127018, Москва, Сущевский Вал, 18
Телефон: (495) 9954820
Факс: (495) 995 4820
http://www.CryptoPro.ru
E-mail: info@CryptoPro.ru

УТВЕРЖДЕНЫ
ЖТЯИ.00101-01 95 01-ЛУ

Средство Криптографической Защиты Информации

КриптоПро CSP

Версия 5.0 КС1

1-Base

Правила пользования

ЖТЯИ.00101-01 95 01

Листов 60

2019 г.

Содержание

Аннотация 4
1 Назначение СКЗИ. Условия эксплуатации 5
2 Порядок распространения СКЗИ 7
3 Ключевая система и ключевые носители 8
3.1 Шифрование данных 8
3.2 Создание и проверка ЭП 8
3.3 Ключевые носители 9
3.4 Ключевые контейнеры 10
3.5 Сертификаты ключа 10
3.6 Размеры и сроки действия ключей 11
3.6.1 Изменение параметра ControlKeyTimeValidity 11
3.7 Хранение ключевых носителей 12
3.8 Взаимодействие с пользователем при работе с ключевыми носителями 13
3.8.1 Счетчики операций при использовании ФКН 13
3.9 Управление ключами 13
3.9.1 Формирование ключей 14
3.9.2 Компрометация ключей пользователя 15
3.9.3 Уничтожение ключей на ключевых носителях 16
4 Требования по встраиванию и использованию ПО СКЗИ 17
5 Требования по защите от НСД 18
5.1 Общие требования по организации работы по защите от НСД 18
5.2 Требования по размещению технических средств с установленным СКЗИ 18
5.3 Требования по установке СКЗИ, общесистемного и специального ПО на ПЭВМ 19
5.4 Меры по обеспечению защиты от НСД 20
5.5 Действия при компрометации ключей 23
5.6 Требования по подключению СКЗИ для работы по общедоступным каналам передачи данных 23
5.7 Требования по использованию в СКЗИ программно-аппаратных средств защиты от НСД 24
6 Использование программных интерфейсов 26
Приложение 1. Контроль целостности программного обеспечения 27
Приложение 2. Перечень вызовов, использование которых при разработке систем на основе СКЗИ «КриптоПро CSP» с учетом п.1.5 Формуляра ЖТЯИ.00101-01 30 01 возможно без дополнительных тематических исследований 30
Литература 59

Аннотация

Данный документ содержит правила пользования средства криптографической защиты информации (СКЗИ) «КриптоПро CSP» версия 5.0 КС1 Исполнение 1-Base, его состав, назначение, ключевую систему, требования по защите от НСД.

Документ предназначен для администраторов информационной безопасности учреждений, осуществляющих установку, обслуживание и контроль за соблюдением требований к эксплуатации средств СКЗИ, для администраторов Серверов, сетевых ресурсов предприятия, для работников службы информационной безопасности, осуществляющих настройку рабочих мест для работы со средствами СКЗИ, а также для пользователей СКЗИ.

Инструкции администраторам безопасности и пользователям различных автоматизированных систем, использующих СКЗИ «КриптоПро CSP» версия 5.0 КС1 Исполнение 1-Base, должны разрабатываться с учетом требований настоящих Правил.

1 Назначение СКЗИ. Условия эксплуатации

СКЗИ «КриптоПро CSP» версии 5.0 КС1 представляет собой программный комплекс, предназначенный для реализации широкого набора решений по обеспечению криптографическими методами информационной безопасности на отдельных рабочих местах, в архитектуре «клиент-сервер», а также в информационных и телекоммуникационных системах различного назначения.

СКЗИ «КриптоПро CSP» может выступать как в качестве готового к применению средства, так и в качестве платформы для построения на его основе программных, программно-аппаратных и аппаратных решений в области обеспечения информационной безопасности, основанных на применении криптографических алгоритмов.

При эксплуатации СКЗИ «КриптоПро CSP» версии 5.0 КС1 должны выполняться следующие требования:

- Средствами СКЗИ не допускается обрабатывать информацию, содержащую сведения, составляющие государственную тайну.
- Допускается использование СКЗИ для криптографической защиты персональных данных.
- Ключевая информация является конфиденциальной.
- Срок действия ключа проверки ЭП – не более 15 лет после окончания срока действия соответствующего ключа ЭП.
- Внешняя гамма, используемая для инициализации состояния программного ДСЧ, является конфиденциальной.
- СКЗИ должно использоваться со средствами антивирусной защиты, сертифицированными ФСБ России. В случае их отсутствия рекомендуется по возможности использовать существующие антивирусные средства защиты. Класс антивирусных средств защиты определяется условиями эксплуатации СКЗИ в автоматизированных системах.
- Размещение СКЗИ в помещениях, предназначенных для ведения переговоров, в ходе которых обсуждаются вопросы, содержащие сведения, составляющие государственную тайну, осуществляется установленным порядком.
- ПЭВМ, на которых используется СКЗИ, должны быть допущены для обработки конфиденциальной информации по действующим в Российской Федерации требованиям по защите информации от утечки по техническим каналам, в том числе, по каналу связи (например, СТР-К), с учетом модели угроз в информационной системе заказчика, которым должно противостоять СКЗИ.
- Инсталляция СКЗИ на рабочих местах должна производиться только с дистрибутива, полученного по доверенному каналу.
- При использовании в СКЗИ сертификатов открытых ключей (сертификатов ключей проверки ЭП) должна быть обеспечена (с использованием сертифицированного ФСБ России УЦ) их актуальность и целостность. Актуальность должна обеспечиваться подтверждением использования сертификата в рамках установленного срока его действия и проверкой на отозванность. Проверка на отозванность может осуществляться при помощи списков отозванных (аннулированных) сертификатов — COC (Certificate Revocation List — CRL), а также при помощи обращения к сервису OCSP. Целостность должна обеспечиваться проверкой ЭП УЦ в сертификате.

Организация и обеспечение безопасности хранения, обработки и передачи по каналам связи с использованием средств криптографической защиты информации сведений, составляющих конфиденциальную информацию, осуществляются в соответствии с документами «Инструкция об организации и обеспечении безопасности хранения, обработки и передачи по

каналам связи с использованием средств криптографической защиты информации с ограниченным доступом, не содержащей сведений, составляющих государственную тайну» (Приказ ФАПСИ № 152 от 13 июня 2001 года), «Положение о разработке, производстве, реализации и эксплуатации шифровальных (криптографических) средств защиты информации» (Приказ ФСБ России № 66 от 9 февраля 2005 года) и другими руководящими документами по обеспечению безопасности информации.

СКЗИ «КриптоПро CSP» версии 5.0 КС1 можно эксплуатировать со следующими программными продуктами без проведения исследований по оценке влияния:

- приложения, входящие в состав ОС;
- приложение командной строки для формирования запроса на сертификат certreq;
- MS Outlook (Office 2007, Office 2010, Office 2013, Office 2016, Office 2019);
- сервер IIS.

Необходимость проведения оценки влияния для прочих программных продуктов (в том числе установленных администратором/пользователем дополнений и расширений программного обеспечения, перечисленного выше) определяется с учетом п.1.5 Формуляра ЖТЯИ.00101-01 30 01.

Следующие приложения из состава СКЗИ «КриптоПро CSP» версии 5.0 КС1:

- приложение командной строки для подписи и шифрования файлов cryptcp;
- приложение командной строки для работы с сертификатами certmgr;
- приложение для создания TLS-туннеля stunnel (кроме ОС iOS и Android),

можно использовать без проведения оценки влияния указанных приложений на программные интерфейсы СКЗИ «КриптоПро CSP» версии 5.0 КС1, но требуется проведение оценки влияния компонентов среды функционирования (прикладных систем, программных продуктов и т.п.), вызывающих указанные приложения, на их штатное функционирование.

Проведение тематических исследований программных продуктов и приложений, указанных в настоящем разделе, не требуется.

Использование СКЗИ «КриптоПро CSP» версии 5.0 КС1 с выключенным режимом усиленного контроля использования ключей не допускается. Включение данного режима описано в документах ЖТЯИ.00101-01 91 02. Руководство администратора безопасности. Windows, ЖТЯИ.00101-01 91 03. Руководство администратора безопасности. Linux, ЖТЯИ.00101-01 91 04. Руководство администратора безопасности. FreeBSD, ЖТЯИ.00101-01 91 05. Руководство администратора безопасности. Solaris, ЖТЯИ.00101-01 91 06. Руководство администратора безопасности. AIX, ЖТЯИ.00101-01 91 07. Руководство администратора безопасности. Mac OS, ЖТЯИ.00101-01 91 08. Руководство администратора безопасности. iOS.

2 Порядок распространения СКЗИ

Установочные модули СКЗИ «КриптоПро CSP» версии 5.0 КС1 и комплект эксплуатационной документации к нему могут поставляться пользователю Уполномоченной организацией двумя способами:

1) на носителе (CD, DVD-диски);

2) посредством загрузки через Интернет.

Для получения возможности загрузки установочных модулей СКЗИ «КриптоПро CSP» версии 5.0 КС1 и комплекта эксплуатационной документации пользователь направляет свои учетные данные Уполномоченной организации. Учетные данные могут быть направлены посредством заполнения специализированной регистрационной формы на сайте Уполномоченной организации.

После получения Уполномоченной организацией учетных данных пользователю предоставляется доступ на страницу загрузки установочных модулей СКЗИ «КриптоПро CSP» версии 5.0 КС1 и комплекта эксплуатационной документации. При загрузке пользователем установочных модулей СКЗИ «КриптоПро CSP» версии 5.0 КС1 и комплекта эксплуатационной документации Уполномоченной организацией присваивается учетный номер, идентифицирующий экземпляр СКЗИ «КриптоПро CSP» версии 5.0 КС1, предоставленный пользователю.

На странице загрузки вместе с дистрибутивом и документацией размещается отделенная электронная подпись, для проверки которой необходимо использовать утилиту cpverify, полученную доверенным образом и содержащую ключ проверки данной электронной подписи.

Установка СКЗИ «КриптоПро CSP» версии 5.0 КС1 на рабочее место пользователя может быть осуществлена только в случае подтверждения целостности полученных установочных модулей СКЗИ «КриптоПро CSP» версии 5.0 КС1 и эксплуатационной документации.

Примечание.

1) Средство контроля целостности (cpverify.exe) первоначально должно быть получено пользователем на физическом носителе в офисе компании ООО «КРИПТО-ПРО», либо у официального дилера. Такая утилита считается полученной доверенным образом. Далее полученной доверенным образом признается очередная версия утилиты, полученная любым образом, например, скачанная с сайта www.cryptopro.ru, при условии, что она была проверена другим экземпляром утилиты, полученным ранее доверенным образом, и проверка была успешной.

2) Ключ проверки ЭП, а также информация о нем (дата создания, алгоритм хэш-функции, идентификатор алгоритма подписи) записываются в исходный код утилиты на этапе сборки.

3) Контроль целостности дистрибутива СКЗИ и компонентов СФ обеспечивается при помощи утилиты cpverify.exe в соответствии с Приложением 1.

3 Ключевая система и ключевые носители

СКЗИ «КриптоПро CSP» версии 5.0 КС1 является системой с открытым распределением ключей на основе асимметричной криптографии с использованием закрытого ключа на одной стороне и соответствующего ему открытого ключа на другой стороне. Такие пары ключей могут быть двух типов: ключи электронной подписи (ЭП) и ключи обмена.

Ключ ЭП может быть использован только для создания ЭП. Ключ обмена может быть использован как для формирования ключа связи с другим пользователем, так и для создания ЭП.

Пользователь включается в систему и исключается из неё установленным Удостоверяющим центром порядком.

Пользователь, обладающий правом подписи и/или шифрования данных, вырабатывает на своем рабочем месте или получает у администратора безопасности (в зависимости от принятой политики безопасности) личные закрытый ключ (ключ ЭП) и открытый ключ (ключ проверки ЭП). На основе каждого открытого ключа (ключа проверки ЭП) Удостоверяющим Центром формируется сертификат открытого ключа (сертификат ключа проверки ЭП).

3.1 Шифрование данных

При зашифровании сообщения пользователем А для пользователя Б, пользователь А формирует симметричный ключ связи (сеансовый ключ информационного обмена) на основе своего закрытого ключа обмена и открытого ключа обмена пользователя Б. Соответственно, для расшифрования этого сообщения пользователем Б формируется тот же симметричный ключ на основе своего закрытого ключа обмена и открытого ключа обмена пользователя А.

Таким образом, для обмена данными каждому пользователю необходимо иметь:

- личный закрытый ключ обмена;
- открытые ключи обмена других пользователей.

Сеансовый ключ информационного обмена вырабатывается на основе схемы Диффи-Хеллмана. Схема Диффи-Хеллмана обеспечивает формирование сеансовых ключей, но не обеспечивает аутентификацию связывающихся сторон. Поэтому данная схема должна использоваться совместно с протоколами аутентификации, например «КриптоПро IKE» и т.п.

3.2 Создание и проверка ЭП

Для создания электронной подписи используется ключ электронной подписи, для проверки – соответствующий ключ проверки электронной подписи. Проверяющий должен быть полностью уверен в принадлежности ключа проверки ЭП конкретному пользователю. Для этой цели используется сертификат ключа проверки ЭП, подписанный Удостоверяющим Центром.

Каждому пользователю, обладающему правом подписи, необходимо иметь:

- ключ электронной подписи;
- ключи проверки электронной подписи (сертификаты ключей проверки электронной подписи) других пользователей.

3.3 Ключевые носители

Ключи пользователей могут храниться на различных устройствах, называемых «ключевые носители» («ключевые хранилища»). Ключи на ключевых носителях хранятся в виде специальной структуры, называемой «ключевой контейнер».

Виды ключевых носителей и их использование в зависимости от программно-аппаратной платформы отражено в ЖТЯИ.00101-01 30 01. КриптоПро CSP. Формуляр, п. 3.9.

С точки зрения принципов использования ключей ключевые носители подразделяются на следующие типы:

– Функциональный ключевой носитель (ФКН). Такой ключевой носитель содержит в себе функциональные возможности генерации ключей, обеспечения невозможности экспорта хранимых ключей за пределы носителя и содержит в себе реализацию криптографических алгоритмов, выполняемых с использованием хранимых на носителе ключей (т.н. активный вычислитель). Конкретный набор функций и их реализация зависят от ФКН конкретного разработчика и от модуля поддержки ФКН в СКЗИ. Важной особенностью при работе с ФКН, поддерживаемыми СКЗИ «КриптоПро CSP», является защита канала связи между ФКН и СКЗИ по протоколу SESPAKE. Для некоторых носителей такая поддержка отсутствует (см. соответствующие примечания к таблице 3.1 документа ЖТЯИ.00101-01 30 01. КриптоПро CSP. Формуляр).

– Пассивный ключевой носитель (пассивное хранилище ключевой информации). Такой ключевой носитель предназначен только для хранения ключевых контейнеров пользователей. При необходимости использования такого ключевого носителя требуемый ключевой контейнер обрабатывается средствами СКЗИ.

– Облачный токен (модуль взаимодействия с КриптоПро DSS). Реализует те же функциональные возможности, что и ФКН. Позволяет выполнять операции с ключами, хранящимися в сервисе «КриптоПро DSS». При этом защита канала связи между СКЗИ и сервисом «КриптоПро DSS» обеспечивается по протоколу TLS.

Подробнее об использовании облачного токена см. п. 2.4 документа ЖТЯИ.00101-01 92 06. КриптоПро CSP. Инструкция по использованию графического приложения. Инструменты КриптоПро.

При использовании пассивных ключевых носителей и ФКН без поддержки SESPAKE необходимо обеспечить выполнение следующих условий:

– подключение съемных носителей должно осуществляться непосредственно к считывателю (считавателю смарт-карт, USB-портам и т.п.), с обеспечением отсутствия канала связи между носителем и СКЗИ, в котором может действовать нарушитель;

– при конструктивном исполнении считывателя ключевого носителя с кабелем необходимо обеспечить нахождение считывателя и кабеля на той же контролируемой территории, что и ПЭВМ.

При невозможности выполнения указанных условий необходимо с учетом модели возможных угроз и нарушителя разработать организационно-технические мероприятия по защите взаимодействия носителя с СКЗИ с последующей оценкой таких мероприятий в рамках проведения соответствующих исследований.

При использовании любых типов ключевых носителей помимо требований эксплуатационной документации на СКЗИ «КриптоПро CSP» необходимо руководствоваться эксплуатационной документацией на сами носители.

Доступ к ключевому носителю любого типа должен быть защищен паролем и/или PIN-кодом (за исключением случаев, отдельно оговоренных в документации на СКЗИ «КриптоПро

CSP» и используемые носители). Носители могут поддерживать код доступа PUK, используемый для разблокировки носителя. Перед началом работы с ключевым носителем необходимо сменить установленные по умолчанию значения PIN и PUK (при поддержке PIN и PUK на носителе).

3.4 Ключевые контейнеры

Закрытые ключи СКЗИ КриптоПро CSP версии 5.0 КС1 хранятся в ключевом контейнере, который может содержать в себе:

- только ключ ЭП;
- только ключ обмена;
- ключ ЭП и ключ обмена одновременно.

Единственный ключ ключевого контейнера (ключ ЭП или ключ обмена) называется первичным ключом. Если в контейнере два ключа, то первый ключ (ключ ЭП) называется первичным, второй ключ (ключ обмена) — вторичным.

Кроме ключей, ключевой контейнер содержит служебную информацию, необходимую для обеспечения криптографической защиты ключей, их целостности и т.п.

Каждый ключевой контейнер (независимо от типа носителя), является самодостаточным и содержит всю необходимую информацию для работы как с самим контейнером, так и с закрытыми (и соответствующими им открытыми) ключами.

Ключевой контейнер содержит следующую информацию:

- первичный ключ;
- маски первичного ключа;
- контрольную информацию первичного ключа;
- вторичный ключ (опциональный).

Каждый закрытый ключ хранится в формате, дополнительно содержащем все константы, необходимые для формирования и экспорта открытого ключа.

Формат ключевого контейнера обеспечивает чтение ключей и соответствующих масок отдельными операциями в раздельные (разнесенные по адресам) области памяти, для чего он содержит шесть зон (реализация зон зависит от типа ключевого носителя).

Ключевой контейнер содержит также дополнительную информацию, необходимую для обеспечения его восстановления при возникновении различных программно-аппаратных сбоев (дополнительная информация включается в тех случаях, когда размер ключевого контейнера не ограничен размерами памяти физического носителя).

3.5 Сертификаты ключа

Сертификат открытого ключа обмена и сертификат ключа проверки ЭП представляет собой структурированную двоичную запись в формате ASN.1, содержащую:

- имя субъекта или объекта системы, однозначно идентифицирующее его в системе;
- открытый ключ или ключ проверки ЭП субъекта или объекта системы;
- дополнительные атрибуты, определяемые требованиями использования сертификата в системе;
- ЭП Издателя (Удостоверяющего центра), заверяющую совокупность этих данных.

Формат сертификата определен в рекомендациях ITU-T X.509, IETF RFC 5280 и рекомендациях ТК26 «Р 1323565.1.023-2018. Использование алгоритмов ГОСТ Р 34.10-2012, ГОСТ Р 34.11-2012 в сертификате, списке аннулированных сертификатов (CRL) и запросе на

сертификат PKCS #10 инфраструктуры открытых ключей X.509». В настоящее время основным принятым форматом является формат версии 3, позволяющий определить дополнения (extensions), с помощью которых реализуется определенная политика безопасности в системе.

3.6 Размеры и сроки действия ключей

Длины ключей электронной подписи:

ключ электронной подписи	256 или 512 бит
ключ проверки электронной подписи	512 или 1024 бит

Длины ключей, используемых при шифровании:

закрытый ключ	256 бит или 512 бит
открытый ключ	512 бит или 1024 бита
симметричный ключ	256 бит

При эксплуатации СКЗИ «КриптоПро CSP» версия 5.0 КС1 должны соблюдаться следующие сроки использования пользовательских закрытых ключей и сертификатов:

- максимальный срок действия ключа ЭП — 1 год 3 месяца;
- максимальный срок действия ключа проверки ЭП — 15 лет после окончания срока действия соответствующего закрытого ключа;
- максимальный срок действия закрытых и открытых ключей обмена — 1 год 3 месяца;
- максимальный срок действия ключей ЭП/закрытых ключей обмена в неэкспортируемом виде при их использовании в ФКН и Облачном провайдере — 3 года.

При формировании закрытого ключа в контейнер записывается дата истечения срока действия этого ключа, по истечении которого в зависимости от значения параметра ControlKeyTimeValidity возможны различные варианты использования этого ключа.

Значение «0» параметра не накладывает ограничений на использование ключа.

Значение «1» параметра запрещает формирование ЭП и шифрование в контексте этого ключа (возможно расшифрование ранее зашифрованных сообщений) (значение по умолчанию);

Значение «2» параметра запрещает любые действия с закрытым ключом.

Срок действия ключа берется из (в порядке уменьшения приоритета):

- Расширения контейнера ключа;
- Расширения сертификата ключа;
- Даты создания ключа + 1 год 3 месяца для пассивных ключевых носителей и + 3 года для неэкспортируемых ключей в ФКН и в Облачном провайдере.

3.6.1 Изменение параметра ControlKeyTimeValidity

Для ОС семейства Windows необходимо изменить значение ключа реестра:

HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Crypto Pro\Cryptography\CurrentVersion\Parameters\ControlKeyTimeValidity (для 64-битных операционных систем),

HKEY_LOCAL_MACHINE\SOFTWARE\Crypto Pro\Cryptography\CurrentVersion\Parameters\ControlKeyTimeValidity (для 32-битных операционных систем).

Настройка СКЗИ для остальных ОС осуществляется с помощью утилиты cpconfig с помощью команды:

./cpconfig -ini \config\parameters -add long ControlKeyTimeValidity <значение>

Примечание. При работе в режиме усиленного контроля использования ключей (режим обязателен к использованию, отключение может производиться только в целях тестирования) значение параметра ControlKeyTimeValidity принимается равным «2».

3.7 Хранение ключевых носителей

При хранении ключей необходимо обеспечить невозможность доступа к ключевым носителям не допущенных к ним лиц. Пользователь несет персональную ответственность за хранение личных ключевых носителей.

Запрещается оставлять без контроля вычислительные средства с установленным СКЗИ после ввода ключевой информации.

В случае централизованного хранения ключевых носителей в организации, эксплуатирующей СКЗИ, администратор безопасности (если он имеется) несет персональную ответственность за хранение личных ключевых носителей пользователей.

Хранение ключей на несъемных носителях (в реестре ОС Windows, в разделе HDD/SSD ПЭВМ, на устройстве Apple iOS/Android/SailfishOS и т.п.) допускается только при условии распространения на носитель требований по обращению с ключевыми носителями (в том числе и после удаления ключей).

В случае невозможности отчуждения ключевого носителя с ключевой информацией от ПЭВМ организационно-техническими мероприятиями должен быть исключен доступ нарушителей к ПЭВМ с ключами.

При хранении ключей необходимо использовать парольную защиту, если не оговорено иное.

СКЗИ может функционировать и хранить ключевую информацию в двух режимах:

- в памяти приложения;
- в «Службе хранения ключей», реализованной в виде системного сервиса.

Ключи находятся в кэше до завершения приложения или до выключения компьютера (остановки службы), что позволяет использовать закрытый ключ даже после закрытия криптографического контекста (только для пассивных хранилищ ключевой информации без использования криптографических механизмов, реализованных на смарт-карте/токене).

Функционирование и хранение ключей СКЗИ «КриптоПро CSP» версии 5.0 КС1 в «Службе хранения ключей» обеспечивает дополнительную защиту ключевой информации от других приложений, выполняющихся на ПЭВМ, но может незначительно замедлить производительность системы.

В случае необходимости проведения ремонтных и регламентных работ аппаратной части СФ необходимо обеспечить невозможность доступа нарушителя к ключевой информации, содержащейся в СФ. Конкретный перечень мер должен быть определен исходя из условий эксплуатации СКЗИ.

3.8 Взаимодействие с пользователем при работе с ключевыми носителями

При работе с ключевыми носителями СКЗИ может использовать какой-либо пользовательский интерфейс (UI). Это может происходить, например, при необходимости выбрать носитель или ввести PIN. Чтобы отключить пользовательский интерфейс (например, для автоматизации), в некоторых приложениях существует опция -silent. Также возможно запретить СКЗИ отображать пользовательский интерфейс глобально для всех приложений на данном ПК. Для этого в настройках СКЗИ нужно задать параметр force_silent равным единице (см. ниже), force_silent равный нулю вернёт поведение по умолчанию. Если же вызовы функций требуют отображения пользовательского интерфейса, будет возвращена ошибка NTE_SILENT_CONTEXT.

Изменение параметра force_silent

Для операционных систем группы Windows необходимо изменить значение ключа реестра

HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Crypto Pro\Cryptography\CurrentVersion\Parameters\force_silent (для 64-битных операционных систем),

HKEY_LOCAL_MACHINE\SOFTWARE\Crypto Pro\Cryptography\CurrentVersion\Parameters\force_silent (для 32-битных операционных систем).

Настройка СКЗИ для остальных ОС осуществляется с помощью утилиты cpconfig:

./cpconfig -ini '\config\parameters' -add long force_silent 1

3.8.1 Счетчики операций при использовании ФКН

В носителях ФКН используются три декрементируемых счетчика аутентификации:

- C1 — общий счетчик попыток аутентификации;
- C2 — общий счетчик неуспешных попыток аутентификации;
- C3 — счетчик последовательных неуспешных попыток аутентификации.

Начальное значение счетчиков определяется производителем носителя. При успешной аутентификации счетчик C3 восстанавливается в значение по умолчанию. Если любой из счетчиков достигнет нулевого значения, исполнение защищенных операций на носителе блокируется.

Для восстановления счетчиков в начальное значение требуется сменить пароль.

СКЗИ «КриптоПро CSP» следит за данными счетчиками и заранее предупреждает пользователя о потенциальной блокировке карты при дальнейших неуспешных попытках аутентификации и предлагает заранее сменить пароль.

3.9 Управление ключами

Ключевая система СКЗИ КриптоПро CSP версии 5.0 КС1 базируется на архитектуре PKI рекомендаций X.509. Формирование и управление сертификатами открытых ключей производится УЦ. В качестве УЦ может выступать Удостоверяющий центр «КриптоПро УЦ» или другие сертифицированные ФСБ России УЦ, обеспечивающие выполнение функций доверенного обращения с сертификатами. Также допускается использование Центра Сертификации корпорации Microsoft (Microsoft Certification Authority) в тестовых целях.

СКЗИ КриптоПро CSP версии 5.0 КС1 может использоваться в качестве криптоядра в составе различных прикладных систем, организационные схемы управления ключевой системой которых могут отличаться от рассматриваемой.

Взаимодействие с компонентами УЦ при управлении ключами должно осуществляться в соответствии с Регламентом УЦ.

3.9.1 Формирование ключей

Формирование ключей пользователя производится с использованием функции CPGenKey (см. ЖТЯИ.00101-01 96 01 КриптоПро CSP. Руководство программиста) и спецификацией типа формируемого ключа: AT_KEYEXCHANGE, AT_SIGNATURE, AT_SYMMETRIC.

Формирование ключей возможно, если:

– контекст криптопровайдера «КриптоПро CSP» открыт функцией CPAcquireContext с флагом CRYPT_NEWKEYSET и несуществующим именем ключевого контейнера, специфицированным параметром pszContainer;

– контекст криптопровайдера «КриптоПро CSP» открыт функцией CPAcquireContext с указанием ранее созданного ключевого контейнера, специфицированного параметром pszContainer.

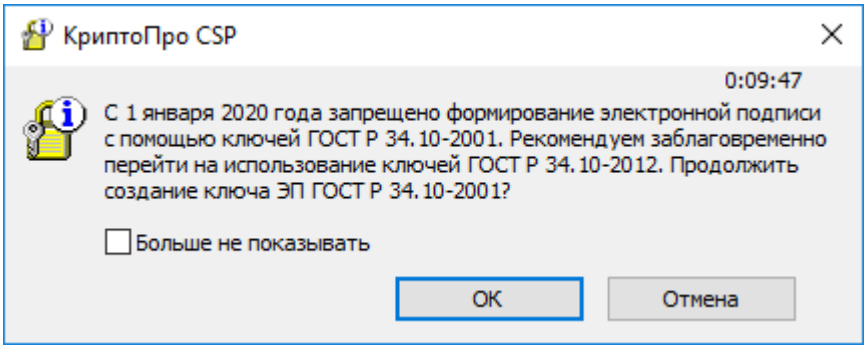
Примечание.

1) Закрытые ключи ЭП и обмена формируются с использованием ПДСЧ с инициализацией его от БиоДСЧ, от внешней гаммы или от физического ДСЧ встраиваемого ПАК защиты от НСД.

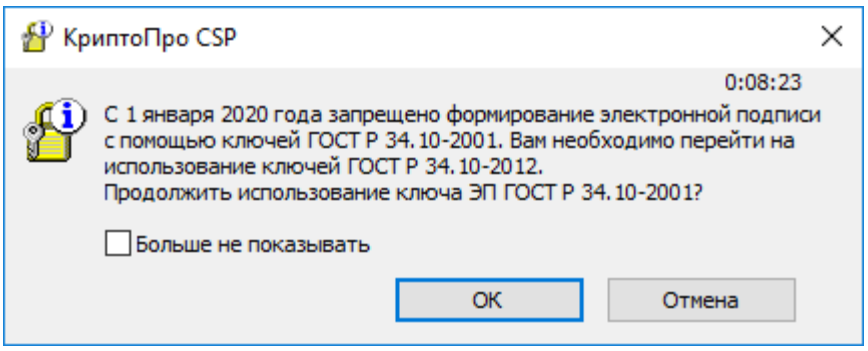
2) При использовании считывателей смарт-карт необходимо проведение проверки настройки используемых ими портов ПЭВМ в BIOS и ОС.

3) При использовании НГМД в качестве ключевого носителя во избежание потери ключевой информации рекомендуется хранить ее копию.

В связи с переходом на использование алгоритма ГОСТ Р 34.10-2012 и соответствующем запрете использования алгоритма ГОСТ Р 34.10-2001, при попытке генерации ключа алгоритма ГОСТ Р 34.10-2001 будет выдано следующее предупреждение:



При попытке создания подписи с использованием ключа алгоритма ГОСТ Р 34.10-2001 будет выдано следующее предупреждение:



Примечание. При использовании операционных систем, отличных от ОС семейства Windows, окно предупреждения может выглядеть иначе, но текстовая составляющая аналогична приведенной рисунках выше.

При установке флага «Больше не показывать» предупреждения о генерации ключа и создании подписи будут отложены до 01 января 2020 года. При повторном выборе «Больше не показывать» предупреждения более появляться не будут.

Создание подписи с использованием ключа алгоритма ГОСТ Р 34.10-2001 с 01 января 2020 года запрещено.

3.9.2 Компрометация ключей пользователя

К событиям, связанным с компрометацией ключей относятся, в частности, следующие:

- потеря ключевых носителей;
- потеря ключевых носителей с их последующим обнаружением;
- увольнение сотрудников, имевших доступ к ключевой информации;
- нарушение правил хранения и уничтожения (после окончания срока действия) закрытого ключа;
- возникновение подозрений на утечку информации или ее искажение в системе конфиденциальной связи;
- нарушение печати на сейфе с ключевыми носителями;
- случаи, когда нельзя достоверно установить, что произошло с ключевыми носителями (в том числе случаи, когда ключевой носитель вышел из строя и доказательно не опровергнута возможность того, что, данный факт произошел в результате несанкционированных действий злоумышленника).

Различаются два вида компрометации закрытого ключа: явная и неявная. Первые четыре события трактуются как явная компрометация ключей. Следующие три требуют специального рассмотрения в каждом конкретном случае.

При компрометации своего ключа пользователь должен немедленно прекратить связь по сети с другими пользователями. При этом пользователь (или администратор безопасности организации) должен немедленно известить ЦР (УЦ) о компрометации ключа пользователя.

После компрометации ключей пользователь формирует новый закрытый ключ и запрос на сертификат. Так как пользователь не может использовать скомпрометированный ключ для формирования ЭП и передачи запроса в защищенном виде по сети, запрос на сертификат вместе с бланками доставляется лично пользователем (администратором безопасности) в Центр Регистрации.

3.9.3 Уничтожение ключей на ключевых носителях

Ключевые носители с ключом ЭП/закрытым ключом, срок действия которого истек, уничтожаются путем переформатирования (очистки) ключевых носителей средствами СКЗИ, после чего ключевые носители могут использоваться для записи на них новой ключевой информации.

4 Требования по встраиванию и использованию ПО СКЗИ

Встраивание СКЗИ в защищаемые информационные системы должно производиться в соответствии с Положением ПКЗ-2005. Встраивание должны проводить организации, имеющие лицензию на право проведения таких работ.

При создании защищенной информационной системы должны быть определены модель возможных угроз и политика безопасности. В зависимости от политики безопасности определяется необходимый набор криптографических функций и организационно-технических мер, реализуемых в создаваемой системе.

Защита от закладок, вирусов, модификации системного и прикладного ПО должна быть обеспечена использованием средств антивирусной защиты и организационных мероприятий.

При встраивании СКЗИ КриптоПро CSP в прикладное программное обеспечение или использовании его в составе стандартного прикладного ПО должны выполняться следующие требования:

1. При использовании открытого ключа или ключа проверки ЭП должны быть обеспечены его авторизация, достоверность, целостность и идентичность с помощью процедур Удостоверяющего центра.
2. При использовании сертификатов открытых ключей и ключей проверки ЭП, заверенных подписью доверенной стороны, должна быть обеспечена безопасная доставка и хранение сертификата ключа доверенной стороны, с использованием которого проверяются остальные сертификаты ключей проверки ЭП пользователей.
3. Криптографическое средство, с помощью которого производится заверение ключей проверки ЭП, открытых ключей или справочников открытых ключей, должно быть сертифицировано по классу, соответствующему принятой политике безопасности.
4. Для отзыва (вывода из действия) открытых ключей и ключей проверки ЭП должны использоваться средства, позволяющие произвести авторизацию отзывающего лица (в этих целях может быть использован список отозванных сертификатов, заверенный ЭП доверенной стороны).
5. При вызове приложением функций СКЗИ в прикладном программном обеспечении должна быть предусмотрена проверка кода завершения вызываемой функции.
6. При вызове функций СКЗИ в обязательном порядке требуется указывать имя и тип провайдера. Использование провайдера по умолчанию не допускается.
7. При встраивании СКЗИ в средство ЭП в функции проверки подписи с помощью сертификата ключа проверки ЭП должна быть исключена возможность проверки ЭП без предварительной проверки ЭП в сертификате.
8. Для ограничения возможности влияния аппаратных компонентов СВТ на функционирование СКЗИ необходимо проведение исследований ПО BIOS СВТ, на которых установлено СКЗИ, на соответствие действующим требованиям ФСБ России по исследованию ПО BIOS СВТ.

При использовании СКЗИ под управлением ОС Android должны выполняться следующие условия:

- необходимо использовать ОС Android версий 8.0 и выше;
- должно быть запрещено использование Accessibility Service;
- мобильное устройство не должно иметь root прав;
- приложения должны функционировать на мобильном устройстве с установленным средством антивирусной защиты;
- обязательна установка всех патчей обновлений;
- должна быть запрещена установка ПО из недоверенного источника.

5 Требования по защите от НСД

5.1 Общие требования по организации работы по защите от НСД

Защита аппаратного и программного обеспечения от НСД при установке и использовании СКЗИ является составной частью общей задачи обеспечения безопасности информации в системе, в состав которой входит СКЗИ.

Наряду с применением средств защиты от НСД необходимо выполнение приведенных ниже организационно-технических и административных мер по обеспечению правильного функционирования средств обработки и передачи информации, а также установление соответствующих правил для обслуживающего персонала, допущенного к работе с конфиденциальной информацией.

Защита информации от НСД должна обеспечиваться на всех технологических этапах обработки информации и во всех режимах функционирования, в том числе при проведении ремонтных и регламентных работ.

Защита информации от НСД должна предусматривать контроль эффективности средств защиты от НСД. Этот контроль должен периодически выполняться администратором безопасности на основе требований документации на средства защиты от НСД.

В организации, эксплуатирующей СКЗИ, должен быть назначен администратор безопасности, на которого возлагаются задачи организации работ по использованию СКЗИ, выработки соответствующих инструкций для пользователей, а также контроль за соблюдением требований по безопасности.

Администратор безопасности не должен иметь возможности доступа к конфиденциальной информации пользователей.

Правом доступа к рабочим местам с установленными СКЗИ должны обладать только определенные для эксплуатации лица, прошедшие соответствующую подготовку. Администратор безопасности должен ознакомить каждого пользователя, применяющего СКЗИ, с документацией на СКЗИ, а также с другими нормативными документами, созданными на её основе.

5.2 Требования по размещению технических средств с установленным СКЗИ

При размещении технических средств с установленным СКЗИ:

– Должны быть приняты меры по исключению несанкционированного доступа в помещения, в которых размещены технические средства с установленным СКЗИ, посторонних лиц, по роду своей деятельности не являющихся персоналом, допущенным к работе в этих помещениях. В случае необходимости присутствия посторонних лиц в указанных помещениях должен быть обеспечен контроль за их действиями и обеспечена невозможность негативных действий с их стороны на СКЗИ, технические средства, на которых эксплуатируется СКЗИ и защищаемую информацию.

– Должны быть приняты меры, исключающие несанкционированное вскрытие корпусов и средств, входящих в состав СКЗИ.

– Внутренняя планировка, расположение и укомплектованность рабочих мест в помещениях должны обеспечивать исполнителям работ, сохранность доверенных им конфиденциальных документов и сведений, включая ключевую информацию.

– Размещение СКЗИ «КриптоПро CSP» версии 5.0 КС1 в помещениях, в которых осуществляется обработка информации, содержащей сведения, составляющие государственную тайну, осуществляется установленным порядком.

В целях защиты открытой конфиденциальной информации от утечки по техническим каналам, в том числе по каналам связи, от объектов информатизации и СКЗИ, ввод в действие и эксплуатация указанных объектов и СКЗИ должна осуществляться в соответствии с действующими в Российской Федерации требованиями по защите информации от утечки по техническим каналам, в том числе по каналу связи (например, СТР-К).

Необходимость и достаточность мер, в том числе по каналу связи, должна оцениваться порядком, предусмотренным упомянутыми руководящими документами, с учетом целевых установок предполагаемого нарушителя и угроз безопасности информации, определяемых моделью угроз и нарушителя. При этом, если объекты аттестованы на соответствие установленным требованиям по защите информации без учета оценки канала связи, при подключении таких средств к каналам связи, выходящим за пределы контролируемой территории, необходимо использовать любое из следующих средств:

– Волоконно-оптические линии связи;

– Оптические развязывающие устройства, устанавливаемые в тракт передачи информации для создания оптоволоконного фрагмента сети;

– Сертифицированные СКЗИ для передачи информации соответствующего уровня конфиденциальности.

Для обеспечения защиты информации по классу КС от утечки по каналу линейной передачи достаточно, чтобы канал связи был реализован в виде радиоканала GSM, либо GPRS, либо 3G/4G, либо Wi-Fi, либо другого канала мобильной и беспроводной связи, работающего в диапазоне частот несущей выше 800 МГц с цифровой модуляцией штатного информационного сигнала.

5.3 Требования по установке СКЗИ, общесистемного и специального ПО на ПЭВМ

1. ПЭВМ, на которых используется СКЗИ, должны быть допущены для обработки конфиденциальной информации по действующим в Российской Федерации требованиям по защите информации от утечки по техническим каналам, в том числе, по каналу связи (например, СТР-К), с учетом модели угроз в информационной системе заказчика, которым должно противостоять СКЗИ «КриптоПро CSP» версии 5.0 КС1.

2. Инсталляция СКЗИ «КриптоПро CSP» версии 5.0 КС1 на рабочих местах должна производиться только с дистрибутива, полученного по доверенному каналу.

3. К установке общесистемного и специального программного обеспечения, а также СКЗИ «КриптоПро CSP» версии 5.0 КС1, допускаются лица, прошедшие соответствующую подготовку и изучившие документацию на соответствующее ПО и на СКЗИ.

4. На технических средствах, предназначенных для работы с СКЗИ, использовать только лицензионное программное обеспечение фирм-изготовителей.

5. При установке ПО СКЗИ на ПЭВМ должен быть обеспечен контроль целостности и достоверность дистрибутива СКЗИ и совместно поставляемых с СКЗИ компонент СФ (см. Приложение 1).

6. На ПЭВМ не должны устанавливаться средства разработки ПО и отладчики. Если средства отладки приложений нужны для технологических потребностей организации, то их использование должно быть санкционировано администратором безопасности. При этом должны быть реализованы меры, исключающие возможность использования этих средств для редактирования кода и памяти СКЗИ и приложений, использующих СКЗИ, а также для

просмотра кода и памяти СКЗИ и приложений, использующих СКЗИ, в процессе обработки СКЗИ защищаемой информации и/или при загруженной ключевой информации.

7. При установке программного обеспечения СКЗИ «КриптоПро CSP» версии 5.0 КС1 следует:

– Предусмотреть меры, исключающие возможность несанкционированного не обнаруживаемого изменения аппаратной части технических средств, на которых установлены СКЗИ (например, путем опечатывания системного блока и разъемов ПЭВМ).

– После завершения процесса установки должны быть выполнены действия, необходимые для осуществления периодического контроля целостности установленного ПО СКЗИ, а также его окружения в соответствии с документацией (см. Приложение 1).

– Программное обеспечение, устанавливаемое на ПЭВМ с СКЗИ, не должно содержать возможностей, позволяющих:

- модифицировать содержимое произвольных областей памяти;
- модифицировать собственный код и код других подпрограмм;
- модифицировать память, выделенную для других подпрограмм;
- передавать управление в область собственных данных и данных других подпрограмм;
- несанкционированно модифицировать файлы, содержащие исполняемые коды при их хранении на жестком диске;
- повышать предоставленные привилегии;
- модифицировать настройки ОС;
- использовать недокументированные фирмой-разработчиком функции ОС.

5.4 Меры по обеспечению защиты от НСД

При использовании СКЗИ должны выполняться следующие меры по защите информации от НСД:

– необходимо разработать и применить политику назначения и смены паролей (для входа в ОС, BIOS, при шифровании на пароле и т.д.).

– необходимо использовать фильтры паролей в соответствии со следующими правилами:

- длина пароля должна быть не менее 8 символов;
- в числе символов пароля обязательно должны присутствовать буквы в верхнем и нижнем регистрах, цифры и специальные символы (@, #, $, &, *, % и т.п.);
- пароль не должен включать в себя легко вычисляемые сочетания символов (имена, фамилии и т.д.), а также общепринятые сокращения (USER, ADMIN, ALEX и т.д.);
- при смене пароля новое значение должно отличаться от предыдущего не менее чем в 4-х позициях;
- личный пароль пользователь не имеет права сообщать никому;
- периодичность смены пароля определяется принятой политикой безопасности, но не должна превышать 6 месяцев.

– пароли для аутентификации пользователей на носителях, работающих в режиме активного вычислителя с защитой канала между носителем и СКЗИ по протоколу SESPAKE, должны удовлетворять следующим требованиям:

- для выработки общего ключа с аутентификацией на основе пароля при использовании кривых с 512-битовым порядком группы точек:
  - пароль должен состоять из букв английского алфавита и верхнем и нижнем регистрах, цифр и спецзнаков;

- минимальная длина пароля — 4 символа;
- периодичность смены пароля — не реже 1 раза в полгода.

- для выработки общего ключа с аутентификацией на основе пароля при использовании кривых с 256-битовым порядком группы точек:
  - пароль должен состоять из букв английского алфавита и верхнем и нижнем регистрах, цифр и спецзнаков;
  - минимальная длина пароля — 10 символов;
  - периодичность смены пароля — не реже 1 раза в 56 дней.

– политика назначения и смены паролей должна удовлетворять следующим требованиям: после трех неверных попыток ввода пароля пользователем при входе в ОС система должна блокироваться на 1 час (с обеспечением возможности разблокировки учетной записи при обращении пользователя к администратору безопасности).

– указанная политика обязательна для всех учетных записей, зарегистрированных в ОС.

– средствами BIOS должна быть исключена возможность работы на ПЭВМ с СКЗИ, если во время её начальной загрузки не проходят встроенные тесты.

– администратор безопасности (если он имеется) несет персональную ответственность за хранение личных ключевых носителей пользователей.

При эксплуатации СКЗИ **запрещено**:

– оставлять без контроля вычислительные средства, на которых эксплуатируется СКЗИ (в том числе смарт-карты), после ввода ключевой информации либо иной конфиденциальной информации;

– вносить какие-либо изменения в программное обеспечение СКЗИ;

– осуществлять несанкционированное администратором безопасности копирование ключевых носителей;

– разглашать содержимое носителей ключевой информации или передавать сами носители лицам, к ним не допущенным, выводить ключевую информацию на дисплей, принтер иные средства отображения информации;

– использовать ключевые носители в режимах, не предусмотренных функционированием СКЗИ;

– записывать на ключевые носители постороннюю информацию.

Администратор безопасности должен сконфигурировать операционную систему, в среде которой планируется использовать СКЗИ, и осуществлять периодический контроль сделанных настроек в соответствии со следующими требованиями:

– не использовать нестандартные, измененные или отладочные версии ОС;

– исключить возможность загрузки и использования ОС, отличной от предусмотренной штатной работой;

– исключить возможность удаленного управления, администрирования и модификации ОС и её настроек;

– на ПЭВМ должна быть установлена только одна операционная система;

– правом установки и настройки ОС и СКЗИ должен обладать только администратор безопасности;

– все неиспользуемые ресурсы системы необходимо отключить (протоколы, сервисы и т.п.);

– режимы безопасности, реализованные в ОС, должны быть настроены на максимальный уровень;

– всем пользователям и группам, зарегистрированным в ОС, необходимо назначить минимально возможные для нормальной работы права;

– необходимо предусмотреть меры, максимально ограничивающие доступ к следующим ресурсам системы (в соответствующих условиях возможно полное удаление ресурса или его неиспользуемой части):
  - системный реестр;
  - файлы и каталоги;
  - временные файлы;
  - журналы системы;
  - файлы подкачки;
  - кэшируемая информация (пароли и т.п.);
  - отладочная информация.

– кроме того, необходимо организовать стирание (по окончании сеанса работы СКЗИ) временных файлов и файлов подкачки, формируемых или модифицируемых в процессе работы СКЗИ. Если это не выполнимо, то на жесткий диск должны распространяться требования, предъявляемые к ключевым носителям;

– должно быть исключено попадание в систему программ, позволяющих повышать предоставленные привилегии за счет ошибок в ОС;

– необходимо регулярно устанавливать пакеты обновления безопасности ОС (Service Packs, Hot fix и т.п.), обновлять антивирусные базы, а также исследовать информационные ресурсы по вопросам компьютерной безопасности с целью своевременной минимизации опасных последствий от возможного воздействия на ОС;

– в случае подключения ПЭВМ с установленным СКЗИ к общедоступным сетям передачи данных, необходимо исключить возможность открытия и исполнения файлов и скриптовых объектов (например, JavaScript, VBScript, ActiveX), полученных из общедоступных сетей передачи данных без проведения соответствующих проверок на предмет содержания в них программных закладок и вредоносного ПО, загружаемых из сети;

– при использовании СКЗИ на ПЭВМ, подключенных к общедоступным сетям связи, с целью исключения возможности несанкционированного доступа к системным ресурсам используемых операционных систем, к программному обеспечению, в окружении которого функционируют СКЗИ, и к компонентам СКЗИ со стороны указанных сетей должны использоваться дополнительные методы и средства защиты (например, установка межсетевых экранов, организация VPN сетей и т.п.). При этом предпочтение должно отдаваться средствам защиты, имеющим сертификат уполномоченного органа по сертификации;

– организовать и использовать систему аудита, организовать регулярный анализ результатов аудита;

– организовать и использовать комплекс мероприятий антивирусной защиты;

– должно быть запрещено использование СКЗИ для защиты речевой информации;

– должны быть отключены радиоканалы, неиспользуемые в работе СКЗИ.

Запрещено:

– использовать ключи с истекшим сроком действия;

– осуществлять несанкционированное копирование ключевых носителей;

– разглашать содержимое носителей ключевой информации или передавать сами носители лицам, к ним не допущенным, выводить ключевую информацию на дисплей и принтер (за исключением случаев, предусмотренных данными правилами);

– вставлять ключевой носитель в устройство считывания в режимах, не предусмотренных штатным режимом использования ключевого носителя;

– подключать к ПЭВМ дополнительные устройства и соединители, не предусмотренные штатной комплектацией;

– работать на ПЭВМ, если во время его начальной загрузки не проходит встроенный тест ОЗУ, предусмотренный в ПЭВМ;

- вносить какие-либо изменения в программное обеспечение СКЗИ;
- изменять настройки, установленные программой установки СКЗИ или администратором;
- использовать синхропосылки, вырабатываемые не средствами СКЗИ;
- обрабатывать на ПЭВМ, оснащенной СКЗИ, информацию, содержащую государственную тайну, без проведения исследований по обоснованию невозможности обработки СКЗИ информации, содержащей государственную тайну;
- использовать бывшие в работе ключевые носители для записи новой информации без предварительного уничтожения на них ключевой информации, подлежащей уничтожению в соответствии с п. 3.9.3, средствами СКЗИ;
- осуществлять несанкционированное вскрытие системных блоков ПЭВМ;
- приносить и использовать в помещении, где размещены средства СКЗИ, радиотелефоны и другую радиопередающую аппаратуру (требование носит рекомендательный характер).

Аппаратуру, на которой устанавливается СКЗИ, рекомендуется проверить на отсутствие аппаратных закладок.

5.5 Действия при компрометации ключей

В случае компрометации ключей по факту компрометации должно быть проведено служебное расследование. Скомпрометированные ключи выводятся из действия.

Выведенные из действия скомпрометированные ключевые носители после проведения служебного расследования уничтожаются, о чем делается запись в «Журнале пользователя сети».

Скомпрометированные ключи подлежат замене.

Подробные действия при компрометации ключей описаны в п. 3.9.2.

5.6 Требования по подключению СКЗИ для работы по общедоступным каналам передачи данных

Порядок подключения СКЗИ к каналам связи должен быть определен эксплуатирующей организацией. Лицом, ответственным за безопасность работы СКЗИ по общедоступным каналам, как правило, должен быть администратор безопасности.

При подключении СКЗИ к общедоступным каналам передачи данных должна быть обеспечена безопасность защищенной связи. При этом должны быть определены:

- порядок подключения СКЗИ к каналам;
- лицо, ответственное за безопасность работы по общедоступным каналам;
- типовой регламент защищенной связи, включающий:
    - политику безопасности защищенной связи;
    - допустимый состав прикладных программных средств, для которого должно быть исследовано и обосновано отсутствие негативного влияния на СКЗИ по каналу передачи данных;
    - перечень допустимых сетевых протоколов;
    - защиту сетевых соединений (перечень допустимых сетевых экранов);
    - система и средства антивирусной защиты.

Перечень стандартных средств ОС может включаться администратором в типовой регламент без проведения дополнительных исследований по оценке их влияния на СКЗИ. При этом должны выполняться:

- своевременное обновление программных средств, включенных в состав регламента;
- контроль среды функционирования СКЗИ;
- определение и контроль за использованием сетевых протоколов;
- соблюдение правил пользования СКЗИ и средой функционирования СКЗИ.

Должен быть обеспечен организационно-технический контроль запросов на установление соединения абонентов по протоколу TLS с использованием эфемерных ключей, исключающих возможность использования абонентом не своих атрибутов соединения (такие, как Client_Id и т.п.).

При использовании СКЗИ с другими стандартными программными средствами возможность подключения СКЗИ к общедоступным каналам передачи данных должна быть возможна только после проведения дополнительных исследований с оценкой негативного влияния нарушителя на функционирование СКЗИ, использующего средства общедоступных каналов.

При установке параметров, позволяющих создавать соединения, отличные от криптографически защищенных, в соответствии с настоящими правилами (TLS-соединение на основе сертификатов ключей ГОСТ Р 34.10-2001, ГОСТ Р 34.10-2012 (ГОСТ 34.10-2018)), должны быть приняты меры, исключающие утечку требующей защиты информации с защищаемого объекта информации. Проверка достаточности принятых мер защиты проводится при аттестации объекта информатизации с СКЗИ по требованиям информационной безопасности.

5.7 Требования по использованию в СКЗИ программно-аппаратных средств защиты от НСД

Программно-аппаратные средства защиты от НСД предназначены для организации защиты компьютера от входа посторонних пользователей. Под посторонними пользователями понимаются все лица, не зарегистрированные в системе как пользователи данного компьютера.

Допускается использовать средства защиты от несанкционированного доступа, сертифицированные ФСБ России.

Программно-аппаратные средства защиты от НСД обеспечивают:

- регистрацию пользователей компьютера и назначение им персональных идентификаторов и паролей на вход в систему;
- идентификацию и аутентификацию пользователя при загрузке компьютера;
- ведение системного журнала, в котором производится регистрация событий, имеющих отношение к безопасности системы;
- контроль целостности программного и аппаратного обеспечения защищаемого компьютера;
- защиту от несанкционированной загрузки операционной системы со съемных носителей;
- наличие в составе аппаратного датчика случайных чисел (ДСЧ).

Установка и настройка программно-аппаратного средства защиты от НСД на АРМ пользователя должна производиться в соответствии с эксплуатационной документацией. Перед эксплуатацией программно-аппаратного средства защиты от НСД в составе АРМ пользователя необходимо ознакомиться с комплектом документации на данное средство и принять рекомендуемые в документации защитные организационные меры.

Настройка программно-аппаратного средства защиты от НСД на требуемую конфигурацию выполняется администратором безопасности. Настройка должна исключать

возможность вмешательства пользователя в процессы загрузки операционной системы и прикладного ПО и проверки целостности программной среды.

6 Использование программных интерфейсов

Разработка программного обеспечения на основе СКЗИ «КриптоПро CSP» с учетом п.1.5 Формуляра ЖТЯИ.00101-01 30 01 может производиться без создания новых СКЗИ в случае использования вызовов функций интерфейсов CryptoAPI (JCA/JCE) СКЗИ из перечня Приложения 2. Данные вызовы функций могут использоваться как напрямую, так и опосредованно через вызовы промежуточных функций интерфейсов.

В случае опосредованного использования в программном обеспечении вызовов функций интерфейсов CryptoAPI (JCA/JCE), техническое задание на проведение работ по оценке влияния такого программного обеспечения на СКЗИ необходимо согласовывать, в том числе, с ООО «КРИПТО-ПРО».

В случае использования (напрямую или опосредованно) в программном обеспечении прочих вызовов интерфейсов CryptoAPI (JCA/JCE) СКЗИ необходимо производить разработку отдельного СКЗИ в соответствии с действующей нормативной базой (в частности, с Постановлением Правительства Российской Федерации от 16 апреля 2012 г. №313 и Положением о разработке, производстве, реализации и эксплуатации шифровальных (криптографических) средств защиты информации (Положение ПКЗ-2005)).

Приложение 1. Контроль целостности программного обеспечения

Контроль целостности программного обеспечения с помощью алгоритмов хэширования

Модуль cpverify.exe позволяет осуществлять контроль целостности установленного программного обеспечения. Контроль целостности файлов осуществляется при загрузке файла на исполнение (и периодически во время выполнения) или при ручном запуске программы контроля целостности (см. опцию –rv ниже).

При помощи перечисленных ниже опций модуль cpverify.exe может быть использован для следующих контрольных целей:

cpverify -mk filename [-alg algid] [-inverted_halfbytes <inv>] – вычисление значения хэш-функции для файла с именем filename с помощью алгоритма algid. Поле algid может принимать значения GR3411, GR3411_2012_256 и GR3411_2012_512 (по умолчанию используется GR3411). [-inverted_halfbytes <inv>] указывается, если полубайты в hashvalue перевернуты (по умолчанию для GR3411 inv принимается равным «1», для алгоритмов GR3411_2012_256 и GR3411_2012_512 «0»).

cpverify filename [-alg algid] [hashvalue] [-inverted_halfbytes <inv>] – проверка целостности файла с именем filename, используя алгоритм algid и хэш-значение hashvalue. Поле algid может принимать значения GR3411, GR3411_2012_256 и GR3411_2012_512 (по умолчанию используется GR3411). Если hashvalue не указан, то хэш-значение берется из файла filename.hsh. [-inverted_halfbytes <inv>] указывается, если полубайты в hashvalue перевернуты (по умолчанию для GR3411 inv принимается равным «1», для алгоритмов GR3411_2012_256 и GR3411_2012_512 «0»).

cpverify -rm [-alg algid] [catname] – вычисление значения хэш-функции для каждого из файлов, содержащихся в каталоге catname в разделе реестра (если catname не указан, то будут пересчитаны все хэш-значения в разделе реестра). Текущее значение хэш-функций при этом заменяется на вновь посчитанное. Поле algid может принимать значения GR3411, GR3411_2012_256 и GR3411_2012_512 (по умолчанию используется GR3411).

cpverify –rv [catname] – проверка целостности файлов из каталога catname в разделе реестра (если catname не указан, то будут проверены все файлы в разделе реестра).

cpverify -xm in_file out_file [-alg algid] [xmlcatname] – вычисление значения хэш-функции для файлов, перечисленных в xml-файле с именем in_file в каталоге xmlcatname (если xmlcatname не указан, то хэш-значения будут посчитаны для всех фалов, перечисленных в xml-файле с именем in_file), и запись полученных значений в xml-файл с именем out_file. Текущее значение хэш-функций при этом заменяется на вновь посчитанное. Поле algid может принимать значения GR3411, GR3411_2012_256 и GR3411_2012_512 (по умолчанию используется GR3411).

cpverify –xv in_file [xmlcatname] – проверка целостности файлов, перечисленных в xml-файле с именем in_file в каталоге xmlcatname (если xmlcatname не указан, то проверка будет выполнена для всех фалов, перечисленных в xml-файле с именем in_file).

cpverify –r2x out_file – формирование xml-файла с именем out_file, содержащего список файлов, находящихся в разделе реестра под контролем целостности, и хэш-значения этих файлов.

cpverify –x2r in_file – установка под контроль целостности файлов, перечисленных в xml-файле с именем in_file.

Список контролируемых модулей зависит от исполнения и может быть получен при помощи команды cpverify –r2x in_file.

Для того, чтобы поставить под контроль целостности установленное программное обеспечение, нужно выполнить следующую последовательность действий:

1. Создать xml-файл, содержащий список устанавливаемых под контроль целостности файлов. Данный xml-файл должен иметь следующую структуру:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<CProIntegrity>
<catalog>
<entry name="calc.exe">
<Path>C:\WINDOWS\system32\calc.exe</Path>
<Algid>00008021</Algid>
</entry>
<entry name="verifier.exe">
<Path>C:\WINDOWS\system32\verifier.exe</Path>
<Algid>00008021</Algid>
</entry>
</catalog>
</CProIntegrity>
```

Примечание. Значение поля Algid зависит от используемого алгоритма хэширования:

- GR3411 — значение 0000801E
- GR3411_2012_256 — значение 00008021
- GR3411_2012_512 — значение 00008022

2. Запустить модуль cpverify –xm in_file out_file TestControl, указав в качестве параметра in_file имя созданного xml-файла. Результатом работы модуля будет являться xml-файл с именем out_file, содержащий вычисленные значения хэш-функции для перечисленных в in_file файлов и имеющий следующую структуру:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<CProIntegrity>
<catalog>
<entry name="calc.exe">
```

```
<Path>C:\WINDOWS\system32\calc.exe</Path>
<Algid>00008021</Algid>
<Tag>679837307CDC7AA1E4BDBB75194A24D42C782079AF08E2D362D7624A90D604C7</
Tag>
</entry>
<entry name="verifier.exe">
<Path>C:\WINDOWS\system32\verifier.exe</Path>
<Algid>00008021</Algid>
<Tag>9DF987B89A323BEBC3C29BAC0AED42A4F5BD651892AAE79F1EC1D05288D06B9C</
Tag>
</entry>
</catalog>
</CProIntegrity>
```

Примечание. Значение поля Algid зависит от используемого алгоритма хэширования:

- GR3411 — значение 0000801E
- GR3411_2012_256 — значение 00008021
- GR3411_2012_512 — значение 00008022

3. Установить под контроль целостности файлы, для которых было вычислено значение хэш-функции, используя модуль cpverify –x2r in_file TestControl, где параметром in_file является xml-файл, полученный в результате вычисления значения хэш-функции в пункте 2.

Контроль целостности программного обеспечения с помощью алгоритмов подписи

- cpverify -file_verify имя_файла [значение_подписи] -timestamp дата

Проверка подписи файла с именем «имя_файла». Параметр «значение_подписи» необходимо передавать в виде байтовой строки. Если параметр «значение_подписи» не указан, то значение подписи берется из файла имя_файла.sgn. Параметр «дата» указывает, когда подпись была сформирована, необходимо указывать в формате дд.мм.гггг. Данная команда проверяет подпись с прямой последовательностью полубайт, для проверки подписи с обратной последовательностью байт необходимо использовать команду versign с аналогичным набором параметров. Подпись проверяется на открытом ключе из специального сертификата для подписи кода компании «КРИПТО-ПРО».

- cpverify -file_sign имя_файла -cont имя_контейнера [-pin пароль] [-provname имя_провайдера] [-provtype тип_провайдера]

Создание подписи файла с именем «имя_файла» с помощью ключа, взятого из контейнера с именем «имя_контейнера». Параметр password содержит пароль защиты контейнера. Параметры Provname и Provtype указывают, какой провайдер необходимо использовать. Параметр Provtype может принимать значения 75,80 и 81. Если Provtype не указан, то используется 75.

Приложение 2. Перечень вызовов, использование которых при разработке систем на основе СКЗИ «КриптоПро CSP» с учетом п.1.5 Формуляра ЖТЯИ.00101-01 30 01 возможно без дополнительных тематических исследований

Функция	Описание	Ограничения на использование функции
Функции инициализации и настройки провайдера		
CryptAcquireContext	Функция CryptAcquireContext используется для создания дескриптора криптопровайдера с именем ключевого контейнера, определённым параметром pszContainer	Полученный дескриптор криптопровайдера должен быть в обязательном порядке освобождён с помощью вызова функции CryptReleaseContext (за исключением вызовов с флагом CRYPT_DELETEKEYSET).
CryptReleaseContext	Функция CryptReleaseContext используется для удаления дескриптора криптопровайдера, созданного CryptAcquireContext.	Перед вызовом данной функции все дескрипторы объектов ключей и хэширования, работа с которыми производилась совместно с удаляемым дескриптором криптопровайдера, должны быть удалены с помощью вызовов CryptDestroyKey и CryptDestroyHash соответственно.
CryptContextAddRef	Управляет счетчиком дескрипторов созданного CryptAcquireContext.	
CryptEnumProviders	Перечисление установленных криптопровайдеров	
CryptEnumProviderTypes	Перечисление установленных типов криптопровайдеров	
CryptGetDefaultProvider	Получение контекста провайдера, установленного в системе по умолчанию	
CryptGetProvParam	Функция CryptGetProvParam получает параметры криптопровайдера.	
CryptSetProvParam	Функция CryptSetProvParam устанавливает параметры криптопровайдера.	
FreeCryptProvFromCertEx	Функция используется для удаления дескриптора криптопровайдера, созданного	

	CryptAcquireContext или через CNG.	
CryptInstallDefaultContext, CryptSetProvider, CryptSetProviderEx, CryptUninstallDefaultContext	Функции управления контекстом провайдера по умолчанию	
Функции генерации и обмена ключами, создания, конфигурирования и удаления ключей		
CryptGenKey	Функция CryptGenKey генерирует случайные криптографические ключи или ключевую пару (закрытый/открытый ключи).	Полученный дескриптор ключа должен в обязательном порядке быть удалён с помощью вызова функции CryptDestroyKey до вызова функции CryptReleaseContext для рабочего дескриптора криптопровайдера.
CryptDestroyKey	Функция CryptDestroyKey удаляет ключ, передаваемый через параметр hKey. После удаления ключ (дескриптор ключа) не может использоваться.	
CryptExportKey	Функция CryptExportKey используется для экспорта криптографических ключей из ключевого контейнера криптопровайдера, сохраняя их в защищённом виде.	Разрешено экспортировать только открытые ключи (PUBLICKEYBLOB) с нулевым значением флага или со значением флага CRYPT_PUBLICCOMPRESS.
CryptGenRandom	Функция CryptGenRandom заполняет буфер случайными байтами.	
CryptGetKeyParam	Функция CryptGetKeyParam возвращает параметры ключа.	
CryptGetUserKey	Функция CryptGetUserKey возвращает дескриптор одной из долговременных ключевых пар в ключевом контейнере.	Полученный дескриптор ключа должен в обязательном порядке быть удалён с помощью вызова функции CryptDestroyKey до вызова функции CryptReleaseContext для рабочего дескриптора криптопровайдера.
CryptImportKey	Функция CryptImportKey используется для импорта криптографического ключа из ключевого блоба в контейнер криптопровайдера.	Разрешено импортировать только открытые ключи (PUBLICKEYBLOB) при нулевом значении флагов. Полученный дескриптор ключа должен в обязательном порядке быть удалён с помощью вызова функции CryptDestroyKey до вызова функции CryptReleaseContext для

		рабочего дескриптора криптопровайдера.
CryptSetKeyParam	Функция CryptSetKeyParam устанавливает параметры ключа.	Разрешено использование только со следующими символьными аргументами: KP_CERTIFICATE, KP_CIPHEROID, KP_DHOID, KP_HASHOID.
Функции обработки криптографических сообщений		
CryptSignMessage	Функция CryptSignMessage создает хэш определенного содержания, подписывает хэш и затем производит закодирование и текста исходного сообщения, и подписанного хэша	
CryptVerifyMessageSignature	Функция CryptVerifyMessageSignature проверяет электронную подпись сообщения.	
CryptVerifyDetachedMessageSignature	Функция CryptVerifyDetachedMessageSignature проверяет подписанное сообщение, содержащее отсоединенную (detached) подпись или подписи	
CryptDecodeMessage	Функция декодирует, расшифровывает и проверяет сообщение	
CryptDecryptAndVerifyMessageSignature	Функция декодирует и проверяет сообщение	
CryptEncryptMessage	Функция CryptEncryptMessage зашифровывает и производит закодирование сообщения. Аутентичность сообщения не обеспечивается.	
CryptDecryptMessage	Функция CryptDecryptMessage производит раскодирование и расшифрование сообщения. Проверка аутентичности сообщения не производится. Примечание: Не допускается автоматический анализ результата работы функции, направленный на проверку корректности сообщения.	
CryptGetMessageCertificates	Функция возвращает хранилище сертификатов и списки аннулированных сертификатов из сообщения	

CryptGetMessageSignerCount	Функция возвращает количество подписавших сообщение	
CryptHashMessage	Функция создает хэшированное сообщение	
CryptSignAndEncryptMessage	Функция создает подписанное и зашифрованное сообщение	
CryptSignMessageWithKey	Функция создает подписанное сообщение	
CryptVerifyDetachedMessageHash	Функция проверяет откреплённый хэш	
CryptVerifyMessageHash	Функция проверяет хэшированное сообщение	
CryptVerifyMessageSignatureWithKey	Функция проверяет подписанное сообщение	
CryptMsgCalculateEncodedLength	Функция CryptMsgCalculateEncodedLength вычисляет максимальное количество байтов, необходимое для закодированного криптографического сообщения, заданного типом сообщения, параметрами кодирования и общей длиной информации, которая должна быть закодирована.	
CryptMsgOpenToEncode	Функция CryptMsgOpenToEncode открывает криптографическое сообщение для закодирования и возвращает дескриптор открытого сообщения.	
CryptMsgOpenToDecode	Функция CryptMsgOpenToDecode открывает криптографическое сообщение для раскодирования и возвращает дескриптор открытого сообщения.	
CryptMsgUpdate	Функция CryptMsgUpdate пополняет текст криптографического сообщения.	
CryptMsgGetParam	Функция CryptMsgGetParam получает параметр сообщения после того, как криптографическое сообщение было раскодировано или закодировано.	
CryptMsgControl	Функция CryptMsgControl выполняет контрольное	

	действие.	
CryptMsgClose	Функция CryptMsgClose закрывает дескриптор криптографического сообщения.	
CryptMsgDuplicate	Функция CryptMsgDuplicate дублирует дескриптор криптографического сообщения путем увеличения счетчика ссылок	
Функции работы с алгоритмами хэширования		
CryptCreateHash	Функция CryptCreateHash инициализирует дескриптор нового объекта функции хэширования потока данных.	Разрешено использование только со следующими символьными аргументами: CALG_GR3411, CALG_GR3411_2012_256, CALG_GR3411_2012_512, CALG_GR3411_HMAC, CALG_GR3411_2012_256_HMAC, CALG_GR3411_2012_512_HMAC, CALG_PBKDF2_94_256, CALG_PBKDF2_2012_256, CALG_PBKDF2_2012_512, CALG_SHAREDKEY_HASH. Полученный дескриптор объекта хэширования должен в обязательном порядке быть удалён с помощью вызова функции CryptDestroyHash до вызова функции CryptReleaseContext для рабочего дескриптора криптопровайдера.
CryptDestroyHash	Функция CryptDestroyHash удаляет объект функции хэширования.	
CryptDuplicateHash	Функция CryptDuplicateHash создаёт точную копию объекта функции хэширования, включая все его переменные, определяющие внутреннее состояние объекта функции хэширования.	Полученный дескриптор объекта хэширования должен в обязательном порядке быть удалён с помощью вызова функции CryptDestroyHash до вызова функции CryptReleaseContext для рабочего дескриптора криптопровайдера.
CryptGetHashParam	Функция CryptGetHashParam возвращает параметры объекта функции хэширования и значение функции хэширования.	Запрещено использование с символьными аргументами HP_OPAQUEBLOB, HP_HASHSTATEBLOB.
CryptHashData	Функция CryptHashData передаёт данные указанному объекту	

	функции хэширования.	
CryptSetHashParam	Функция CryptSetHashParam устанавливает параметры объекта хэширования.	Разрешено использование только с символьными аргументами HP_HASHSTARTVECT (при условии, что передаваемый объект функции хэширования был создан функцией CryptCreateHash с символьным аргументом CALG_GR3411), HP_PBKDF2_SALT, HP_PBKDF2_PASSWORD, HP_PBKDF2_COUNT, HP_OID/KP_HASHOID, HP_OPEN.
CryptSignHash	Функция CryptSignHash возвращает значение электронной подписи от значения функции хэширования.	Разрешено использование только с ключевыми контейнерами, полученными ранее с помощью вызова CryptAcquireCertificatePrivateKey, либо с помощью вызова CertGetCertificateContextProperty из сертификата, проверенного с помощью функции CertVerifyCertificateChainPolicy
CryptVerifySignature	Функция CryptVerifySignature осуществляет проверку электронной подписи.	Разрешено использование только с дескрипторами ключей, полученными ранее с помощью вызова CryptImportPublicKeyInfo (CryptImportPublicKeyInfoEx) из сертификата, проверенного с помощью функции CertVerifyCertificateChainPolicy
Функции работы с сертификатами, списками отозванных (аннулированных) сертификатов (СОС), хранилищем сертификатов		
Списки аннулированных сертификатов		
CertAddCRLContextToStore	Функция CertAddCRLContextToStore добавляет контекст СОС в хранилище сертификатов.	
CertAddCRLLinkToStore	Функция создает ссылку на список аннулированных сертификатов в другом хранилище	
CertAddEncodedCRLToStore	Функция CertAddEncodedCRLToStore создает контекст СОС из закодированного СОС и добавляет его в хранилище сертификатов. Функция создает копию контекста СОС перед	

	добавлением его в хранилище.	
CertEnumCRLsInStore	Функция CertEnumCRLsInStore получает первый или следующий COC в хранилище. Эта функция используется в цикле для того, чтобы последовательно получить все COC в хранилище.	
CertFreeCRLContext	Функция CertFreeCRLContext освобождает контекст COC, уменьшая счетчик ссылок на единицу. Когда счетчик ссылок обнуляется, функция CertFreeCRLContext освобождает память, выделенную под контекст COC.	
CertCreateCRLContext	Функция CertCreateCRLContext создает контекст COC из закодированного COC. Созданный контекст не помещается в хранилище сертификатов. В созданном контексте функция размещает копию закодированного COC.	
CertDeleteCRLFromStore	Функция удаляет список аннулированных сертификатов из хранилища	
CertDuplicateCRLContext	Функция CertDuplicateCRLContext дублирует контекст COC, увеличивая счетчик ссылок на COC на единицу.	
CertFindCRLInStore	Функция CertFindCRLInStore находит первый или следующий контекст COC в хранилище сертификатов, который соответствует критерию поиска, определяемому параметром dwFindType и связанным с ним pvFindPara. Эта функция может быть использована в цикле для того, чтобы найти все COC в хранилище сертификатов, удовлетворяющие заданному критерию поиска.	
CertDeleteCertificateFromStore	Функция CertDeleteCertificateFromStore удаляет определенный контекст COC из хранилища сертификатов.	
CertFindCertificateInCRL	Функция осуществляет поиск	

	заданного сертификата в списке аннулированных сертификатов	
CertGetCRLFromStore	Функция CertGetCRLFromStore получает первый или следующий контекст COC для определенного издателя сертификата из хранилища сертификатов. Эта функция также осуществляет возможную проверку COC.	
CertSerializeCRLStoreElement	Функция сериализации списка аннулированных сертификатов со своими свойствами	
Расширенные свойства сертификата, списка отозванных (аннулированных) сертификатов (COC) и CTL		
CertGetCRLContextProperty	Функция CertGetCRLContextProperty получает расширенные свойства определенного контекста COC.	
CertSetCRLContextProperty	Функция CertSetCRLContextProperty устанавливает расширенные свойства определенного контекста COC.	
CertGetCertificateContextProperty	Функция CertGetCertificateContextProperty получает информацию, содержащуюся в расширенных свойствах контекста сертификата.	
CertEnumCertificateContextProperties	Функция CertEnumCertificateContextProperties позволяет перечислить информацию, содержащуюся в расширенных свойствах контекста сертификата.	
CertSetCertificateContextProperty	Функция CertSetCertificateContextProperty устанавливает расширенные свойства для определенного контекста сертификата.	
CertEnumCRLContextProperties	Перечисление расширенных свойств списка аннулированных сертификатов	
CertEnumCTLContextProperties	Перечисление расширенных свойств CTL	
CertGetCTLContextProperty	Получение расширенного свойства CTL	

CertSetCTLContextProperty	Установка расширенных свойств CTL	
CertAddCTLContextToStore	Функция CertAddCTLContextToStore добавляет контекст CTL в хранилище сертификатов	
CertCreateCTLContext	Функция CertCreateCTLContext создает контекст закодированного CTL. Созданный контекст не помещается в хранилище сертификатов. Функция делает копию CTL в созданном контексте.	
CertDuplicateCTLContext	Функция CertDuplicateCTLContext дублирует контекст CTL, увеличивая счетчик ссылок на CTL на единицу.	
CertFreeCTLContext	Функция CertFreeCTLContext освобождает контекст CTL, уменьшая счетчик ссылок на единицу. Когда счетчик ссылок обнуляется, функция CertFreeCTLContext освобождает память, выделенную под контекст CTL.	
Функции работы с сертификатами		
CertAddCertificateContextToStore	Функция CertAddCertificateContextToStore добавляет контекст сертификата в хранилище сертификатов.	
CertAddCertificateLinkToStore	Добавляет ссылку на сертификат в другом хранилище	
CertAddEncodedCertificateToStore	Функция CertAddEncodedCertificateToStore создает контекст сертификата из закодированного сертификата и добавляет его в хранилище сертификатов. Созданный контекст не содержит никаких расширенных свойств	
CertEnumCertificatesInStore	Функция CertEnumCertificatesInStore получает первый или следующий сертификат в хранилище сертификатов. Эта функция используется в цикле для того, чтобы последовательно получить все сертификаты в хранилище сертификатов	

CertFreeCertificateContext	Функция CertFreeCertificateContext освобождает контекст сертификата, уменьшая счетчик ссылок на единицу	
CertCreateCertificateContext	Функция CertCreateCertificateContext создает контекст сертификата из закодированного сертификата. Созданный контекст не помещается в хранилище сертификатов. В созданном контексте функция размещает копию закодированного сертификата	
CertDuplicateCertificateContext	Функция CertDuplicateCertificateContext дублирует контекст сертификата, увеличивая счетчик ссылок на единицу	
CertFindCertificateInStore	Функция CertFindCertificateInStore находит первый или следующий контекст сертификата в хранилище сертификатов, который соответствует критерию поиска, определяемому параметром dwFindType и связанным с ним pvFindPara	
CertDeleteCertificateFromStore	Функция CertDeleteCertificateFromStore удаляет определенный контекст сертификата из хранилища сертификатов	
CertGetSubjectCertificateFromStore	Функция CertGetSubjectCertificateFromStore получает контекст сертификата из хранилища сертификатов, однозначно определяемый его издателем и серийным номером	
CertGetIssuerCertificateFromStore	Поиск сертификатов издателей заданного сертификата	
CertGetSubjectCertificateFromStore	Поиск сертификата по серийному номеру и издателю	
CertGetValidUsages	Поиск пересечения KeyUsage для массива сертификатов	
CertSerializeCertificateStoreEleme ne	Сериализация элемента хранилища	

CertComparePublicKeyInfo	Функция CertComparePublicKeyInfo сравнивает два открытых ключа и определяет, являются ли они идентичными	
CertFindExtension	Функция CertFindExtension находит расширение в массиве и возвращает указатель на него.	
CertGetPublicKeyLength	Фукнция CertGetPublicKeyLength возвращает длину ключа в битах.	
CertGetIntendedKeyUsage	Функция CertGetIntendedKeyUsage получает назначение ключа из сертификата.	
CertCompareCertificateName	Функция CertCompareCertificateName сравнивает два сертификата и определяет, являются ли они идентичными.	
OCSP		
CertAddRefServerOcspResponse	Увеличение счетчика ссылок на OCSP ответ	
CertAddRefServerOcspResponseContext	Увеличение счетчика ссылок на контекст OCSP ответа	
CertCloseServerOcspResponse	Закрытие дескриптора OCSP ответа	
CertGetServerOcspResponseContext	Получение контекста OCSP ответа	
CertOpenServerOcspResponse	Открытие дескриптора OCSP ответа для заданной цепочки сертификатов	
Оконные функции		
CertSelectCertificate	Отображение диалога выбора сертификата по заданным критериям	
CryptUIDlgCertMgr	Отображение диалога управления сертификатами	
CryptUIDlgSelectCertificate	Отображение диалога выбора сертификата	
CryptUIDlgSelectCertificateFromStore	Отображение диалога выбора сертификата из хранилища	
CryptUIDlgViewCertificate	Отображение диалога со свойствами сертификата	

CryptUIDlgViewContext	Отображение сертификата, списка аннулированных сертификатов или CTL	
CryptUIDlgViewSignerInfo	Отображение диалога с информацией о подписавшем	
CertSelectionGetSerializedBlob	Сериализация сертификата из структуры, используемой для отображения	
GetFriendlyNameOfCert	Преобразование имени сертификата к «читаемому» виду	
Функции проверки цепочек		
CertVerifyCertificateChainPolicy	Функция CertVerifyCertificateChainPolicy проверяет цепочку сертификатов на достоверность, включая соответствие критерию истинности.	
CertGetCertificateChain	Функция CertGetCertificateChain строит цепочку сертификатов, начиная с последнего сертификата, в обратном направлении до доверенного корневого сертификата, если это возможно.	
CertFreeCertificateChain	Функция CertFreeCertificateChain освобождает цепочку сертификатов путем уменьшения счетчика ссылок. Если счетчик ссылок равен нулю, то память, выделенная под цепочку, освобождается.	
CertCreateCertificateChainEngine	Функция CertCreateCertificateChainEngine создает контекст HCERTCHAINENGINE, который позволяет изменять параметры механизма построения цепочки сертификатов. Позволяет ограничивать множество доверенных сертификатов.	
CertFreeCertificate-ChainEngine	Функция CertFreeCertificateChainEngine освобождает контекст HCERTCHAINENGINE.	
CertCreateCTLEntryFromCertificate ContextProperties	Создание CTL на основе свойств атрибутов контекста сертификата	

CertDuplicateCertificateChain	Дублирование контекста цепочки	
CertFindChainInStore	Функция построения цепочки по заданным критериям из хранилища	
CertFreeCertificateChainList	Функция освобождения массива цепочек	
CertIsValidCRLForCertificate	Функция проверки наличия сертификата в списке аннулированных сертификатов	
CertSetCertificateContextPropertiesFromCTLEntry	Установка свойств в контекст сертификата на основе CTL	
Расширенные свойства сертификата (EKU)		
CertGetEnhancedKeyUsage	Функция CertGetEnhancedKeyUsage получает информацию о расширенном использовании ключа из соответствующего расширения или из расширенных свойств сертификата. Расширенное использование ключа служит признаком правомерного использования сертификата.	
CryptAcquireCertificatePrivateKey	Функция CryptAcquireCertificatePrivateKey получает дескриптор HCRYPTPROV и параметр dwKeySpec для определенного контекста сертификата.	
Функции работы с идентификаторами		
CryptFindOIDInfo	Функция CryptFindOIDInfo получает первую предопределенную или зарегистрированную структуру CRYPT_OID_INFO, согласованную с определенным типом ключа и с ключом.	
CryptEnumOIDInfo	Перечисление зарегистрированных идентификаторов и получение информации для них	
Функции работы с хранилищем		
CertOpenStore	Функция CertOpenStore открывает хранилище сертификатов, используя заданный тип провайдера.	

CertDuplicateStore	Функция CertDuplicateStore дублирует дескриптор хранилища, увеличивая счетчик ссылок на хранилища на единицу.	
CertOpenSystemStore	Функция CertOpenSystemStore используется для открытия наиболее часто используемых хранилищ сертификатов.	
CertCloseStore	Функция CertCloseStore закрывает дескриптор хранилища сертификатов и уменьшает счетчик ссылок на хранилища на единицу.	
CertAddStoreToCollection	Добавление хранилища в коллекцию	
CertControlStore	Установка нотификации при различиях в закешированном хранилище и физическом хранилище	
Функции, используемые для работы с открытыми данными и объектами		
CryptImportPublicKeyInfoEx2	Функция CryptImportPublicKeyInfoEx2 импортирует информацию об открытом ключе в CNG и возвращает дескриптор открытого ключа.	
CryptImportPublicKeyInfoEx	Функция CryptImportPublicKeyInfoEx импортирует информацию об открытом ключе в CSP и возвращает дескриптор открытого ключа.	
CryptImportPublicKeyInfo	Функция CryptImportPublicKeyInfo преобразовывает и импортирует информацию об открытом ключе в провайдер и возвращает дескриптор открытого ключа.	
CryptExportPublicKeyInfoEx	Функция CryptExportPublicKeyInfoEx экспортирует информацию об открытом ключе, связанную с соответствующим закрытым ключом провайдера.	
CryptExportPublicKeyInfo	Функция CryptExportPublicKeyInfo экспортирует информацию об открытом ключе,	

	ассоциированную с соответствующим закрытым ключом провайдера.	
CertCompareCertificate	Функция CertCompareCertificate сравнивает два сертификата для того, чтобы определить, являются ли они идентичными.	
CertCompareIntegerBlob	Функция CertCompareIntegerBlob сравнивает два целочисленных блоба для определения того, представляют ли они собой два равных числа.	
CryptExportPublicKeyInfoFromBCryptKeyHandle	Экспортирует информацию об открытом ключе, ассоциированную с соответствующим закрытым ключом провайдера.	
Функции кодирования/декодирования		
CryptDecodeObject	Функция CryptDecodeObject используется для декодирования сертификатов, списков аннулированных сертификатов (COC) и запросов на сертификаты.	
CryptDecodeObjectEx	Функция CryptDecodeObjectEx используется для декодирования сертификатов, списков аннулированных сертификатов и запросов на сертификаты	
CryptEncodeObject	Функция CryptEncodeObject используется для кодирования сертификатов, списков аннулированных сертификатов и запросов на сертификаты.	
CryptEncodeObjectEx	Функция CryptEncodeObjectEx используется для кодирования сертификатов, списков аннулированных сертификатов и запросов на сертификаты.	
Получение объектов из удаленных источников		
CryptRetrieveObjectByUrlA	Функция CryptRetrieveObjectByUrlA получает объект инфраструктуры открытых ключей по заданному URL.	
CryptRetrieveObjectByUrlW	Функция CryptRetrieveObject-ByUrlW является unicode версией	

	функции CryptRetrieveObjectByUrlA.	
Дополнительные функции		
CryptBinaryToString	Функция переводит двоичную строку в строку Base64/HEX.	
CryptStringToBinary	Функция переводит строку Base64/HEX в двоичную строку.	
CertFindAttribute	Функция производит поиск атрибута сертификата по идентификатору.	
CertGetNameString	Функция получает имя владельца или издателя сертификата.	
CertNameToStr	Функция производит раскодирование имени из ASN структуры в DN (RFC1779).	
CertSaveStore	Функция производит запись хранилища сертификатов (включая списки отозванных и доверенных сертификатов) в виде структуры PKCS#7 или бинарного дампа в память или файл.	
CryptFindCertificateKeyProvInfo	Функция осуществляет поиск закрытого ключа, соответствующего открытому ключу сертификата.	
CryptHashPublicKeyInfo	Функция осуществляет ASN1 кодирование и хэширование структуры CERT_PUBLIC_KEY_INFO	
CryptMsgCountersign	Функция вырабатывает добавочную подпись.	
CryptMsgCountersignEncoded	Функция вырабатывает добавочную подпись (кодирует структуру SignerInfo, как определено в PKCS #7).	
CryptMsgVerifyCountersignatureEncoded	Функция проверяет добавочную подпись (декодирует структуру SignerInfo, как определено в PKCS #7).	
CryptMsgVerifyCountersignatureEncodedEx	Функция проверяет добавочную подпись (декодирует структуру SignerInfo, как определено в PKCS #7).	

Перечень вызовов, использование которых при разработке систем с использованием модуля «КриптоПро JavaCSP» и под управлением ОС Android с учетом п.1.5 Формуляра ЖТЯИ.000101-01 30 01 возможно без дополнительных тематических исследований:

Метод	Описание	Ограничения на использование метода
getInstance() класса KeyPairGenerator	Метод *getInstance()* создает объект генерации ключевой пары ЭП и VKO (генератор)	
getInstance() класса KeyFactory	Метод *getInstance()* создаёт объект, создающий объекты закрытых и открытых ключей ЭП и VKO на основе ключевых спецификаций, представляющих собой внутреннее представление ключевой информации	
getInstance() класса KeyStore	Метод создает объект, осуществляющий доступ к ключевому хранилищу.	
getInstance() класса SecureRandom	Метод создает объект класса генератора случайных чисел	
getInstance() класса Signature	Метод создает объект формирования ЭП в соответствии с указанным идентификатором алгоритма подписи.	
getInstance() класса MessageDigest	Метод создает объект функции хэширования в соответствии с указанным идентификатором алгоритма хэширования	
getInstance() класса KeyAgreement	Метод создает объект, вырабатывающий ключи согласования	
getInstance() класса Mac	Метод создает объект, осуществляющий имитозащиту данных по алгоритму, соответствующему указанному идентификатору.	Разрешен при указании параметра «HMAC_GOSTR3411», «HMAC_GOSTR3411_2012_256» или «HMAC_GOSTR3411_2012_512»
Initialize() класса KeyPairGenerator	Метод *initialize()* осуществляет операцию изменения набора параметров вырабатываемых ключевых пар ЭП или VKO (алгоритм ЭП/VKO, параметры используемой группы точек эллиптической кривой, алгоритм хэширования, узла замены для	

	алгоритма хэширования).	
generateKeyPair() класса KeyPairGenerator	Метод *generateKeyPair()* класса KeyPairGenerator осуществляет генерацию ключевой пары	
generatePublic() класса *KeyFactory*	Метод создает объект открытого ключа на основе спецификации открытого ключа (объекта класса PublicKeySpec), содержащей ключевой материал, или закодированного ключа (в виде объекта класса X509EncodedKeySpec).	
generatePrivate() класса *KeyFactory*	Метод создаёт объект закрытого ключа/ключа ЭП на основе спецификации закрытого ключа (объекта класса PrivateKeySpec), содержащей ключевой материал	
Load() класса KeyStore	Метод осуществляет загрузку содержимого стандартного хранилища JCA в память.	
getName() класса KeyStore	Метод возвращает имя хранилища JCA.	
getEntry() класса KeyStore	Метод возвращает дескриптор ключа для дальнейшей работы с ключом при указании верного пароля.	
setKeyEntry() класса KeyStore	Метод осуществляет запись закрытого ключа/ключа ЭП на носитель	
store() класса KeyStore	Метод осуществляет перезапись стандартного хранилища JCA	
getKey() класса KeyStore	Метод осуществляет чтение ключа ЭП с носителя	
setCertificateEntry() класса KeyStore	Метод осуществляет запись сертификата ключа проверки ЭП на носитель Запись сертификата в хранилище	
getCertificate() класса KeyStore	Метод осуществляет чтение сертификата ключа проверки ЭП с носителя Чтение сертификата из хранилища	
*deleteEntry()*класса KeyStore	Метод производит удаление ключевого контейнера с	

	носителя	
reset() класса MessageDigest	Метод переинициализирует объект хэширования для повторного использования после получения хэш-значения предыдущего сообщения. Метод может также изменять параметры хэширования (OID) (при использовании алгоритма хэширования ГОСТ Р 34.11-94)	
Clone() класса MessageDigest	Метод создает точную копию существующего объекта хэширования	
Update() класса MessageDigest	Метод осуществляет обработку хэшируемых данных	
read() класса DigestInputStream	Метод осуществляет обработку хэшируемых данных, получаемых из потока	
digest() класса MessageDigest	Метод завершает операцию хэширования и получает хэш-значение сообщения	
valid() класса PrivateKeyUsageExtension	Метод, проверяющий срок действия ключа электронной подписи	
initSign() класса Signature	Метод устанавливает ключ ЭП и параметры ЭП	
initVerify() класса Signature	Метод устанавливает ключ проверки электронной подписи и параметры ЭП	Метод разрешается использовать только для объектов открытых ключей, полученных с помощью механизмов PKIX.
setParameter() класса Signature	Метод устанавливает используемую хэш-функцию и её параметры в соответствии с переданным идентификатором.	
update() класса Signature	Метод осуществляет обработку переданных данных хэш-функцией.	
sign() класса Signature	Метод производит завершающее преобразование хэш-функции и производит подпись всего накопленного сообщения.	
verify() класса Signature	Метод производит завершающее преобразование хэш-функции и осуществляет проверку подписи	
generateCertificate() класса	Метод производит генерацию	

CertificateFactory	X.509 сертификатов	
getEncoded() класса Certificate, наследуется X509Certificate	Метод осуществляет кодирование существующего сертификата	
getPublicKey() класса Certificate, наследуется X509Certificate	Метод возвращает ключ проверки ЭП из сертификата	
checkValidity() класса X509Certificate	Метод осуществляет проверку срока действия сертификата X509	
getVersion() класса X509Certificate	Метод возвращает номер версии сертификата X509	
*getSerialNumber()*класса X509Certificate	Метод возвращает серийный номер сертификата X509	
getIssuerDN() класса X509Certificate	Метод возвращает DN эмитента сертификата	
getIssuerX500Principal() класса X509Certificate	Метод возвращает DN эмитента сертификата в формате X500	
getSubjectDN() класса X509Certificate	Метод возвращает DN владельца сертификата	
getSubjectX500Principal() класса X509Certificate	Метод возвращает DN владельца сертификата в формате X500	
*getNotBefore()*класса X509Certificate	Метод возвращает дату начала действия сертификата	
getNotAfter() класса X509Certificate	Метод возвращает дату окончания действия сертификата	
*getTBSCertificate()*класса X509Certificate	Метод осуществляет DER-кодирование сертификата	
getSignature() класса X509Certificate	Метод возвращает подпись под сертификатом	
getSigAlgName() класса X509Certificate	Метод возвращает название алгоритма подписи под сертификатом	
getSigAlgOID() класса X509Certificate	Метод возвращает OID алгоритма подписи под сертификатом	
getSigAlgParams() класса X509Certificate	Метод возвращает параметры алгоритма подписи под сертификатом в DER-кодировке	
getIssuerUniqueID() класса X509Certificate	Метод возвращает уникальный ID эмитента сертификата	
getSubjectUniqueID() класса X509Certificate	Метод возвращает уникальный ID владельца сертификата	

getKeyUsage() класса X509Certificate	Метод возвращает набор разрешенных областей использования ключа	
getExtendedKeyUsage() класса X509Certificate	Метод возвращает области расширенного использования ключа	
getBasicConstraints() класса X509Certificate	Метод возвращает длину расширения BasicConstraints	
getIssuerAlternativeNames() класса X509Certificate	Метод возвращает список альтернативных имён эмитента сертификата	
getSubjectAlternativeNames() класса X509Certificate	Метод возвращает список альтернативных имён владельца сертификата	
getOID() интерфейса ParamsInterface	Метод возвращает объектный идентификатор параметров алгоритма ЭП/VKO/хэширования/узлов замены шифрования.	
getDefault() интерфейса ParamsInterface	Метод возвращает значение объектного идентификатора, установленного в контрольной панели, параметров алгоритма ЭП/VKO/хэширования/узлов замены шифрования.	
setDefaultAvailable() интерфейса ParamsInterface	Метод осуществляет проверку наличия необходимых прав для установки новых параметров по умолчанию в контрольную панель.	
setDefault(OID def) интерфейса ParamsInterface	Метод устанавливает объектный идентификатор по умолчанию для параметров алгоритма ЭП/VKO/хэширования/узлов замены шифрования.	
getOIDs() интерфейса ParamsInterface	Метод получает список допустимых объектных идентификаторов параметров для алгоритма ЭП/VKO/хэширования/узлов замены шифрования.	
getDefaultSignParams() класса AlgIdSpec	Метод возвращает параметры алгоритма подписи по умолчанию	
getDefaultDigestParams() класса AlgIdSpec	Метод возвращает параметры алгоритма хэширования по	

	умолчанию	
getDefaultCryptParams() класса AlgIdSpec	Метод возвращает параметры алгоритма шифрования по умолчанию	
setKeyUsage() класса GostCertificateRequest	Метод устанавливает значение поля keyUsage в запросе на сертификат, описывающее разрешенные области использования ключа.	
addExtKeyUsage() класса GostCertificateRequest	Метод устанавливает значение поля extKeyUsage в запросе на сертификат, описывающее дополнительные области использования ключа.	
addExtension() класса *GostCertificateRequest*	Метод устанавливает дополнительное расширение в список расширений.	
setPublicKeyInfo() класса GostCertificateRequest	Метод осуществляет кодирование и запись в структуру запроса параметров и значения передаваемого открытого ключа субъекта	
setSubjectInfo() класса GostCertificateRequest	Метод определяет имя субъекта в формате X.500, устанавливает новое имя субъекта.	
encodeAndSign() класса GostCertificateRequest	Метод выполняет кодирование запроса в DER-кодировку и подпись сертификата.	
getEncodedCert() класса GostCertificateRequest	Метод отправляет запрос центру сертификации и получает соответствующий запросу сертификат	
getEncodedCertFromDER() класса GostCertificateRequest	Метод отправляет запрос в DER-кодировке центру сертификации и получает соответствующий запросу сертификата	
printToDER() класса GostCertificateRequest	Метод осуществляет печать подписанного запроса в DER-кодировке в передаваемый PrintStream	
getEncodedCertFromBASE64() класса GostCertificateRequest	Метод осуществляет отправку запроса в кодировке BASE64 центру сертификации и получение соответствующего запросу сертификата	
printToBASE64() класса	Метод осуществляет печать	

GostCertificateRequest	подписанного запроса в BASE64-кодировке в передаваемый PrintStream	
getEncodedRootCert() класса GostCertificateRequest	Метод запрашивает и получает корневой сертификат центра сертификации	
getRootCertList() класса CA15GostCertificateRequest	Метод получает список корневых сертификатов УЦ (CA15)	
sendCertificateRequest () класса CA15GostCertificateRequest	Метод отправляет в УЦ запрос на сертификат.	
getCertificateRequestList () класса CA15GostCertificateRequest	Метод получает список запросов на сертификат и статус их обработки.	
checkCertificateStatus () класса CA15GostCertificateRequest	Метод получает статус обработки УЦ ранее отправленного запроса на сертификат.	
getCertificateRequestId() класса CA15GostCertificateRequest	Метод получает строки-идентификаторы запроса на сертификат.	
getCertificateByRequestId() класса CA15GostCertificateRequest	Метод получает выпущенный сертификат по идентификатору запроса.	
init() класса Mac	Метод, устанавливающий алгоритм шифрования	
clone() класса Mac	Метод, производящий копирование объекта имитозащиты	
update() класса Mac	Метод, осуществляющий обработку данных в процессе вычисления имитовставки	
doFinal() класса Mac	Метод, завершающий вычисление значения имитовставки	
reset() класса Mac	Метод, осуществляющий переинициализацию объекта класса Mac после окончания вычисления значения имитовставки предыдущего сообщения	
verify() класса CAdESSignature	Метод осуществляет проверку всех подписей CAdES	

getCAdESSignerInfos() класса CAdESSignature	Метод получает список всех подписантов	
setCertificateStore() класса CAdESSignature	Метод устанавливает набор сертификатов, которые будут упакованы в подписанное сообщение	
setCRLStore() класса CAdESSignature	Метод устанавливает набор списков отозванных сертификатов, которые будут упакованы в подписанное сообщение	
*addSigner()*класса CAdESSignature	Метод осуществляет добавление подписи в формируемое подписанное сообщение.	
open() класса CAdESSignature	Метод осуществляет открытие потока подписываемых данных	
update() класса CAdESSignature	Метод осуществляет обработку порции данных	
close() класса CAdESSignature	Метод осуществляет завершение обработки подписываемых данных.	
replaceSigners() класса CAdESSignature	Метод, осуществляющий изменения данных подписывающих в упакованном подписанном сообщении.	
addRecipient() класса EnvelopedSignature	Метод, добавляющий информацию о получателе CMS сообщения, и подготавливающий сообщение к зашифрованию.	
addKeyTransRecipient() класса EnvelopedSignature	Метод добавляет переданный сертификат в список информации о получателе CMS сообщения в виде структуры key_transport.	Метод разрешается использовать только при осуществлении проверки цепочки сертификата с помощью механизмов PKIX и проверки области действия сертификата.
addKeyAgreeRecipient() класса EnvelopedSignature	Метод добавляет переданный сертификат в список информации о получателе CMS сообщения в виде структуры key_agreement.	Метод разрешается использовать только при осуществлении проверки цепочки сертификата с помощью механизмов PKIX и проверки области действия сертификата.
open() класса EnvelopedSignature	Метод, осуществляющий открытие потока зашифрования CMS сообщения.	
update() класса EnvelopedSignature	Метод, осуществляющий шифрование порции данных CMS	

	сообщения.	
close() класса EnvelopedSignature	Метод, осуществляющий шифрование финальной порции данных и закрывающий поток зашифрования CMS сообщения.	
getRecipients() класса EnvelopedSignature	Метод, возвращающий список получателей CMS сообщения	
decrypt() класса EnvelopedSignature	Метод, позволяющий получателю расшифровать CMS сообщение	
getCAdESSignatureType() класса CAdESType	Метод возвращает тип CAdES сообщения	
getSignerInfo() класса CAdESSigner	Метод возвращает информацию о подписанте	
getSignerSignedAttributes() класса CAdESSigner	Метод возвращает список подписываемых атрибутов сообщения	
getSignerUnsignedAttributes() класса CAdESSigner	Метод возвращает список неподписываемых атрибутов сообщения	
getSignatureTimestampToken() класса CAdESSigner	Метод возвращает внутренний штамп (signature-timestamp) времени сообщения	
getCAdESCTimestampToken() класса CAdESSigner	Метод возвращает внешний штамп (CAdES-C-timestamp) времени сообщения	
getCAdESCertificates() класса CAdESSigner	Метод возвращает список сертификатов (certificate-values)	
getSignerCertificate() класса CAdESSigner	Метод возвращает сертификат подписавшего сообщение	
getSignatureType() класса CAdESSigner	Метод возвращает тип подписи	
getCAdESCountersignerInfos() класса CAdESSigner	Метод возвращает список заверителей сообщения	
setCertificateStore() класса CAdESSigner	Метод устанавливает хранилище сертификатов, из которого позже может быть получен сертификат подписи	
enhance() класса CAdESSigner	Метод осуществляет «усовершенствование» подписи CAdES-BES до CAdES-X Long Type 1	
addCountersigner() класса CAdESSigner	Метод, осуществляющий добавление заверяющей подписи	

	к отдельному подписанту.	
verify() класса CAdESSigner	Метод, осуществляющий проверку одной отдельной подписи CAdES	
build() класса CertPathBuilder	Метод, осуществляющий построение цепочки сертификатов	
getAlgorithm() класса CertPathValidator	Метод, возвращающий имя алгоритма, осуществляющего проверку цепочек сертификатов.	
getDefaultType() класса CertPathValidator	Метод, возвращающий имя алгоритма, осуществляющего проверку цепочек сертификатов, принятого по умолчанию.	
getInstance() класса CertPathValidator	Метод, возвращающий объект класса CertPathValidator.	
getProvider() класса CertPathValidator	Метод, осуществляющий получение объекта провайдера алгоритма, осуществляющего проверку цепочек сертификатов, принятого по умолчанию.	
validate() класса CertPathValidator	Метод, осуществляющий проверку цепочек сертификатов.	
addCertPathChecker() класса CertPathParameters	Метод, устанавливающий дополнительные правила проверки сертификатов.	
addCertStore() класса CertPathParameters	Метод, устанавливающий дополнительные хранилища сертификатов и CRL.	
getCertPathCheckers() класса CertPathParameters	Метод, возвращающий список установленных дополнительных правил проверки сертификатов.	
getCertStores() класса CertPathParameters	Метод, возвращающий список хранилищ сертификатов и CRL.	
getDate() класса CertPathParameters	Метод, возвращающий дату, на которую проверяется верность цепочки сертификатов.	
getInitialPolicies() класса CertPathParameters	Метод, возвращающий список OID’ов политик, которые должны быть указаны в сертификатах, из которых строится цепочка.	
getPolicyQualifiersRejected() класса CertPathParameters	Метод, возвращающий флаг PolicyQualifiersRejected.	
getSigProvider() класса	Метод, получающий имя	

CertPathParameters	провайдера ЭП.	
getTargetCertConstraints() класса CertPathParameters	Метод, возвращающий ограничения на целевой сертификат.	
getTrustAnchors() класса CertPathParameters	Метод, получающий список доверенных сертификатов.	
isAnyPolicyInhibited() класса CertPathParameters	Метод, возвращающий признак обработки всех политик, указанных в сертификате.	
isExplicitPolicyRequired() класса CertPathParameters	Метод, возвращающий признак наличия явных описаний политик.	
isPolicyMappingInhibited() класса CertPathParameters	Метод, возвращающий признак подавления PolicyMapping.	
isRevocationEnabled() класса CertPathParameters	Метод, устанавливающий признак необходимости проверки сертификатов с помощью OCSP/CRL.	
setAnyPolicyInhibited() класса CertPathParameters	Метод, устанавливающий признак обработки всех политик, указанных в сертификате.	
setCertPathCheckers() класса CertPathParameters	Метод, устанавливающий набор дополнительных правил проверки сертификатов.	
setCertStores() класса CertPathParameters	Метод, устанавливающий набор дополнительных хранилищ сертификатов и CRL.	
setDate() класса CertPathParameters	Метод, устанавливающий дату, на которую должна производиться проверка цепочки сертификатов.	
setExplicitPolicyRequired() класса CertPathParameters	Метод, устанавливающий признак наличия явных описаний политик.	
setInitialPolicies() класса CertPathParameters	Метод, устанавливающий список OID’ов политик, которые должны быть указаны в сертификатах, из которых строится цепочка.	
setPolicyMappingInhibited() класса CertPathParameters	Метод, устанавливающий признак подавления PolicyMapping.	
setPolicyQualifiersRejected() класса CertPathParameters	Устанавливает флаг PolicyQualifiersRejected.	
setRevocationEnabled() класса	Метод, устанавливающий	

CertPathParameters	признак необходимости проверки сертификатов с помощью OCSP/CRL.	
setTargetCertConstraints() класса CertPathParameters	Метод, устанавливающий ограничения на целевой сертификат.	
setTrustAnchors() класса CertPathParameters	Метод, устанавливающий список доверенных сертификатов.	
getMaxPathLength() класса PKIXBuilderParameters	Метод, получающий максимальную возможную длину строимой цепочки.	
setMaxPathLength() класса PKIXBuilderParameters	Метод, устанавливающий максимальную возможную длину строимой цепочки.	
getXAdESSignerInfos() класса XAdESSIgnature	Метод получает список всех подписантов	
addSigner() класса XAdESSIgnature	Метод осуществляет добавление подписи в формируемое подписанное XML сообщение.	
open() класса XAdESSIgnature	Метод осуществляет открытие потока подписываемых данных	
update() класса XAdESSIgnature	Метод осуществляет обработку порции данных	
close() класса XAdESSIgnature	Метод осуществляет завершение обработки подписываемых данных.	
verify() класса XAdESSIgnature	Метод осуществляет проверку всех подписей XAdES	
getSignatureType() класса XAdESSigner	Метод, возвращающий тип XAdES-подписи.	
getEarliestValidSignatureTimeStampToken() класса XAdESSignerT	Метод, возвращающий самый ранний валидный внутренний штамп времени.	
verify() класса XAdESSigner	Метод, осуществляющий проверку одной отдельной подписи XAdES	
getSignerInfo() класса XAdESSigner	Метод, возвращающий узел подписи в документе.	
getSignerCertificate() класса XAdESSigner	Метод, возвращающий сертификат ключа проверки ЭП данного подписанта.	
getElement() класса XAdESSigner	Метод, возвращающий подписываемый узел.	

getDocument() класса XAdESSigner	Метод, возвращающий подписываемый документ.	
getPrivateKey() класса JCPPrivateKeyEntry	Метод, возвращающий дескриптор ключа privKey.	
isExportable() класса JCPPrivateKeyEntry	Метод, возвращающий значение флага экспортируемости закрытого ключа/ключа ЭП.	
getCertificateChain() класса JCPPrivateKeyEntry	Метод, возвращающий цепочку сертификатов.	
getCertificate() класса JCPPrivateKeyEntry	Метод, возвращающий клиентский сертификат (первый в цепочке сертификатов).	
toString() класса JCPPrivateKeyEntry	Метод, возвращающий строковое представление цепочки сертификатов.	
isSilentMode() класса JCPProtectionParameter	Метод, возвращающий режим открытия ключевого контейнера.	
isAllowEmptyChain() класса JCPProtectionParameter	Метод, возвращающий разрешение использовать null вместо цепочки сертификатов.	
getKeyType() класса JCPProtectionParameter	Метод, возвращающий тип ключа.	
sign() класса XMLSignature	Метод создает ЭП сообщения.	Разрешено использование только со следующими URI/URN: • http://www.w3.org/2001/04/xmldsig-more#gostr34102001-gostr3411 • urn:ietf:params:xml:ns:cpxmlsec:algorithms:gostr34102001-gostr3411 • urn:ietf:params:xml:ns:cpxmlsec:algorithms:gostr34102012-gostr34112012-256 • urn:ietf:params:xml:ns:cpxmlsec:algorithms:gostr34102012-gostr34112012-512
checkSignatureValue() класса XMLSignature	Метод осуществляет проверку ЭП сообщения.	Метод разрешается использовать только при осуществлении проверки сертификата открытого ключа с помощью механизмов PKIX.

Литература

1. ГОСТ 28147-89. Государственный стандарт Российской Федерации. Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования.

2. ГОСТ Р 34.10-2001. Государственный стандарт Российской Федерации. Информационная технология. Криптографическая защита информации. Процессы формирования и проверки электронной цифровой подписи.

3. ГОСТ Р 34.11-94. Государственный стандарт Российской Федерации. Информационная технология. Криптографическая защита информации. Функция хэширования.

4. ГОСТ Р 34.10-2012. Информационная технология. Криптографическая защита информации. Процессы формирования и проверки электронной цифровой подписи.

5. ГОСТ Р 34.11-2012. Информационная технология. Криптографическая защита информации. Функция хэширования.

6. ГОСТ 34.10-2018. Информационная технология. Криптографическая защита информации. Процессы формирования и проверки электронной цифровой подписи.

7. ГОСТ 34.11-2018. Информационная технология. Криптографическая защита информации. Функция хэширования.

8. ГОСТ 34.12-2018. Информационная технология. Криптографическая защита информации. Блочные шифры.

9. ГОСТ 34.13-2018. Информационная технология. Криптографическая защита информации. Режимы работы блочных шифров.

10. Р 50.1.115–2016. Информационная технология. Криптографическая защита информации. Протокол выработки общего ключа с аутентификацией на основе пароля.

11. Р 1323565.1.023-2018. Информационная технология. Криптографическая защита информации. Использование алгоритмов ГОСТ Р 34.10-2012, ГОСТ Р 34.11-2012 в сертификате, списке аннулированных сертификатов (CRL) и запросе на сертификат PKCS #10 инфраструктуры открытых ключей X.509.

12. [X.680-X.699]. OSI NETWORKING AND SYSTEM ASPECTS. Abstract Syntax Notation One (ASN.1)

13. [X.509]. ITU-T Recommendation X.509 (1997 E): Information Technology - Open Systems Interconnection – The Directory: Authentication Framework, June 1997.

14. [PKIX]. RFC 2459. Housley, W. Ford, W. Polk, D. Solo, «Internet X.509 Public Key Infrastructure Certificate and CRL Profile», January 1999.

15. Положение о разработке, производстве, реализации и эксплуатации шифровальных (криптографических) средств защиты информации (Положение ПКЗ-2005).

16. Приказ ФАПСИ от 13.06.2001г. №152 «Об инструкции об организации и обеспечении безопасности хранения, обработки и передачи по каналам связи с использованием средств криптографической защиты информации с ограниченным доступом, не содержащей сведений, составляющих государственную тайну».

Лист регистрации изменений									
	Номера листов (страниц)				Всего листов (страниц) в докум.	№ документа	Входящий № сопроводи-тельного докум. и дата	Подп.	Дата
изм.	измененных	замененных	новых	аннулиро-ванных					