127018, Москва, Сущёвский Вал, 18
Телефон: (495) 995 4820
Факс: (495) 995 4820
https://CryptoPro.ru
E-mail: info@CryptoPro.ru

УТВЕРЖДЕНЫ
ЖТЯИ.00101-02 95 01-ЛУ

Средство Криптографической Защиты Информации

КриптоПро CSP

Версия 5.0 R2 KC1

Исполнение 1-Base

Правила пользования

ЖТЯИ.00101-02 95 01
Листов 70

2021 г

Содержание

Аннотация

Данный документ содержит правила использования средства криптографической защиты информации (СКЗИ) «КриптоПро CSP» версия 5.0 R2 KC1 исполнение 1-Base, его состав, назначение, ключевую систему, требования и условия эксплуатации.

Документ предназначен для администраторов информационной безопасности, осуществляющих установку, обслуживание и контроль за соблюдением требований к эксплуатации средств СКЗИ, для администраторов серверов, сетевых ресурсов предприятия и других 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 R2 KC1 исполнение 1-Base, должны разрабатываться с учетом требований настоящих Правил пользования.

1 Назначение СКЗИ. Условия эксплуатации

СКЗИ «КриптоПро CSP» версия 5.0 R2 KC1 исполнение 1-Base 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» версия 5.0 R2 KC1 исполнение 1-Base 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» версия 5.0 R2 KC1 исполнение 1-Base должны выполняться следующие требования:

- 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.
- При создании защищенных с использованием шифровальных (криптографических) средств информационных систем необходимо на основании модели угроз и нарушителя на эту систему определить необходимость применения антивирусных средств (АВС). Если такая необходимость определена, должны применяться АВС, сертифицированные органом, ответственным за обеспечение информационной безопасности в создаваемой информационной системе.
- 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 — СОС (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 [1, 2] и другими руководящими документами по обеспечению безопасности информации.

СКЗИ «КриптоПро CSP» версия 5.0 R2 KC1 исполнение 1-Base 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;
- Microsoft Outlook (из состава Microsoft Office 2007, Office 2010, Office 2013, Office 2016, Office 2019);
- сервер IIS (защита TLS соединений);
- сервер nginx версии 1.18.0 (защита TLS соединений);

- сервер Apache версии 2.4.41 (защита TLS соединений);
- сервер Apache версии 2.4.25, используемый в ОС Astra Linux (защита TLS соединений);
- веб-браузер Chromium ГОСТ для ОС Astra Linux (защита TLS соединений).

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.00101-02 30 01.

Следующие приложения (модули) из состава СКЗИ «КриптоПро CSP» версия 5.0 R2 KC1 исполнение 1-Base:

- приложение (модуль) командной строки для подписи и шифрования файлов cryptcp,
- приложение (модуль) командной строки для работы с сертификатами certmgr,
- приложения (модули) для создания TLS-туннеля stunnel и stunnel_msspi

можно использовать без проведения оценки влияния указанных приложений (модулей) на программные интерфейсы СКЗИ «КриптоПро CSP» версия 5.0 R2 KC1 исполнение 1-Base. При этом необходимость проведения оценки влияния компонентов среды функционирования (прикладных систем, программных продуктов и т.п.), вызывающих указанные приложения (модули), на их штатное функционирование определяется с учетом положений разд. 4 настоящих Правил пользования.

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

Требования к встраиванию СКЗИ в прикладные системы и к проведению исследований СФ СКЗИ описаны в разд. 4.

Использование СКЗИ «КриптоПро CSP» версия 5.0 R2 KC1 исполнение 1-Base с выключенным режимом усиленного контроля использования ключей **не допускается**. Включение данного режима описано в Руководствах администратора безопасности (для ОС Windows, Linux, FreeBSD, Solaris, AIX, Mac OS, iOS, Sailfish), входящих в состав эксплуатационной документации на СКЗИ.

2 Порядок распространения СКЗИ

Установочные модули СКЗИ «КриптоПро CSP» версия 5.0 R2 KC1 исполнение 1-Base 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 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 и комплекта эксплуатационной документации. При загрузке пользователем установочных модулей СКЗИ и комплекта эксплуатационной документации Уполномоченной организацией присваивается учетный номер, идентифицирующий экземпляр СКЗИ, 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 на рабочее место пользователя может быть осуществлена только в случае подтверждения целостности полученных установочных модулей СКЗИ и эксплуатационной документации.

Примечание.
1) Средство контроля целостности (`cpverify`) 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 https://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 среды функционирования (СФ) СКЗИ обеспечивается при помощи утилиты `cpverify` в соответствии с Приложением 1.

3 Ключевая система и ключевые носители

СКЗИ «КриптоПро CSP» версия 5.0 R2 КС1 исполнение 1-Base 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».

Примечание. Виды поддерживаемых ключевых носителей и их использование в зависимости от программно-аппаратной платформы отражены в ЖТЯИ.00101-02 30 01. КриптоПро CSP. Формуляр, п. 3.10.

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. Таким образом обеспечивается конфиденциальность и неизменность передаваемых между СКЗИ и носителем данных. Для некоторых носителей такая поддержка отсутствует (см. соответствующие примечания к таблице 3.1 документа ЖТЯИ.00101-02 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.

• **КриптоПро Cloud CSP** (облачный провайдер, модуль взаимодействия с 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 КриптоПро Cloud CSP см. п. 2.4 документа ЖТЯИ.00101-02 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.

Примечание. При удаленном подключении ключевых носителей необходимо использовать ФКН с поддержкой SESPAKE.

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 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.

3.6 Размеры и сроки действия ключей

Размеры ключей электронной подписи:

ключ электронной подписи	256 бит или 512 бит;
ключ проверки электронной подписи	512 бит или 1024 бита.

Размеры 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 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 КриптоПро Cloud CSP — 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 никаких 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 определяется следующим образом:

1) Определяется значение срока действия из расширения контейнера. Если в расширении указан срок окончания действия ключа (*NotAfter, Действителен по*), то выбирается это значение. Иначе срок определяется от начала действия ключа (*NotBefore, Действителен с*) + 1 год 3 месяца для пассивных ключевых носителей или + 3 года для неэкспортируемых ключей при их использовании в ФКН и КриптоПро Cloud CSP.

2) Аналогичным образом определяется значение из расширения сертификата. Если расширение в сертификате отсутствует, то срок считается по дате выпуска сертификата.

3) В качестве значения срока действия ключа выбирается наименьшее из полученных значений.

Изменение параметра ControlKeyTimeValidity

Для операционных систем Windows необходимо изменить значение ключа реестра `HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Crypto Pro\Cryptography\CurrentVersion\Parameters\ControlKeyTimeValidity` (для 64-битных ОС), `HKEY_LOCAL_MACHINE\SOFTWARE\Crypto Pro\Cryptography\CurrentVersion\Parameters\ControlKeyTimeValidity` (для 32-битных ОС).

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 ключей и ключевых носителей

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 в «Службе хранения ключей» обеспечивает дополнительную защиту ключевой информации от других приложений, выполняющихся на ПЭВМ, но может незначительно замедлить производительность системы.

При хранении ключей и ключевых носителей должны выполняться следующие требования:

• 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 устройствах Apple iOS/Android/Sailfish 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.

• Необходимо использовать парольную защиту, если не оговорено иное.

• В случае необходимости проведения ремонтных и регламентных работ аппаратной части среды функционирования (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** значение «1» (см. ниже), force_silent равный «0» 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 семейства Windows необходимо изменить значение ключа реестра `HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Crypto Pro\Cryptography\CurrentVersion\Parameters\force_silent` (для 64-битных ОС), `HKEY_LOCAL_MACHINE\SOFTWARE\Crypto Pro\Cryptography\CurrentVersion\Parameters\force_silent` (для 32-битных ОС).

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

```
./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 носителя при дальнейших неуспешных попытках аутентификации и предлагает заранее сменить пароль.

3.8.2 Требование пароля контейнера/носителя для каждой операции

Каждый ключевой контейнер можно создать в режиме повышенной защиты. Созданный таким образом контейнер вне зависимости от ключевого носителя будет требовать предъявление пароля на каждую операцию. Изменить это свойство нельзя.

Для этого при формировании ключа в функцию CryptGenKey необходимо передать флаги (CRYPT_FORCE_KEY_PROTECTION_HIGH | CRYPT_USER_PROTECTED).

В случае импорта ключа из файла pfx с использованием утилиты `certmgr` для требования пароля на каждую операцию необходимо указать флаг `-protected high` (подробнее см. ЖТЯИ.00101-02 93 02. Приложение командной строки для работы с сертификатами).

В ОС семейства Windows настройка требования пароля на каждую операцию может быть изменена с помощью групповой политики «Требовать пароль контейнера/носителя для каждой операции» или с помощью ключа реестра `HKEY_LOCAL_MACHINE\SOFTWARE\[WOW6432Node]\Crypto Pro\Cryptography\CurrentVersion\Parameters\DisableCachePasswords` (см. раздел 4.4 ЖТЯИ.00101-02 91 02. Руководство администратора безопасности. Windows).

Настройка для остальных ОС осуществляется с использованием утилиты `cpconfig` с помощью команды:

```
./cpconfig -ini '\config\parameters' -add long DisableCachePasswords <значение>
```

Значения параметра DisableCachePasswords:

- 0 — не задано;
- 1 — для открываемых ключей: при открытии для всех контейнеров будут требоваться пароли на каждую операцию;
- 2 — для создаваемых ключей: для всех генерируемых на компьютере ключей будет требоваться пароль на каждую операцию;
- 3 — для всех: объединяет 2 предыдущих значения.

3.8.3 Настройка политики паролей контейнеров

Политика паролей указывается с помощью регулярного выражения.

В ОС Windows настройка политики осуществляется с помощью ключа реестра `HKEY_LOCAL_MACHINE\SOFTWARE\[WOW6432Node]\Crypto Pro\Cryptography\CurrentVersion\Parameters\PasswordPolicies\AsciiPasswordPolicy`. Значение по умолчанию — `[ -~]*` (все отображаемые ASCII-символы).

Настройка для остальных ОС осуществляется с использованием утилиты `cpconfig` с помощью команды (указано значение по умолчанию, все отображаемые ASCII-символы):

```
./cpconfig -ini '\config\parameters\PasswordPolicies' -add string "AsciiPasswordPolicy"
      "[ [:punct:][:digit:]ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz]*"
```

3.9 Управление ключами

Ключевая система СКЗИ КриптоПро CSP 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 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.00101-02 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 [3], при попытке генерации ключа алгоритма ГОСТ Р 34.10-2001 будет выдано следующее предупреждение:

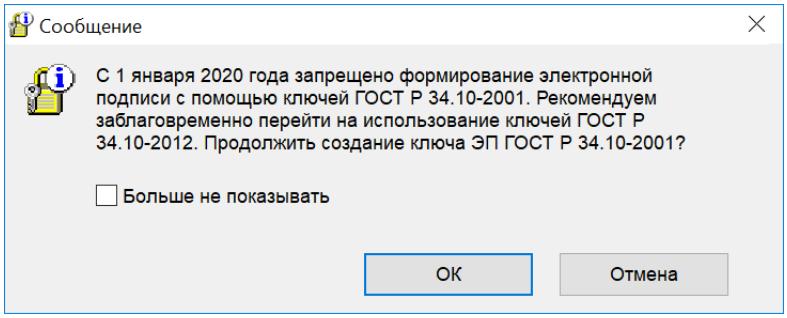

Рисунок 1. Предупреждение о генерации ключа алгоритма ГОСТ Р 34.10-2001 для ОС семейства Windows

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

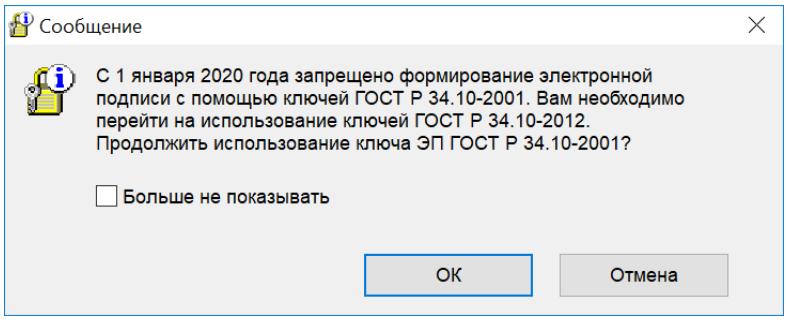

Рисунок 2. Предупреждение о создании подписи с использованием ключа алгоритма ГОСТ Р 34.10-2001 для ОС семейства Windows

Примечание. При использовании операционных систем, отличных от ОС семейства Windows, окно предупреждения может выглядеть иначе, но текстовая составляющая аналогична приведенной на рис. 1 и рис. 2.

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:

1) потеря ключевых носителей;

2) потеря ключевых носителей с их последующим обнаружением;

3) увольнение сотрудников, имевших доступ к ключевой информации;

4) нарушение правил хранения и уничтожения (после окончания срока действия) закрытого ключа;

5) возникновение подозрений на утечку информации или ее искажение в системе конфиденциальной связи;

6) нарушение печати на сейфе с ключевыми носителями;

7) 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. События 1–4 трактуются как явная компрометация ключей. События 5–7 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.

По факту компрометации должно быть проведено служебное расследование. Скомпрометированные ключи выводятся из действия. Выведенные из действия скомпрометированные ключевые носители после проведения служебного расследования уничтожаются.

Скомпрометированные ключи подлежат замене. 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 к встраиванию СКЗИ в прикладные системы и к проведению исследований СФ СКЗИ

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.

Для обеспечения информационной безопасности рабочих мест и информационных систем, использующих СКЗИ, должны быть определены модель возможных угроз и нарушителя и выработана политика безопасности. В зависимости от модели возможных угроз и нарушителя определяется необходимый уровень защиты и, соответственно, необходимый класс СКЗИ.

Возможны следующие варианты применения СКЗИ КриптоПро CSP:

1) применение СКЗИ КриптоПро CSP в составе стандартного программного обеспечения Microsoft, ООО «КРИПТО-ПРО» и других компаний, использующих криптографический интерфейс в соответствии с архитектурой Microsoft (подробнее см. разд. 4.1);

2) встраивание СКЗИ КриптоПро CSP во вновь разрабатываемое или существующее прикладное программное обеспечение, программно-аппаратные и аппаратные решения (подробнее см. ЖТЯИ.00101-02 90 01. КриптоПро CSP. Описание реализации и ЖТЯИ.00101-02 90 01. КриптоПро CSP. Руководство программиста).

При этом указанное программное обеспечение, программно-аппаратные и аппаратные решения по отношению к СКЗИ рассматриваются в качестве среды функционирования (СФ).

Встраивание СКЗИ в СФ проводится в соответствии с порядком, определенным Положением ПКЗ-2005 [2], организациями, имеющими соответствующие лицензии [4] на указанные виды деятельности.

Для указанных вариантов применения СКЗИ в зависимости от условий эксплуатации и от конкретной СФ, в составе которой применяется СКЗИ, могут существовать следующие варианты наличия или отсутствия необходимости проведения исследований СФ:

1) исследования СФ не требуются;

2) требуются исследования по оценке влияния СФ на СКЗИ;

3) требуется проведение тематических исследований СФ со встроенным СКЗИ как самостоятельного шифровального (криптографического) средства[1].

Исследования по оценке влияния СФ на СКЗИ, а также тематические исследования СФ со встроенным СКЗИ как самостоятельного шифровального (криптографического) средства проводятся в соответствии с порядком, определенным Положением ПКЗ-2005 [2], организациями, имеющими соответствующие лицензии [4] на указанные виды деятельности и необходимую область аккредитации [5].

Перечень компонентов СФ, для которых необходимо проводить исследования, определяется исходя из архитектуры информационной системы и выполняемых функций.

Исследования СФ не требуются при выполнении одного из следующих условий:

1) применение СКЗИ и СФ не подпадает под случаи, приведенные в п. 3 Положения ПКЗ-2005, и иными нормативными документами не определены дополнительные условия применения шифровальных (криптографических) средств;

2) в качестве СФ используются следующие программные компоненты:

- приложения, входящие в состав ОС, под управлением которых может функционировать СКЗИ (см. п. 3.2 Формуляра ЖТЯИ.00101-02 30 01);
- приложение командной строки ОС семейства Windows для формирования запроса на сертификат – certreq;
- Microsoft Outlook (из состава Microsoft Office 2007, Office 2010, Office 2013, Office 2016, Office 2019);
- сервер IIS (защита TLS соединений);

[1] В соответствии с [4] термины «шифровальное (криптографическое) средство» и «средство криптографической защиты информации» (СКЗИ) являются эквивалентными и включают в себя, в том числе, термин «Средство электронной подписи».

- сервер nginx версии 1.18.0 (защита TLS соединений);
- сервер Apache версии 2.4.41 (защита TLS соединений);
- сервер Apache версии 2.4.25, используемый в ОС Astra Linux (защита TLS соединений);
- веб-браузер Chromium ГОСТ для ОС Astra Linux (защита TLS соединений);
- приложения (модули) из состава СКЗИ (cryptcp, certmgr, stunnel, stunnel_msspi, cpverify), используемые в режиме «ручного вызова»[2];
- иные изделия, в составе которых СКЗИ КриптоПро CSP прошло установленным порядком исследования о оценке влияния или тематические исследования (с учетом правил эксплуатации таких изделий).

Проведение исследований по оценке влияния СФ на СКЗИ требуется при одновременном выполнении следующих условий:

- применение СКЗИ и СФ подпадает под случаи, приведенные в п. 3 Положения ПКЗ-2005, или иными нормативными документами определены дополнительные условия применения шифровальных (криптографических) средств;
- в качестве СФ используются программные компоненты из следующего перечня (один пункт или несколько):
  - программные компоненты, использующие приложения (модули) cryptcp, certmgr, stunnel, stunnel_msspi, cpverify в режиме «автоматического вызова»[3];
  - программные компоненты, осуществляющие вызовы (напрямую или опосредовано через различные программные интерфейсы) функций СКЗИ, приведенных в Приложении 2 настоящего документа (подробнее об использовании программных интерфейсов СКЗИ см. разд. 4.2).

При реализации (построении) криптографических протоколов с использованием вызовов функций СКЗИ, приведенных в Приложении 2, объем исследований определяется в Техническом задании.

Проведение тематических исследований СФ со встроенным СКЗИ как самостоятельного шифровального (криптографического) средства требуется при одновременном выполнении следующих условий:

- применение СКЗИ и СФ подпадает под случаи, приведенные в п. 3 Положения ПКЗ-2005, или иными нормативными документами определены дополнительные условия применения шифровальных (криптографических) средств;
- в качестве СФ используются программные компоненты, осуществляющие вызовы (напрямую или опосредовано через различные программные интерфейсы) функций СКЗИ, не приведенных в Приложении 2 настоящего документа.

4.1 Использование СКЗИ в стандартном программном обеспечении

Программное обеспечение СКЗИ позволяет использовать российские криптографические алгоритмы и сертификаты открытых ключей (ключей проверки ЭП) стандарта X.509 совместно со следующим программным обеспечением (включая, но не ограничиваясь):

- Центр Сертификации — Microsoft Certification Authority, входящий в состав Windows Server 2008 R2/2012/2012 R2/2016/2019
- Электронная почта — Microsoft Outlook (из состава Microsoft Office 2007/2010/2013/2016/2019)
- Электронная почта — Microsoft Outlook Express в составе Internet Explorer/Microsoft Edge, Почта Windows (Windows Mail), Live Mail
- Microsoft Word, Excel из состава Microsoft Office 2007/2010/2013/2016/2019 (с помощью плагина КриптоПро Office Signature)
- Средства контроля целостности ПО, распространяемого по сети — Microsoft Authenticode
- Службы терминалов для Windows Server 2008 R2/2012/2012 R2/2016/2019 (включая шлюз служб терминалов)
- Internet Explorer/Microsoft Edge и другие браузеры для защиты TCP/IP соединений в сети Интернет (протокол TLS/SSL при взаимодействии) — web-сервер IIS, TLS-сервер, TLS-клиент

[2]Под режимом «ручного вызова» понимается вызов утилиты из командной строки непосредственно пользователем с визуальной обработкой результатов работы утилиты.

[3]Под режимом «автоматического вызова» понимается вызов утилиты из компонентов прикладной системы, реализованных в исполняемых модулях, скриптах, bat-файлах и т.п.

- Приложение командной строки для формирования запроса на сертификат certreq
- SQL-сервер
- Сервер TMG
- Сервер UAG
- Сервер терминалов и клиент (RDP)

Программное обеспечение СКЗИ также используется совместно с Adobe Reader или Adobe Acrobat для создания/проверки усовершенствованной ЭП pdf-файлов (см. ЖТЯИ.00064-01 90 02. КриптоПро PDF. Руководство по автоматизации создания и проверки электронных подписей).

СКЗИ КриптоПро CSP используется в составе следующих программных продуктов ООО «КРИПТО-ПРО» (включая, но не ограничиваясь):

- КриптоПро УЦ
- Службы УЦ (КриптоПро OCSP, КриптоПро TSP, КриптоПро SVS, КриптоПро Revocation Provider)
- КриптоПро DSS
- КриптоПро HSM
- КриптоПро NGate
- КриптоПро ЭЦП Browser plug-in
- КриптоАРМ
- КриптоПро EFS
- КриптоПро Office Signature
- Secure Pack RUS

Кроме того, обеспечена техническая совместимость СКЗИ с продуктами:

- КриптоПро SSF
- КриптоПро PDF

4.2 Требования при встраивании СКЗИ в прикладные системы

При разработке пользовательских приложений необходимо использовать функции интерфейса CryptoAPI 1.0 и 2.0. Рекомендуется использовать функции, указанные в Приложении 2 настоящего документа. При этом в части описания параметров функций необходимо руководствоваться документацией Microsoft. В части использования специальных флагов и констант, поддерживаемых СКЗИ «КриптоПро CSP», необходимо руководствоваться разделами документа ЖТЯИ.00101-02 96 01. Руководство программиста (CSP_5_0.chm) с эквивалентными по названию CP функциями.

При встраивании СКЗИ КриптоПро CSP в прикладные системы (прикладное ПО, СФ) должны выполняться следующие требования:

1) Сертификаты открытых ключей (ключей проверки ЭП), используемые СКЗИ КриптоПро CSP должны быть выпущены Удостоверяющим центром, сертифицированным ФСБ России по классу защиты не ниже класса защиты используемого СКЗИ.

2) При использовании функций СКЗИ должна быть обеспечена актуальность сертификатов открытых ключей/сертификатов ключей проверки ЭП (с использованием УЦ, построенного на базе Средств УЦ, сертифицированных ФСБ России и правил обработки объектов PKI в СФ).

Актуальность сертификатов открытых ключей/сертификатов ключей проверки ЭП (далее - сертификатов) должна обеспечиваться:

- проверкой корректности ЭП сертификатов из цепочки сертификатов;
- проверкой сроков действия сертификатов из цепочки сертификатов;
- проверка области использования сертификата;
- проверкой сертификатов из цепочки сертификатов на отозванность при помощи списков отозванных (аннулированных) сертификатов (COC, CRL), (или с помощью OCSP-службы);

- проверкой сроков действия СОС (ответа OCSP-службы);
- проверкой корректности ЭП СОС (ЭП ответа OCSP-службы);
- проверкой наличия СОС
- проверкой соответствия сертификата, указанного в полученном ответе OCSP-службы, сертификату, который был указан в соответствующем запросе к OCSP-службе;
- проверкой актуальности сертификата, используемого для проверки ЭП под СОС (сертификата OCSP-службы). В рамках данной проверки выполняются все выше перечисленные проверки;
- при использовании метки времени должны быть проведены аналогичные проверки для сертификата TSP-сервера, а также проверка ЭП ответа TSP-сервера и проверка соответствия хэш-значения ЭП, содержащегося в ответе TSP-сервера, хэш-значению ЭП, содержащемуся в соответствующем запросе TSP-серверу.

3) При вызове функций СКЗИ в обязательном порядке требуется указывать имя и тип провайдера, использование провайдера «по умолчанию» (NULL) не допускается.

4) При вызове функций СКЗИ в прикладном программном обеспечении должна быть предусмотрена проверка кода завершения вызываемой функции с обработкой ошибочных ситуаций и регистрацией событий.

5) В эксплуатационной документации на СФ должна быть предусмотрена необходимость проведения проверки ПО BIOS ПЭВМ, на которых предполагается функционирование СКЗИ и СФ, на соответствие методическим документам ФСБ России в области исследований программного обеспечения BIOS, либо проведение указанной проверки для конкретной конфигурации аппаратной платформы.

6) При встраивании СКЗИ в компоненты СФ, в которых функции создания и/или проверки электронной подписи не являются автоматическими, необходимо проводить оценку соответствия компонентов СФ п.п. 8 и/или 9 «Требований к средствам электронной подписи», утвержденных Приказом ФСБ России от 27 декабря 2011 г. № 796 «Об утверждении Требований к средствам электронной подписи и Требований к средствам удостоверяющего центра».

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

При проведении исследований по оценке влияния должны быть предусмотрены следующие направления исследований (включая, но не ограничиваясь):

- по исходным кодам и эксплуатационной документации на СФ осуществляется анализ выполнения требований и рекомендаций по встраиванию СКЗИ, изложенных в документации на СКЗИ, а также анализ полноты и корректности эксплуатационной документации на компоненты СФ, включая отражение в указанной документации организационно-технических мер по защите информации с использованием СКЗИ и правил обращения с ключевыми документами;
- проверка корректности реализации в СФ работы с криптографическими ключами;
- проверка корректности реализации механизмов контроля целостности СФ и СКЗИ;
- проверка корректности работы криптографических функций в штатном и не штатном режимах и в условиях искаженных и/или отсутствующих объектов PKI;
- проверка механизмов визуализации информации для пользователя (п.п. 8 и/или 9 Приложения 1 к Приказу ФСБ России от 27 декабря 2011 г. № 796 «Об утверждении Требований к средствам электронной подписи и Требований к средствам удостоверяющего центра»);
- в случае опосредованного использования функций программного интерфейса CryptoAPI 2.0 из других программных интерфейсов, необходимо проведение исследований прохождения вызовов и передаваемых данных между вызываемым интерфейсом и интерфейсом CryptoAPI 2.0 (построение трасс вызовов).

5 Требования по защите от НСД

5.1 Общие требования по организации работ по защите от НСД

Защита аппаратного и программного обеспечения от НСД при установке и использовании СКЗИ КриптоПро CSP 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 возможность 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 к ПЭВМ/устройствам, на которых установлены СКЗИ, посторонних лиц, по роду своей деятельности не являющихся персоналом, допущенным к работе с указанными ПЭВМ/устройствами. В случае такой необходимости должен быть обеспечен контроль за их действиями и обеспечена невозможность негативных действий с их стороны на СКЗИ, ПЭВМ/устройства, на которых эксплуатируется СКЗИ, и защищаемую информацию.

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

• Внутренняя планировка, расположение и укомплектованность рабочих мест в помещениях, в которых расположены ПЭВМ с установленным СКЗИ, должны обеспечивать исполнителям работ, сохранность доверенных им конфиденциальных документов и сведений, включая ключевую информацию.

• Размещение СКЗИ 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.

2) Инсталляция СКЗИ КриптоПро CSP 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, допускаются лица, прошедшие соответствующую подготовку и изучившие документацию на соответствующее ПО и на СКЗИ.

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

5) Из состава системы должно быть исключено оборудование, которое может создавать угрозу безопасности ОС. Также необходимо избегать использования нестандартных аппаратных средств, имеющих возможность влиять на функционирование ПЭВМ или ОС.

6) После загрузки ОС должен быть реализован контроль целостности программного обеспечения, входящего в состав СКЗИ, самой ОС и всех исполняемых файлов, функционирующих совместно с СКЗИ, с использованием утилиты cpverify.

7) 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).

8) На ПЭВМ/мобильном устройстве 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.

9) При установке программного обеспечения СКЗИ КриптоПро CSP 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 в 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 в 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 10 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.

• пароль для входа в BIOS должен быть известен только администратору и быть отличным от пароля администратора для входа в ОС.

• в качестве меры по усилению защиты от НСД рекомендуется запретить сохранение паролей, используемых в работе СКЗИ (подробнее см. разд. 3.8.2).

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

• политика паролей ключевых контейнеров может быть настроена с помощью механизмов СКЗИ (подробнее см. разд. 3.8.3).

При эксплуатации СКЗИ **запрещено**:

• оставлять без контроля вычислительные средства, на которых эксплуатируется СКЗИ (в том числе ключевые носители), после ввода ключевой информации либо иной конфиденциальной информации (при оставлении ПЭВМ/мобильного устройства без контроля должна включаться парольная заставка);

• 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 и иные средства отображения информации (за исключением случаев, предусмотренных данными правилами);

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

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

• записывать на ключевые носители постороннюю информацию;

• использовать ключи с истекшим сроком действия;

• 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;

• изменять настройки, установленные программой установки СКЗИ или администратором;

• использовать синхропосылки, вырабатываемые не средствами СКЗИ;

• использовать бывшие в работе ключевые носители для записи новой информации без предварительного уничтожения на них ключевой информации, подлежащей уничтожению, средствами СКЗИ (в соответствии с п. 3.9.3);

• осуществлять несанкционированное вскрытие системных блоков ПЭВМ;

• приносить и использовать в помещении, где размещены средства СКЗИ, радиотелефоны и другую радиопередающую аппаратуру.

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

• в системе регистрируется один пользователь, обладающий правами администратора, на которого возлагается обязанность конфигурировать ОС, настраивать безопасность ОС, а также конфигурировать ПЭВМ/мобильное устройство, на которую установлена ОС;

• не допускается использовать нестандартные, измененные или отладочные версии ОС;

• необходимо исключить возможность загрузки и использования ОС, отличной от предусмотренной штатной работой (например, с помощью настроек BIOS, использования АПМДЗ и т.п.);

• необходимо исключить возможность удаленного управления, администрирования и модификации ОС и её настроек с использованием незащищенного канала связи; для защиты канала связи необходимо использовать средства, сертифицированные ФСБ России по классу не ниже класса используемого СКЗИ;

• на ПЭВМ должна быть установлена только одна операционная система (в случае использования виртуальной инфраструктуры допускается установка одной хостовой и неограниченного числа гостевых ОС);

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

• все неиспользуемые ресурсы системы необходимо удалить или отключить (протоколы, сервисы, порты, общие ресурсы, пользователи и т.п.);

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

• всем пользователям и группам, зарегистрированным в ОС, администратор в соответствии с политикой безопасности, принятой в организации, дает минимально возможные для нормальной работы права; каждый пользователь ОС, не являющийся администратором, может просматривать и редактировать только свои настройки в рамках прав доступа, назначенных ему администратором;

• 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;

• должна быть исключена возможность создания аварийного дампа оперативной памяти, так как он может содержать криптографически опасную информацию;

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

• средствами BIOS должна быть исключена возможность работы на ПЭВМ с СКЗИ, если во время её начальной загрузки не проходят встроенные тесты (POST);

• средствами BIOS должна быть исключена возможность отключения пользователями ISA-устройств и PCI-устройств;

• должна быть исключена возможность создания аварийного дампа оперативной памяти, так как он может содержать криптографически опасную информацию;

• после инсталляции ОС следует установить все рекомендованные программные обновления и программные обновления, связанные с безопасностью, существующие на момент инсталляции 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/устройствах, 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;

• в настройках ОС, отвечающих за ведение журналов событий, необходимо установить режим архивирования журнала при его заполнении;

• организовать и использовать комплекс мероприятий антивирусной защиты;

• должно быть запрещено использование СКЗИ для защиты речевой информации;

• должны быть отключены радиоканалы, неиспользуемые в работе СКЗИ.

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

5.5 Требования по обеспечению физической безопасности сервера

Следует исключить возможность доступа неавторизованного персонала к консоли, системе питания и дополнительным устройствам, подключенным к защищаемому серверу путем установки оборудования в специально выделенное и запираемое помещение (аппаратную или серверную комнату).

Доступ персонала в серверную комнату должен быть регламентирован внутренним распорядком эксплуатирующей организации и должностными инструкциями.

Для исключения сбоев компьютера, вызванных отключением электропитания, необходимо обеспечить электропитание сервера от источника бесперебойного питания достаточной мощности. Как минимум, мощности батарей источника бесперебойного питания должно хватать на время достаточное для корректного автоматического завершения работы сервера.

5.6 Требования по организации процедуры резервного копирования и хранения резервных копий

При определении регламента резервного копирования и хранения резервных копий следует обеспечить ответственное хранение резервных копий в запираемых сейфах (шкафах) и определить процедуру выдачи резервных копий ответственному персоналу и уничтожения вышедших из употребления носителей.

Стандартными мерами по организации ответственного хранения носителей являются:

- маркировка носителей;
- составление описи хранимых носителей с указанием серийных (инвентарных) номеров, дат записи носителей, фамилией сотрудника, создавшего копию для каждого сейфа (шкафа);
- периодическая сверка описи и содержимого сейфов (шкафов);
- организация ответственного хранения и выдачи ключей от сейфов (шкафов);
- возможное опечатывание (опломбирование) сейфов (шкафов);
- уничтожение вышедших из употребления носителей производится комиссией с составлением акта об уничтожении.

5.7 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 определена только после проведения дополнительных исследований с оценкой невозможности 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.

5.8 Требования по использованию в СКЗИ аппаратно-программных средств защиты от НСД

Примечание. Использование аппаратно-программного средства защиты от НСД при работе с СКЗИ «КриптоПро CSP» версия 5.0 R2 KC1 исполнение 1-Base необязательно.

Аппаратно-программные 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.

Аппаратно-программные 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 аппаратно-программного средства защиты от НСД на АРМ пользователя должна производиться в соответствии с эксплуатационной документацией. Перед эксплуатацией необходимо ознакомиться с комплектом документации на используемое средство и принять рекомендуемые в документации защитные организационные меры.

Настройка аппаратно-программного 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 Требования по криптографической защите

Должны выполняться следующие требования по криптографической защите:

1) Настройки операционных систем для работы с СКЗИ должны производиться в соответствии с документами:

- ЖТЯИ.00101-02 91 02 Руководство администратора безопасности. Windows
- ЖТЯИ.00101-02 91 03 Руководство администратора безопасности. Linux
- ЖТЯИ.00101-02 91 04 Руководство администратора безопасности. FreeBSD
- ЖТЯИ.00101-02 91 05 Руководство администратора безопасности. Solaris
- ЖТЯИ.00101-02 91 06 Руководство администратора безопасности. AIX
- ЖТЯИ.00101-02 91 07 Руководство администратора безопасности. Mac OS
- ЖТЯИ.00101-02 91 08 Руководство администратора безопасности. iOS
- ЖТЯИ.00101-02 91 09 Руководство администратора безопасности. Виртуальные среды
- ЖТЯИ.00101-02 91 10 Руководство администратора безопасности. Sailfish
- ЖТЯИ.00101-02 91 11 Руководство администратора безопасности. Android

2) При установке ПО СКЗИ должен быть обеспечен контроль целостности и достоверность дистрибутива СКЗИ и совместно поставляемых с СКЗИ компонент СФ. После завершения процесса установки должны быть выполнены действия, необходимые для осуществления периодического контроля целостности установленного ПО СКЗИ, а также его окружения.

3) Контролем целостности должны быть охвачены файлы, указанные в разделах «Требования по криптографической защите» документов ЖТЯИ.00101-02 91 02, ЖТЯИ.00101-02 91 03, ЖТЯИ.00101-02 91 04, ЖТЯИ.00101-02 91 05, ЖТЯИ.00101-02 91 06, ЖТЯИ.00101-02 91 07, ЖТЯИ.00101-02 91 08, ЖТЯИ.00101-02 91 09, ЖТЯИ.00101-02 91 10, ЖТЯИ.00101-02 91 11

4) Если проверка целостности компонентов СКЗИ завершается ошибкой, администратор безопасности должен выявить причину и обстоятельства нарушения целостности СКЗИ и переустановить СКЗИ в соответствии с инструкцией по установке, описанной в руководстве администратора безопасности для используемой программно-аппаратной платформы.

5) Периодичность тестового контроля криптографических функций — 10 минут.

6) Периодичность перезагрузки ПЭВМ — 7 дней.

7) Периодичность контроля системы охлаждения процессорного блока ПЭВМ — 1 месяц.

8) Запрещается использовать режим простой замены (ECB) ГОСТ 28147-89, ГОСТ Р 34.13-2015 (ГОСТ 34.13-2018) для шифрования информации, кроме ключевой.

9) Должно даваться предупреждение о том, что при использовании режима шифрования CRYPT_SIMPLEMIX_MODE материал, обрабатываемый на одном ключе, автоматически ограничивается величиной 4 MB.

10) При функционировании СКЗИ должны выполняться требования эксплуатационной документации на используемый ПАК защиты от НСД.

11) Запрещается использование беспроводных клавиатур и компьютерных мышей.

7 Разбор конфликтных ситуаций, связанных с применением ЭП

Применение электронной подписи в автоматизированной системе может приводить к конфликтным ситуациям, заключающимся в оспаривании сторонами (участниками системы) авторства и/или содержимого документа, подписанного электронной подписью.

Разбор подобных конфликтных ситуаций требует применения специального программного обеспечения для выполнения проверок и документирования данных, используемых при выполнении процедуры проверки соответствия ЭП содержимому электронного документа.

Разбор конфликтной ситуации заключается в доказательстве авторства подписи конкретного электронного документа конкретным исполнителем.

Данный разбор основывается на математических свойствах алгоритмов ЭП, реализованных в соответствии со стандартами РФ ГОСТ Р 34.10-2001, ГОСТ Р 34.10-2012 (ГОСТ 34.10-2018), ГОСТ Р 34.11-2012 (ГОСТ 34.11-2018) и ГОСТ Р 34.11-94, гарантирующих невозможность подделки значения ЭП любым лицом, не обладающим закрытым ключом подписи.

При проверке значения ЭП используется ключ проверки ЭП, значение которого вычисляется по значению ключа ЭП при их формировании.

В системе должны быть предусмотрены средства ведения архивов электронных документов с ЭП и сертификатов ключей проверки ЭП.

Разбор конфликтной ситуации выполняется комиссией, состоящей из представителей сторон, службы безопасности и экспертов. Состав комиссии, порядок ее формирования, регламент работы, рассмотрение результатов определяется в приложении к Регламенту (Договору), заключаемому между участниками информационного обмена.

Оспаривание результатов работы комиссии и возмещение пострадавшей стороне принесенного ущерба выполняется в установленном действующим законодательством Российской Федерации порядке.

7.1 Порядок разбора конфликтной ситуации

Разбор конфликтной ситуации выполняется по инициативе любого участника информационного обмена и состоит из:

1) предъявления претензии одной стороны другой;
2) формирования комиссии;
3) разбора конфликтной ситуации;
4) принятие мер по урегулированию конфликта.

Разбор конфликтной ситуации проводится с использованием программного обеспечения СКЗИ КриптоПро CSP для электронного документа, авторство или содержание которого оспаривается.

Проверка подписанного электронного документа включает в себя выполнение следующих действий:

1) определение сертификата или нескольких сертификатов, необходимых для проверки ЭП;
2) проверка ЭП электронного документа с использованием каждого сертификата;
3) определение даты создания каждой ЭП в электронном документе;
4) проверка ЭП каждого сертификата, путем построения цепочки сертификатов до сертификата Главного ЦС;
5) проверка действительности сертификатов на текущий момент времени;
6) проверка действительности сертификатов на момент создания ЭП;
7) проверка отсутствия сертификатов в СОС.

При проверке ЭП документа, верификации цепочки сертификатов, отсутствии сертификата в СОС, авторство подписи под документом считается установленным.

Примечание. Несовпадение даты формирования документа и сроков действия сертификата и/или сроков действия ключа ЭП не влияют на определение авторства документа. В таком случае можно сделать предположение о несоблюдении пользователем Регламента (Договора) в части сроков действия ключей, сертификатов или некорректного использования сертификата в прикладном ПО.

7.2 Случаи невозможности проверки значения ЭП

При отсутствии в архиве сертификата открытого ключа (ключа проверки ЭП) пользователя, выполнившего ЭП, доказать авторство документа невозможно. В связи с этим, архив с сертификатами открытых ключей необходимо подвергать регулярному резервному копированию и хранить в течение всего установленного срока хранения.

Литература

[1] Приказ ФАПСИ от 13.06.2001 N 152 «Об утверждении 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». URL: http://base.garant.ru/183628/.

[2] Приказ ФСБ РФ от 09.02.2005 № 66 «Об утверждении Положения о разработке, производстве, реализации и эксплуатации шифровальных (криптографических) средств защиты информации (Положение ПКЗ-2005)». URL: http://base.garant.ru/187947/.

[3] Министерство связи и массовых коммуникаций Российской Федерации. Уведомление об организации перехода на использование схемы электронной подписи по ГОСТ Р 34.10-2012. URL: https://sc-new.minsvyaz.ru/web/guest/-/opublikovano-uvedomlenie-ob-organizacii-perehoda-na-ispol-zovanie-shemy-elektronnoj-podpisi-po-gost-r-34-10-2012.

[4] Постановление Правительства РФ от 16.04.2012 N 313 «Об утверждении Положения о лицензировании деятельности по разработке, производству, распространению шифровальных (криптографических) средств, информационных систем и телекоммуникационных систем, защищенных с использованием шифровальных (криптографических) средств, выполнению работ, оказанию услуг в области шифрования информации, техническому обслуживанию шифровальных (криптографических) средств, информационных систем и телекоммуникационных систем, защищенных с использованием шифровальных (криптографических) средств (за исключением случая, если техническое обслуживание шифровальных (криптографических) средств, информационных систем и телекоммуникационных систем, защищенных с использованием шифровальных (криптографических) средств, осуществляется для обеспечения собственных нужд юридического лица или индивидуального предпринимателя)». URL: http://www.consultant.ru/document/cons_doc_LAW_128739/.

[5] Перечень организаций, аккредитованных в качестве испытательных лабораторий в Системе сертификации средств криптографической защиты информации. URL: http://clsz.fsb.ru/accred.htm.

Приложение 1
Контроль целостности программного обеспечения

Контроль целостности программного обеспечения с помощью алгоритмов хэширования

Модуль cpverify 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 (опция -rv).

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

1) Вычисление значения хэш-функции для файла

```
cpverify -mk filename [-alg algid] [-inverted_halfbytes <inv>]
```

- filename — имя файла, для которого производится вычисление значения хэш-функции
- algid — используемый алгоритм хэширования:
  - GR3411 (по умолчанию)
  - GR3411_2012_256
  - GR3411_2012_512
- -inverted_halfbytes <inv> — указывается, если полубайты в хэш-значении hashvalue перевернуты. Значения inv по умолчанию:
  - 1 — для алгоритма GR3411
  - 0 — для алгоритмов GR3411_2012_256 и GR3411_2012_512

2) Проверка целостности файла

```
cpverify filename [-alg algid] [hashvalue] [-inverted_halfbytes <inv>]
```

- filename — имя файла, целостность которого проверяется
- algid — используемый алгоритм хэширования:
  - GR3411 (по умолчанию)
  - GR3411_2012_256
  - GR3411_2012_512
- hashvalue — значение хэш-функции. Если hashvalue не указан, то хэш-значение берется из файла filename.hsh
- -inverted_halfbytes <inv> — указывается, если полубайты в хэш-значении hashvalue перевернуты. Значения inv по умолчанию:
  - 1 — для алгоритма GR3411
  - 0 — для алгоритмов GR3411_2012_256 и GR3411_2012_512

3) Вычисление значения хэш-функции для каждого из файлов, содержащихся в каталоге в разделе реестра

```
cpverify -rm [-alg algid] [catname]
```

- catname — имя каталога в разделе реестра (если catname не указан, то будут пересчитаны все хэш-значения в разделе реестра)
- algid — используемый алгоритм хэширования:

  * GR3411 (по умолчанию)
  * GR3411_2012_256
  * GR3411_2012_512

Примечание. Текущее значение хэш-функций заменяется на вновь посчитанное.

4) Проверка целостности файлов из каталога в разделе реестра

```
cpverify -rv [catname]
```

• catname — имя каталога в разделе реестра (если catname не указан, то будут проверены все файлы в разделе реестра)

5) Вычисление значения хэш-функции для файла и запись полученного значения в реестр

```
cpverify -addreg -file filename
```

• filename — имя файла, для которого производится вычисление значения хэш-функции

6) Удаление значения хэш-функции из реестра

```
cpverify -delreg -file filename
```

• filename — имя файла, для которого значение хэш-функции необходимо удалить из реестра

7) Вычисление значения хэш-функции для файлов, перечисленных в xml-файле, и запись полученных хэш-значений в файл

```
cpverify -xm in_file out_file [-alg algid] [xmlcatname]
```

• in_file — имя xml-файла с перечнем файлов, для которых производится вычисление значений хэш-функции

• xmlcatname — имя каталога в xml-файле с именем in_file с перечислением файлов, для которых производится вычисление значений хэш-функции (если xmlcatname не указан, то хэш-значения будут посчитаны для всех фалов, перечисленных в xml-файле с именем in_file)

• out_file — имя xml-файла, в который записываются вычисленные значения хэш-функции

• algid — используемый алгоритм хэширования:
  * GR3411 (по умолчанию)
  * GR3411_2012_256
  * GR3411_2012_512

Примечание. Текущее значение хэш-функций заменяется на вновь посчитанное.

8) Проверка целостности файлов, перечисленных в xml-файле

```
cpverify -xv in_file [xmlcatname]
```

• in_file — имя xml-файла с перечнем файлов, для выполняется проверка целостности

• xmlcatname — имя каталога в xml-файле с именем in_file с перечислением файлов, для которых выполняется проверка целостности (если xmlcatname не указан, то проверка будет выполнена для всех файлов, перечисленных в xml-файле с именем in_file)

9) Формирование xml-файла, содержащего список файлов, находящихся в разделе реестра под контролем целостности, и хэш-значения этих файлов

```
cpverify -r2x out_file
```

- `out_file` — имя xml-файла с перечнем файлов и хэш-значениями

Примечание. Список контролируемых модулей зависит от версии и исполнения СКЗИ.

10) Установка под контроль целостности файлов, перечисленных в xml-файле

```
cpverify -x2r in_file
```

- `in_file` — имя xml-файла с перечнем файлов

Примечание. Список контролируемых модулей может быть получен при помощи команды `cpverify -r2x out_file`.

Алгоритм действий

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 name="xmlcatname">
        <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 name="xmlcatname">
        <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

1) Проверка подписи файла

```
cpverify -file_verify filename [signval] -timestamp date
```

- `filename` — имя файла, для которого производится проверка подписи
- `signval` — значение подписи, байтовая строка (если параметр signval не указан, то значение подписи берется из файла filename.sgn)
- `date` — дата формирования подписи в формате ДД.ММ.ГГГГ

Примечание. 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».

2) Создание подписи файла

```
cpverify -file_sign filename -cont cont_name [-pin password]
   [-provname имя_провайдера] [-provtype тип_провайдера]
```

- `filename` — имя файла, для которого производится формирование подписи
- `cont_name` — контейнер ключа, который используется для формирования подписи
- `password` — пароль контейнера ключа
- `Provname` — имя используемого провайдера
- `Provtype` — тип используемого провайдера:
  - 75 (по умолчанию)
  - 80
  - 81

Приложение 2
Перечень вызовов, использование которых при разработке систем на основе СКЗИ «КриптоПро CSP» версия 5.0 R2 КС1 исполнение 1-Base с учетом п.1.5 Формуляра ЖТЯИ.00101-02 30 01 возможно без дополнительных тематических исследований

Интерфейс CryptoAPI

Функция	Описание	Ограничения на использование функции
Функции инициализации и настройки провайдера		
`CryptAcquireContext`	Функция 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 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`	Функция управляет счетчиком дескрипторов созданного CryptAcquireContext.	
`CryptEnumProviders`	Функция перечисления установленных криптопровайдеров.	
`CryptEnumProviderTypes`	Функция перечисления установленных типов криптопровайдеров.	
`CryptGetDefaultProvider`	Функция получения контекста провайдера, установленного в системе по умолчанию.	
`CryptGetProvParam`	Функция получает параметры криптопровайдера.	
`CryptSetProvParam`	Функция 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 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 удаляет ключ, передаваемый через параметр hKey. После удаления ключ (дескриптор ключа) не может использоваться.	
CryptExportKey	Функция 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 заполняет буфер случайными байтами.	
CryptGetKeyParam	Функция возвращает параметры ключа.	
CryptGetUserKey	Функция 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 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 устанавливает параметры ключа.	Разрешено использование только со следующими символьными аргументами: KP_CERTIFICATE, KP_CIPHEROID, KP_DHOID, KP_HASHOID.
Функции обработки криптографических сообщений		
CryptSignMessage	Функция создает хэш определенного содержания, подписывает хэш и затем производит кодирование текста исходного сообщения, и подписанного хэша.	
CryptVerifyMessageSignature	Функция проверяет электронную подпись сообщения.	
CryptVerifyDetachedMessageSignature	Функция проверяет подписанное сообщение, содержащее отсоединенную (detached) подпись или подписи.	
CryptDecodeMessage	Функция декодирует, расшифровывает и проверяет сообщение.	
CryptDecryptAndVerifyMessageSignature	Функция декодирует и проверяет сообщение.	
CryptEncryptMessage	Функция зашифровывает и производит кодирование сообщения. Аутентичность сообщения не обеспечивается.	
CryptDecryptMessage	Функция производит декодирование и расшифрование сообщения. Проверка аутентичности сообщения не производится. 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.	
CryptVerifyMessageSignatureWith-Key	Функция проверяет подписанное сообщение.	
CryptMsgCalculateEncodedLength	Функция вычисляет максимальное количество байтов, необходимое для кодированного криптографического сообщения, заданного типом сообщения, параметрами кодирования и общей длиной информации, которая должна быть кодирована.	
CryptMsgOpenToEncode	Функция открывает криптографическое сообщение для кодирования и возвращает дескриптор открытого сообщения.	
CryptMsgOpenToDecode	Функция открывает криптографическое сообщение для декодирования и возвращает дескриптор открытого сообщения.	
CryptMsgUpdate	Функция пополняет текст криптографического сообщения.	
CryptMsgGetParam	Функция получает параметр сообщения после того, как криптографическое сообщение было кодировано или декодировано.	
CryptMsgControl	Функция выполняет контрольное действие.	
CryptMsgClose	Функция закрывает дескриптор криптографического сообщения.	
CryptMsgDuplicate	Функция дублирует дескриптор криптографического сообщения путем увеличения счетчика ссылок.	
Функции работы с алгоритмами хэширования		

CryptCreateHash	Функция 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. Дескриптор, полученный с использованием аргумента CALG_GR3411, запрещено подавать в функцию CryptSignHash. 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 удаляет объект функции хэширования.	
CryptDuplicateHash	Функция 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 возвращает параметры объекта функции хэширования и значение функции хэширования.	Запрещено использование с символьными аргументами HP_OPAQUEBLOB, HP_HASHSTATEBLOB.
CryptHashCertificate	Функция предназначена для вычисления хэш-функции от всего содержимого переданного буфера данных сертификата (включая электронную подпись этого сертификата).	
CryptHashData	Функция передаёт данные указанному объекту функции хэширования.	

CryptSetHashParam	Функция устанавливает параметры объекта хэширования.	Разрешено использование только со следующими 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 возвращает значение электронной подписи от значения функции хэширования.	Запрещено использовать дескриптор, полученный вызовом функции CryptCreateHash с использованием аргумента CALG_GR3411.
CryptVerifySignature	Функция 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 добавляет контекст СОС в хранилище сертификатов.	
CertAddCRLLinkToStore	Функция создает ссылку на список аннулированных сертификатов в другом хранилище.	
CertAddEncodedCRLToStore	Функция создает контекст СОС из кодированного СОС и добавляет его в хранилище сертификатов. Функция создает копию контекста СОС перед добавлением его в хранилище. CertEnumCRLsInStore Функция CertEnumCRLsInStore получает первый или следующий СОС в хранилище. Эта функция используется в цикле для того, чтобы последовательно получить все СОС в хранилище.	

CertFreeCRLContext	Функция освобождает контекст СОС, уменьшая счетчик ссылок на единицу. Когда счетчик ссылок обнуляется, функция CertFreeCRLContext освобождает память, выделенную под контекст СОС.	
CertCreateCRLContext	Функция создает контекст СОС из кодированного СОС. Созданный контекст не помещается в хранилище сертификатов. В созданном контексте функция размещает копию кодированного СОС.	
CertDeleteCRLFromStore	Функция удаляет список аннулированных сертификатов из хранилища.	
CertDuplicateCRLContext	Функция дублирует контекст СОС, увеличивая счетчик ссылок на СОС на единицу.	
CertFindCRLInStore	Функция находит первый или следующий контекст СОС 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 СОС в хранилище сертификатов, удовлетворяющие заданному критерию поиска.	
CertDeleteCertificateFromStore	Функция удаляет определенный контекст СОС из хранилища сертификатов.	
CertFindCertificateInCRL	Функция осуществляет поиск заданного сертификата в списке аннулированных сертификатов.	
CertGetCRLFromStore	Функция получает первый или следующий контекст СОС для определенного издателя сертификата из хранилища сертификатов. Эта функция также осуществляет возможную проверку СОС.	
CertSerializeCRLStoreElement	Функция сериализации списка аннулированных сертификатов со своими свойствами.	
CertVerifyCRLRevocation	Функция предназначена для проверки наличия сертификата в указанном CRL.	

CertVerifyCRLTimeValidity	Функция предназначена для проверки актуальности списка аннулированных сертификатов (CRL) на момент времени, указанный через входной параметр.	Перед вызовом функции требуется выполнять проверку на неравенство NULL параметра pCrlInfo.
Расширенные свойства сертификата, списка отозванных (аннулированных) сертификатов (COC) и CTL		
CertGetCRLContextProperty	Функция получает расширенные свойства определенного контекста COC.	
CertSetCRLContextProperty	Функция устанавливает расширенные свойства определенного контекста COC.	
CertGetCertificateContextProperty	Функция получает информацию, содержащуюся в расширенных свойствах контекста сертификата.	
CertEnumCertificateContextProperties	Функция позволяет перечислить информацию, содержащуюся в расширенных свойствах контекста сертификата.	
CertSetCertificateContextProperty	Функция устанавливает расширенные свойства для определенного контекста сертификата.	
CertEnumCRLContextProperties	Функция перечисления расширенных свойств списка аннулированных сертификатов.	
CertEnumCTLContextProperties	Функция перечисления расширенных свойств CTL.	
CertGetCTLContextProperty	Функция получения расширенного свойства CTL.	
CertSetCTLContextProperty	Функция установки расширенных свойств CTL.	
CertAddCTLContextToStore	Функция добавляет контекст CTL в хранилище сертификатов.	
CertCreateCTLContext	Функция создает контекст кодированного CTL. Созданный контекст не помещается в хранилище сертификатов. Функция делает копию CTL в созданном контексте.	
CertDuplicateCTLContext	Функция дублирует контекст CTL, увеличивая счетчик ссылок на CTL на единицу.	

CertFreeCTLContext	Функция освобождает контекст CTL, уменьшая счетчик ссылок на единицу. Когда счетчик ссылок обнуляется, функция CertFreeCTLContext освобождает память, выделенную под контекст CTL.	
Сертификаты		
CertAddCertificateContextToStore	Функция добавляет контекст сертификата в хранилище сертификатов.	
CertAddCertificateLinkToStore	Функция добавляет ссылку на сертификат в другом хранилище.	
CertAddEncodedCertificateToStore	Функция создает контекст сертификата из кодированного сертификата и добавляет его в хранилище сертификатов. Созданный контекст не содержит никаких расширенных свойств.	
CertEnumCertificatesInStore	Функция 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 освобождает контекст сертификата, уменьшая счетчик ссылок на единицу.	
CertCreateCertificateContext	Функция создает контекст сертификата из кодированного сертификата. Созданный контекст не помещается в хранилище сертификатов. В созданном контексте функция размещает копию кодированного сертификата.	
CertDuplicateCertificateContext	Функция дублирует контекст сертификата, увеличивая счетчик ссылок на единицу.	
CertFindCertificateInStore	Функция находит первый или следующий контекст сертификата в хранилище сертификатов, который соответствует критерию поиска, определяемому параметром dwFindType и связанным с ним pvFindPara.	

CertDeleteCertificateFromStore	Функция удаляет определенный контекст сертификата из хранилища сертификатов.	
CertGetSubjectCertificateFromStore	Функция получает контекст сертификата из хранилища сертификатов, однозначно определяемый его издателем и серийным номером.	
CertGetIssuerCertificateFromStore	Функция поиска сертификатов издателей заданного сертификата.	
CertGetValidUsages	Функция поиска пересечения KeyUsage для массива сертификатов.	
CertSerializeCertificateStoreElement	Функция предназначена для сериализации кодированных объектов (сертификата, списка отозванных сертификатов (CRL) и списка доверенных сертификатов (CTL)) и их свойств, расположенных в контексте объекта.	Перед вызовом функции требуется выполнять проверку на неравенство NULL параметров pCertContext, pbElement и pcbElement.
CertComparePublicKeyInfo	Функция сравнивает два открытых ключа и определяет, являются ли они идентичными.	
CertFindExtension	Функция находит расширение в массиве и возвращает указатель на него.	
CertGetPublicKeyLength	Фукнция возвращает длину ключа в битах.	
CertGetIntendedKeyUsage	Функция получает назначение ключа из сертификата.	
CertCompareCertificateName	Функция сравнивает два сертификата и определяет, являются ли они идентичными.	

CryptSignCertificate	Функция предназначена для формирования значения хэш-функции/электронной подписи для поля «To Be Signed» в кодированном объекте X509 (которым может являться сертификат, список отзыва сертификатов, список доверенных сертификатов или запрос на сертификат).	Для формирования электронной подписи разрешается передача по указателю pSignatureAlgorithm структуры, содержащей только следующие значения OID-идентификаторов: • szOID_CP_GOST_R3411_12_256_R3410 ("1.2.643.7.1.1.3.2") • szOID_CP_GOST_R3411_12_512_R3410 ("1.2.643.7.1.1.3.3") Для вычисления хэш-функции разрешается передача по указателю pSignatureAlgorithm структуры, содержащей только следующие возможные варианты значения OID-идентификаторов: • szOID_CP_GOST_R3411_12_256 ("1.2.643.7.1.1.2.2") • szOID_CP_GOST_R3411_12_512 ("1.2.643.7.1.1.2.3")
CryptSignAndEncodeCertificate	Функция предназначена для формирования значения хэш-функции/электронной подписи для поля «To Be Signed» в кодированном объекте X509 (которым может являться сертификат, список отзыва сертификатов, список доверенных сертификатов или запрос на сертификат), кодирует результат вычисления и размещает в объекте X509.	Разрешается использование функции только для формирования запроса на сертификат с передачей в качестве параметра lpszStructType только символьного значения X509_CERT_REQUEST_TO_BE_SIGNED (0x3). Разрешается передача по указателю pSignatureAlgorithm структуры, содержащей только следующие возможные значения OID-идентификаторов: • szOID_CP_GOST_R3411_12_256_R3410 ("1.2.643.7.1.1.3.2") • szOID_CP_GOST_R3411_12_512_R3410 ("1.2.643.7.1.1.3.3")
CertVerifyTimeValidity	Функция предназначена для проверки актуальности срока действия сертификата на момент времени, указанный через входной параметр.	Перед вызовом функции требуется выполнять проверку на неравенство NULL параметра pCertInfo.
CertVerifyValidityNesting	Функция предназначена для проверки того, что срок действия указанного сертификата не выходит за рамки срока действия сертификата издателя.	Перед вызовом функции требуется выполнять проверку на неравенство NULL параметров pSubjectInfo и pIssuerInf.
CryptVerifyCertificateSignature	Функция предназначена для проверки электронной подписи в сертификате, списке отозванных сертификатов или запросе на сертификат с использованием открытого ключа заявителя.	Допускается использование только совместно с проверкой всей цепочки сертификатов для обрабатываемого объекта PKI.

CryptVerifyCertificateSignatureEx	Функция предназначена для проверки электронной подписи в сертификате, списке отозванных сертификатов, запросе на сертификат или подписанном ответе от OCSP-сервера с использованием как объекта открытого ключа заявителя, так и сертификата/цепочки сертификатов.	Допускается использование только совместно с проверкой всей цепочки сертификатов для обрабатываемого объекта PKI.
CryptGetObjectUrl	Функция предназначена для получения в отношении сертификата, списка доверенных сертификатов (CTL) или списка отозванных сертификатов (CRL) пути URL удалённого объекта.	
OCSP		
CertAddRefServerOcspResponse	Функция увеличения счетчика ссылок на OCSP ответ.	
CertAddRefServerOcspResponseContext	Функция увеличения счетчика ссылок на контекст OCSP ответа.	
CertCloseServerOcspResponse	Функция закрытия дескриптора OCSP ответа.	
CertGetServerOcspResponseContext	Функция получения контекста OCSP ответа.	
CertOpenServerOcspResponse	Функция открытия дескриптора OCSP ответа для заданной цепочки сертификатов.	
Оконные функции		
CertSelectCertificate	Функция отображения диалога выбора сертификата по заданным критериям.	
CryptUIDlgCertMgr	Функция отображения диалога управления сертификатами.	
CryptUIDlgSelectCertificate	Функция отображения диалога выбора сертификата.	
CryptUIDlgSelectCertificateFromStore	Функция отображения диалога выбора сертификата из хранилища.	
CryptUIDlgViewCertificate	Функция отображения диалога со свойствами сертификата.	
CryptUIDlgViewContext	Функция отображения сертификата, списка аннулированных сертификатов или CTL.	
CryptUIDlgViewSignerInfo	Функция отображения диалога с информацией о подписавшем.	

CertSelectionGetSerializedBlob	Функция сериализации сертификата из структуры, используемой для отображения.	
GetFriendlyNameOfCert	Функция преобразования имени сертификата к «читаемому» виду.	
	Проверка цепочек сертификатов	
CertVerifyCertificateChainPolicy	Функция проверяет цепочку сертификатов на достоверность, включая соответствие критерию истинности.	
CertGetCertificateChain	Функция строит цепочку сертификатов, начиная с последнего сертификата, в обратном направлении до доверенного корневого сертификата, если это возможно.	
CertFreeCertificateChain	Функция освобождает цепочку сертификатов путем уменьшения счетчика ссылок. Если счетчик ссылок равен нулю, то память, выделенная под цепочку, освобождается.	
CertCreateCertificateChainEngine	Функция создает контекст HCERTCHAINENGINE, который позволяет изменять параметры механизма построения цепочки сертификатов. Позволяет ограничивать множество доверенных сертификатов.	
CertFreeCertificateChainEngine	Функция освобождает контекст HCERTCHAINENGINE.	
CertCreateCTLEntryFromCertificateContextProperties	Функция создания CTL на основе свойств атрибутов контекста сертификата.	
CertDuplicateCertificateChain	Функция дублирования 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	Функция установки свойств в контекст сертификата на основе CTL.	

Расширенные свойства сертификата (EKU)		
`CertGetEnhancedKeyUsage`	Функция получает информацию о расширенном использовании ключа из соответствующего расширения или из расширенных свойств сертификата. Расширенное использование ключа служит признаком правомерного использования сертификата.	
`CryptAcquireCertificatePrivateKey`	Функция получает дескриптор HCRYPTPROV и параметр dwKeySpec для определенного контекста сертификата.	
Идентификаторы		
`CryptFindOIDInfo`	Функция получает первую предопределенную или зарегистрированную структуру CRYPT_OID_INFO, согласованную с определенным типом ключа и с ключом.	
`CryptEnumOIDInfo`	Функция перечисления зарегистрированных идентификаторов и получение информации для них.	
Хранилище сертификатов		
`CertOpenStore`	Функция открывает хранилище сертификатов, используя заданный тип провайдера.	
`CertDuplicateStore`	Функция дублирует дескриптор хранилища, увеличивая счетчик ссылок на хранилища на единицу.	
`CertOpenSystemStore`	Функция используется для открытия наиболее часто используемых хранилищ сертификатов.	
`CertCloseStore`	Функция закрывает дескриптор хранилища сертификатов и уменьшает счетчик ссылок на хранилища на единицу.	
`CertAddStoreToCollection`	Функция добавления хранилища в коллекцию.	
`CertControlStore`	Функция установки нотификации при различиях в закешированном хранилище и физическом хранилище.	
Работа с открытыми данными и объектами		

CryptImportPublicKeyInfoEx2	Функция импортирует информацию об открытом ключе в CNG и возвращает дескриптор открытого ключа.	
CryptImportPublicKeyInfoEx	Функция импортирует информацию об открытом ключе в CSP и возвращает дескриптор открытого ключа.	
CryptImportPublicKeyInfo	Функция преобразовывает и импортирует информацию об открытом ключе в провайдер и возвращает дескриптор открытого ключа.	
CryptExportPublicKeyInfoEx	Функция экспортирует информацию об открытом ключе, связанную с соответствующим закрытым ключом провайдера.	
CryptExportPublicKeyInfo	Функция экспортирует информацию об открытом ключе, ассоциированную с соответствующим закрытым ключом провайдера.	
CertCompareCertificate	Функция сравнивает два сертификата для того, чтобы определить, являются ли они идентичными.	
CertCompareIntegerBlob	Функция сравнивает два целочисленных блоба для определения того, представляют ли они собой два равных числа.	
CryptExportPublicKeyInfoFromBCryptKeyHandle	Функция экспортирует информацию об открытом ключе, ассоциированную с соответствующим закрытым ключом провайдера.	
Кодирование/декодирование		
CryptDecodeObject	Функция используется для декодирования сертификатов, списков аннулированных сертификатов (СОС) и запросов на сертификаты.	
CryptDecodeObjectEx	Функция используется для декодирования сертификатов, списков аннулированных сертификатов и запросов на сертификаты.	
CryptEncodeObject	Функция используется для кодирования сертификатов, списков аннулированных сертификатов и запросов на сертификаты.	

CryptEncodeObjectEx	Функция используется для кодирования сертификатов, списков аннулированных сертификатов и запросов на сертификаты.	
Получение объектов из удаленных источников		
CryptRetrieveObjectByUrlA	Функция получает объект инфраструктуры открытых ключей по заданному URL.	
CryptRetrieveObjectByUrlW	Функция 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.	
CryptQueryObject	Функция предназначена для получения сведений о содержимом объекта интерфейса CryptoAPI (например, сертификате, списке аннулированных сертификатов или списке доверенных сертификатов) в формате соответствующего хэндла.	
CertFindAttribute	Функция производит поиск атрибута сертификата по идентификатору.	
CertGetNameString	Функция получает имя владельца или издателя сертификата.	
CertNameToStr	Функция производит декодирование 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).	
CryptMsgVerifyCountersignature-Encoded	Функция проверяет добавочную подпись (декодирует структуру SignerInfo, как определено в PKCS#7).	
CryptMsgVerifyCountersignature-EncodedEx	Функция проверяет добавочную подпись (декодирует структуру SignerInfo, как определено в PKCS#7).	
CryptMemFree	Функция предназначена для освобождения памяти, которая была ранее выделена с использованием CryptMemAlloc или CryptMemRealloc.	
CertStrToName	Функция предназначена для трансформации X.500-строки с завершающим NULL-символом в кодированное имя сертификата.	
SignError	Функция предназначена для получения кода последней ошибки в формате HRESULT.	
SignerFreeSignerContext	Функция предназначена для освобождения памяти, содержащей кодированные подписанные данные файла.	

Интерфейс cpcurl

Функция	Описание	Ограничения на использование функции
Функции установки TLS-соединения и приёма/передачи данных		
curl_global_init	Функция инициализации работы библиотеки.	Не является потокобезопасной, должна быть вызвана единожды перед началом использования библиотеки. После завершения использования библиотеки необходимо вызвать curl_global_cleanup.
curl_global_cleanup	Функция деинициализации работы библиотеки.	Не является потокобезопасной, должна быть вызвана единожды после завершения использования библиотеки.
curl_easy_init	Функция инициализации сессии работы с библиотекой.	Является потокобезопасной. Полученный данным вызовом хэндл должен быть после завершения работы с ним освобождён вызовом curl_easy_cleanup.
curl_easy_cleanup	Функция освобождения ресурсов, занятых в рамках указанной сессии работы с библиотекой.	Является потокобезопасной.
curl_easy_setopt	Функция меняет параметры работы TLS-сессии.	Запрещено использовать с параметрами CURLOPT_VERBOSE, CURLOPT_FTP_SSL_CCC, CURLOPT_OPENSOCKETFUNCTION, CURLOPT_KEYPASSWD, CURLOPT_PROXY_KEYPASSWD, CURLOPT_SSL_VERIFYPEER, CURLOPT_PROXY_SSL_VERIFYPEER, CURLOPT_SSL_VERIFYHOST, CURLOPT_PROXY_SSL_VERIFYHOST, CURLOPT_SSL_OPTIONS, CURLOPT_PROXY_SSL_OPTIONS, CURLOPT_PROXY, CURLOPT_PROXYTYPE, CURLOPT_HTTPPROXYTUNNEL, CURLOPT_PRE_PROXY, CURLOPT_PROXY_TRANSFER_MODE.
curl_easy_getinfo	Функция получения сведений о соединении.	

curl_easy_perform	Функция синхронизируемой передачи данных по заданному дескриптору.	Используемый дескриптор CURL должен быть получен вызовом curl_easy_init. Перед вызовом данной функции с помощью curl_easy_setopt и параметров CURLOPT_STRICT_GOST, CURLOPT_NOPROXY, CURLOPT_REDIR_PROTOCOLS, CURLOPT_FTPSSLAUTH необходимо: • задать проведение проверки алгоритма ключа проверки ЭП для всех сертификатов в цепочке сертификата сервера; • запретить использовать прокси-сервер при соединении с указанным TLS-сервером; • ограничить возможность redirect-перехода с TLS-соединения на незащищенное; • задать использование протокола TLS для защиты канала FTPS-соединения. При установке TLS-соединения с двусторонней аутентификацией перед вызовом данной функции с помощью curl_easy_setopt и параметра CURLOPT_PROXY_SSLCERT и/или CURLOPT_SSLCERT необходимо указать адрес сертификата TLS-клиента в системном хранилище сертификатов.
curl_easy_send	Функция отправляет данные типа raw-data по установленному ранее каналу.	Используемый дескриптор CURL должен быть получен вызовом curl_easy_init. Для вызова функции необходимо предварительно вызвать curl_easy_setopt (с указанием флага CURLOPT_CONNECT_ONLY) и curl_easy_perform.
curl_easy_recv	Функция принимает данные типа raw-data по установленному ранее каналу.	Используемый дескриптор CURL должен быть получен вызовом curl_easy_init. Для вызова функции необходимо предварительно вызвать curl_easy_setopt (с указанием флага CURLOPT_CONNECT_ONLY) и curl_easy_perform.

Вспомогательные функции обработки строк и данных		
curl_easy_escape	Функция перевода заданной строки в URL-кодировку.	
curl_easy_unescape	Функция перевода заданной URL-кодированной строки в строку символов char.	
curl_formadd	Функция добавления новой секции в состав данных типа multipart/formdata, отправляемых по HTTP POST-запросу.	
curl_formfree	Функция освобождения ресурсов, выделенных для данных типа multipart/formdata, составленных ранее набором вызовов curl_formadd.	
curl_slist_append	Функция добавляет строку в список строк структуры curl_slist.	
curl_slist_free_all	Функция освобождает все ресурсы, занятые структурой curl_slist.	

Интерфейс JCA/JCE (КриптоПро JavaCSP и Android)

Метод	Описание	Ограничения на использование метода
`getInstance()` класса `KeyPairGenerator`	Метод создает объект генерации ключевой пары ЭП и VKO (генератор).	
`getInstance()` класса `KeyFactory`	Метод 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 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 осуществляет генерацию ключевой пары.	

`generatePublic()` класса `KeyFactory`	Метод создает объект открытого ключа на основе спецификации открытого ключа (объекта класса PublicKeySpec), содержащей ключевой материал, или кодированного 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 `CertificateFactory`	Метод производит генерацию 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 `GostCertificateRequest`	Метод осуществляет печать 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 `CertPathParameters`	Метод, получающий имя 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`	Метод устанавливает флаг PolicyQualifiersRejected.	
`setRevocationEnabled()` класса `CertPathParameters`	Метод, устанавливающий 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.	
`getEarliestValidSignatureTime-StampToken()` 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.

Функции установки TLS-соединения		
`initClientSSL` объекта `ru.CryptoPro.ssl.android.util.TLSContext`	Функция создания и инициализации объекта SSLContext для клиентской стороны TLS-соединения с односторонней аутентификацией.	Разрешена при указании в качестве параметра tlsProvider значения «JTLS». Должна быть вызвана в случае TLS с односторонней аутентификацией. Перед использованием системная переменная «tls_prohibit_disabled_validation» должна быть установлена в значение «True».
`initAuthClientSSL` объекта `ru.CryptoPro.ssl.android.util.TLSContext`	Функция создания и инициализации объекта SSLContext для клиентской стороны TLS-соединения с двусторонней аутентификацией.	Разрешена при выполнении следующих условий: • должна быть исключена параллельная установка TLS-соединений с двусторонней аутентификацией другими TLS-клиентами и параллельная установка TLS-соединений с односторонней аутентификацией текущим TLS-клиентом; • в качестве tlsProvider и keyStoreProvider должны быть указаны значения «JTLS» и «JCSP» соответственно. Должна быть вызвана в случае TLS с двусторонней аутентификацией. Перед использованием системная переменная «tls_prohibit_disabled_validation» должна быть установлена в значение «True».
`getSocketFactory` объекта `SSLContext`	Функция создания объекта SSLSocketFactory на основе текущего SSLContext.	Разрешена только для объекта SSLContext, полученного разрешённым вызовом initClientSSL или initAuthClientSSL объекта ru.CryptoPro.ssl.android.util.TLSContext.
`createSocket` объекта `SSLSocketFactory`	Функция создания программного сокета с указанием физического адреса сервера.	Разрешена только для объекта SSLSocketFactory, полученного комбинацией разрешённых вызовов TLSContext.initClientSSL или TLSContext.initAuthClientSSL и SSLContext.getSocketFactory. Если createSocket был вызван без параметров, то для полученного объекта SSLSocket нужно вызвать его метод connect.

`connect объекта SSLSocket`	Функция для установки связи сокета с сервером по указанному адресу.	Разрешена только для объекта SSLSocket, полученного комбинацией разрешённых вызовов TLSContext.initClientSSL или TLSContext.initAuthClientSSL, SSLContext.getSocketFactory и SSLSocketFactory.createSocket.
`startHandshake объекта SSLSocket`	Функция для установки TLS-соединения или проведения RENEGOTIATION.	Разрешена только для объекта SSLSocket, полученного комбинацией разрешённых вызовов TLSContext.initClientSSL или TLSContext.initAuthClientSSL, SSLContext.getSocketFactory и SSLSocketFactory.createSocket.
`getOutputStream объекта SSLSocket`	Функция для получения доступа к буферу OutputStream для отправки данных TLS-серверу.	Разрешена только для объекта SSLSocket, полученного комбинацией разрешённых вызовов TLSContext.initClientSSL или TLSContext.initAuthClientSSL, SSLContext.getSocketFactory и SSLSocketFactory.createSocket.
`getInputStream объекта SSLSocket`	Функция для получения доступа к буферу InputStream для получения данных от TLS-сервера.	Разрешена только для объекта SSLSocket, полученного комбинацией разрешённых вызовов TLSContext.initClientSSL или TLSContext.initAuthClientSSL, SSLContext.getSocketFactory и SSLSocketFactory.createSocket.
`write объекта OutputStream`	Функция для отправки данных TLS-серверу.	Разрешена только для объекта OutputStream, полученного комбинацией разрешённых вызовов TLSContext.initClientSSL или TLSContext.initAuthClientSSL, SSLContext.getSocketFactory, SSLSocketFactory.createSocket и SSLSocket.getOutputStream. Значение провайдера по умолчанию (параметра «ru.CryptoPro. defaultSSLProv») должно быть равно «JCSP».

`read объекта InputStream`	Функция для получения данных от TLS-сервера.	Разрешена только для объекта InputStream, полученного комбинацией разрешённых вызовов TLSContext.initClientSSL или TLSContext.initAuthClientSSL, SSLContext.getSocketFactory, SSLSocketFactory.createSocket и SSLSocket.getInputStream. Значение провайдера по умолчанию (параметра «ru.CryptoPro. defaultSSLProv») должно быть равно «JCSP».
`close объекта SSLSocket`	Функция для закрытия TLS-соединения.	Разрешена только для объекта SSLSocket, полученного комбинацией разрешённых вызовов TLSContext.initClientSSL или TLSContext.initAuthClientSSL, SSLContext.getSocketFactory, SSLSocketFactory.createSocket.

Лист регистрации изменений

Лист регистрации изменений									
№ п/п	Номера листов (страниц)				Всего листов (страниц) в документе	№ документа	Входящий № сопроводительного документа и дата	Подпись	Дата
	измененных	замененных	новых	аннулированных					