbGVjdD0iQCp8
bm9kZSgpIi8+DQogICAgICA8eHNsOnRleHQ+IHRyMSA8L3hzbDp0ZXh0PgogICAgDwveHNsOmNv
cHk+DQogIDwveHNsOnRlbXBsYXRlPg0KPC94c2w6c3R5bGVzaGVldD4=",
"doc":"PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0idXRmLTgiPz4NCjxkb2M+DQogIDxjb250ZW5
0
IGlkPSJFeGFtcGxlMSI+DQoJCTxmaWVsZCBzaWduPSIxIiBvcmRlcj0iMSIgbGFiZWw9ItCU0LDR
gtCwIiB2YWx1ZT0iMDEuMDguMjAxMiIvPg0KICA8L2NvbnRlbnQ+DQo8L2RvYz4NCg==="},
"id":"6365b504-b770-42ac-af82-9c75e2c5694d", "method":"SignDocumentWithTransform"}

Документация

1. Программа доверенной визуализации и подписи Jinn-Client. Версия 2. Руководство пользователя.
2. Программа доверенной визуализации и подписи Jinn-Client. Версия 2. Руководство программиста.
3. Программа доверенной визуализации и подписи Jinn-Client. Версия 2. Руководство администратора.