Internet Engineering Task Force (IETF)
Request for Comments: 6030
Category: Standards Track
ISSN: 2070-1721

P. Hoyer
ActivIdentity
M. Pei
VeriSign
S. Machani
Diversinet

October 2010

Переносимый контейнер с симметричным ключом (PSKC)

Portable Symmetric Key Container (PSKC)

Тезисы

Этот документ задает формат симметричного ключа для транспортировки и обеспечения симметричными ключами криптомодулей различных типов. Примерами могут служить разделяемые секреты с одноразовым паролем (OTP[1]) или симметричные ключи для устройств со строгой аутентификацией. Стандартный формат транспортировки ключей обеспечивает предприятиям возможность внедрения лучших в своем классе решений от разных производителей в рамках одной инфраструктуры.

Статус документа

Этот документ содержит проект стандарта Internet (Internet Standards Track).

Документ является результатом работы IETF и выражает согласованное мнение сообщества IETF. Документ был представлен на публичное рассмотрение и одобрен для публикации IESG. Дополнительная информация о стандартах Internet доступна в разделе 2 RFC 5741.

Информацию о текущем статусе документа, обнаруженных ошибках и способах обратной связи можно получить, воспользовавшись ссылкой http://www.rfc-editor.org/info/rfc6030.

Авторские права

Авторские права ((c) 2010) принадлежат IETF Trust и лицам, указанным в числе авторов документа. Все права защищены.

Этот документ является субъектом прав и ограничений, перечисленных в BCP 78 и IETF Trust Legal Provisions и относящихся к документам IETF (http://trustee.ietf.org/license-info), на момент публикации данного документа. Прочтите упомянутые документы внимательно, поскольку в них описаны права и ограничения, относящиеся к данному документу. Фрагменты программного кода, включенные в этот документ, распространяются в соответствии с упрощенной лицензией BSD, как указано в параграфе 4.e документа Trust Legal Provisions, без каких-либо гарантий (как указано в Simplified BSD License).

Оглавление

[1]One-Time Password.

1. Введение

По мере расширения использования систем на основе симметричных ключей типа шифрования данных или строгой проверки подлинности и, в частности, систем на основе одноразовых паролей (OTP[1]) и механизмов «запрос-отклик» (CR)[2] все более важным становится взаимодействие систем разных производителей и стандартизация форматов для импорта и экспорта (представления) симметричный ключей. Например, производители серверов аутентификации и поставщики услуг традиционно используют фирменные форматы для импорта и экспорта таких ключей в своих системах, что существенно осложняет использование идентификаторов (token) двух разных производителей.

В этом документе определяется стандартный контейнер для ключа на основе XML, называемый переносимым контейнером симметричных ключей (PSKC[3]), для переноса симметричных ключей и относящихся к ним метаданных. Документ также задает информационные элементы, которые требуются для конкретного применения симметричных ключей типа начального счетчика в алгоритме HOTP[4] [HOTP]. В документе также создан реестр IANA для профилей алгоритмов, в котором могут быть записаны алгоритмы, их метаданные и профили передачи PSKC для централизованных и стандартизованных ссылок на них.

1.1. Ключевые слова

Ключевые слова **необходимо** (MUST), **недопустимо** (MUST NOT), **требуется** (REQUIRED), **нужно** (SHALL), **не нужно** (SHALL NOT), **следует** (SHOULD), **не следует** (SHOULD NOT), **рекомендуется** (RECOMMENDED), **возможно** (MAY), **необязательно** (OPTIONAL) в данном документе должны интерпретироваться в соответствии с документом [RFC2119].

1.2. Поддержка версий

Обеспечение выполняется с использованием синтаксиса явного указания номера версии. В настоящее время поддерживается только версия «1.0».

Схема нумерации версий PSKC имеет вид «<major>.<minor>». Значения старшего и младшего номеров **должны** трактоваться, как раздельные целые числа, и каждое из этих чисел **может** инкрементироваться[5] более, чем на 1. Таким образом PSKC 2.4 будет иметь более низкую (младшую) версию по сравнению с PSKC 2.13, а та, в свою очередь, будет ниже PSKC 12.3. Ведущие нули (например, PSKC 6.01) **должны** игнорироваться получателями, а установка их **недопустима**.

Значение старшего номера версии следует увеличивать только для случаев, когда формат сообщений (например, структура элемента) изменен столь существенно, что старые версии не могут взаимодействовать с новой версией. Значение младшего номера версии показывает наличие новых возможностей и **должно** игнорироваться объектами с меньшим значением младшего номера версии, но использоваться для информационных целей объектами с большим значением младшего номера версии протокола.

[1]One-Time Password.
[2]Challenge/Response.
[3]Portable Symmetric Key Container.
[4] HMAC-Based One-Time Password - одноразовый пароль на основе HMAC.
[5]При вводе новой версии. *Прим. перев.*

1.3. Идентификаторы пространства имен

В этом документе используются идентификаторы ресурсов URI[1] [RFC3986] для обозначения ресурсов, алгоритмов и семантики.

1.3.1. Определенные здесь идентификаторы

Пространство имен XML [XMLNS] URI для версии 1.0 протокола PSKC представляет собой:

urn:ietf:params:xml:ns:keyprov:pskc

Определенные здесь ссылки на элементы схемы PSKC используют префикс pskc (определен, как pskc=«urn:ietf:params:xml:ns:keyprov:pskc»). В реализациях **рекомендуется** использовать данное пространство имен.

1.3.3. Ссылочные идентификаторы

Кроме того, представленный в документе синтаксис PSKC использует идентификаторы алгоритмов, определенные в пространстве имен XML Signature [XMLDSIG]:

xmlns:ds="http://www.w3.org/2000/09/xmldsig#"

Ссылки на идентификаторы алгоритмов в пространстве имен XML Signature используют префикс ds.

PSKC также опирается на идентификаторы алгоритмов и элементы, определенные в пространстве имен XML Encryption [XMLENC]:

xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"

Ссылки на пространство имен XML Encryption используют префикс xenc.

При защите транспортировки ключей с использованием основанного на парольной фразе ключа PSKC опирается также на производные элементы ключей, определенные в пространстве имен XML Encryption версии 1.1 [XMLENC11]:

xmlns:xenc11="http://www.w3.org/2009/xmlenc11#"

Ссылки на пространство имен XML Encryption Version 1.1 используют префикс xenc11.

При защите транспортировки ключей с использованием основанного на парольной фразе ключа PSKC опирается также на идентификаторы алгоритмов и элементы, определенные в пространстве имен PKCS #5 [PKCS5]:

xmlns:pkcs5="http://www.rsasecurity.com/rsalabs/pkcs/schemas/pkcs-5v2-0#"

Ссылки на пространство имен PKCS #5 используют префикс pkcs5.

1.3.3. Ссылочные идентификаторы

Кроме того, представленный в документе синтаксис DSKPP использует идентификаторы алгоритмов, опрделенные в пространстве имен XML Signature [XMLDSIG]:

xmlns:ds="http://www.w3.org/2000/09/xmldsig#"

Ссылки на идентификаторы алгоритмов в пространстве имен XML Signature используют префикс ds.

2. Терминология

Примечание. В последующих параграфах этого документа **обязательные** элементы и атрибуты XML выделены шрифтом. Необязательные элементы и атрибуты явно не выделяются, т. е. атрибуты или элементы, не помеченные как **обязательные**, являются опциональными.

3. Обзор переносимых контейнеров и связей между ними

Переносимый контейнер базируется на определении схемы XML и содержит перечисленные ниже основные элементы.

1. **KeyContainer** - представление контейнера, содержащее множество элементов KeyPackage. Пригодный контейней **должен** включать хотя бы один элемент KeyPackage.
2. **KeyPackage** - представление «упаковки», содержащей не более одного ключа и связанной с его представлением конечной точки (endpoint) или конечной точки текущего использования типа физического или виртуального устройства и конкретного CryptoModule.
3. **DeviceInfo** - представление информации об устройстве и критерии уникальной идентификации устройства.
4. **CryptoModuleInfo** - представление информации о CryptoModule, где ключи содержатся или куда они представляются.
5. **Key** - представление переносимого или представляемого ключа.
6. **Data** - представление списка метаданных, связанных с ключом, где имя элемента является названием метаданных, а связанное с именем значение является зашифрованным (например, для элемента данных <Secret>) или открытым (например, для элемента данных <Counter>).

На рисунке 1 представлен верхний уровень структуры элементов данных PSKC.

В последующих параграфах подробно описаны все объекты, а также элементы и атрибуты связанных с ними схем XML.

[1]Uniform Resource Identifier.

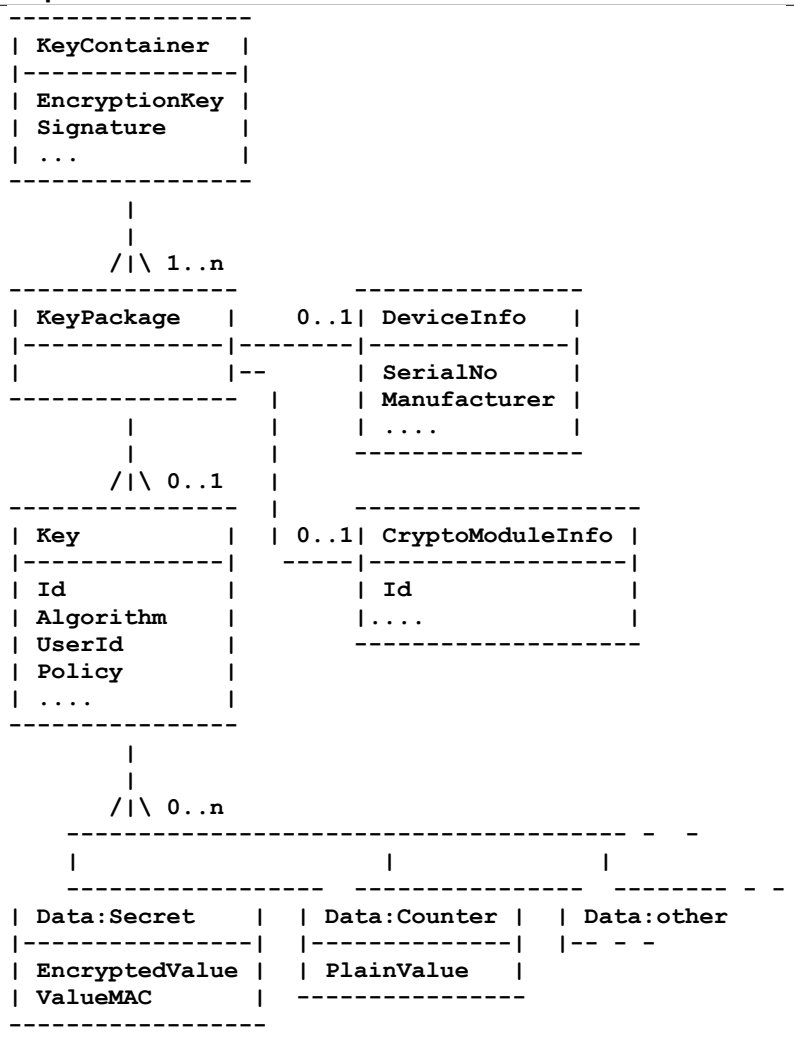

Рисунок 1. Связи между элементами данных PSKC.

4. Элемент <KeyContainer> - основы

В своей базовой форме документ PSKC использует элемент верхнего уровня <KeyContainer> и один элемент <KeyPackage> для передачи информации о ключе.

Ниже приведен пример документа PSKC для описания структуры элемента <KeyContainer> и его дочерних элементов.

```
<?xml version="1.0" encoding="UTF-8"?>
<KeyContainer Version="1.0"
    Id="exampleID1"
    xmlns="urn:ietf:params:xml:ns:keyprov:pskc">
    <KeyPackage>
        <Key Id="12345678"
            Algorithm="urn:ietf:params:xml:ns:keyprov:pskc:hotp">
            <Issuer>Issuer-A</Issuer>
            <Data>
                <Secret>
                    <PlainValue>MTIzNA==
                    </PlainValue>
                </Secret>
            </Data>
        </Key>
    </KeyPackage>
</KeyContainer>
```

Рисунок 2. Пример базового контейнера с ключом PSKC.

Значения атрибутов элемента <KeyContainer> описаны ниже.

Version

Служит для идентификации номера версии схемы PSKC. Данная спецификация определяет начальную версию (1.0) схемы PSKC. Этот атрибут **должен** присутствовать.

Id

Уникальный идентификатор контейнера, помогающий определить конкретный контейнер в случае наличия множества контейнеров в большом документе XML.

4.1. <Key> - вложенный ключевой материал и связанные с ключом данные

Как минимум **должны** включаться описанные ниже атрибуты элемента <Key>.

Id

Этот атрибут передает уникальный идентификатор симметричного ключа в контексте обмена ключами между двумя сторонами. Это означает, что при использовании PSKC во множестве взаимодействий между передающей и приемной стороной, использующих разные контейнеры, ссылающиеся на одни ключи, атрибут Id в <Key> **должен** содержать одно значение (например, после начального предоставления если система хочет обновить значения метаданных ключа в другой системе, значение атрибута Id в <Key>, где метаданные будут обновляться, **должно** совпадать с представленным изначально атрибутом Id). Идентификатор определен как строка алфавитно-цифровых символов.

Algorithm

Этот атрибут содержит уникальный идентификатор профиля алгоритма PSKC. Профиль связывает конкретную семантику с элементами и атрибутами, содержащимися в элементе <Key>. Данный документ описывает профили для алгоритмов открытых стандартов в разделе 10. Дополнительные профили определены в информационном документе [PSKC-ALGORITHM-PROFILES].

Элемент <Key> имеет множество необязательных дочерних элементов. Первоначальный набор описан ниже.

<Issuer>

Этот элемент представляет имя выпустившей сертификат стороны. Например, банк Foobar Bank, Inc., выпускающий маркеры для своих пользователей, может установить для элемента значение.

<FriendlyName>

Читаемое человеком имя для секретного ключа. Этот элемент служит лишь для информирования и может задаваться строками на разных языках, поэтому ему **следует** иметь атрибут xml:lang="xx", где xx - идентификатор языка, заданный в [RFC5646]. Если атрибут xml:lang не задан, реализации **должны** предполагать английский язык, как при указании xml:lang="en".

<AlgorithmParameters>

Этот элемент содержит параметры, которые влияют на результат криптографических расчетов, например, отсечка и формат отклика в алгоритмах OTP и CR. Более подробное описание дано в параграфе 4.3.4.

<Data>

Этот элемент содержит данные о ключе и связанные с ним данные. Ниже перечислены дочерние элементы.

<Secret>

Этот элемент содержит двоичное представление ключа. Форматы кодирования описаны в параграфе 4.2.

<Counter>

Этот элемент содержит счетчик событий для основанных на событиях алгоритмов OTP.

<Time>

Этот элемент указывает время для основанных на времени алгоритмов OTP (при использовании временных интервалов элемент содержит число интервалов от заданной точки старта, обычно зависящее от алгоритма).

<TimeInterval>

Этот элемент указывает интервал времени (в секундах) для основанных на времени алгоритмов OTP (типовое значение 30).

<TimeDrift>

Этот элемент указывает сдвиг (drift) часов устройства для основанных на времени алгоритмов OTP. Целочисленное значение (положительный или отрицательный сдвиг) указывает число временных интервалов, которое сервер проверки установил для сдвига часов устройства после последней успешной аутентификации. Например, если при последней успешной аутентификации было установлено время устройства в 8 интервалов от конкретной стартовой даты, а сервер проверки определил значение в 9, серверу **следует** записать сдвиг -1.

Все элементы, приведенные выше (и те, которые будут определены впредь), следуют простой структуре в том смысле, что они **должны** поддерживать дочерние элементы для передачи значений данных в открытом и шифрованном виде.

Plaintext

Элемент <PlainValue> содержит открытое значение, которое типизовано, например, xs:integer.

Encrypted

Элемент <EncryptedValue> содержит зашифрованное значение.

ValueMAC

Элемент <ValueMAC> заполняется кодом аутентификации сообщения (MAC[1]), создаваемым из шифрованного значения в том случае, если алгоритм шифрования не поддерживает проверки целостности. Пример на рисунке 2 показывает использование элемента <Data> с двумя дочерними элементами <Secret> и <Counter>, которые содержат открытые значения в дочерних элементах <PlainValue>.

4.2. Представление ключа

Две стороны, получившие одно значение ключа (OCTET STRING), что приводит к декодированию xs:base64Binary внутри элементов <PlainValue> или <EncryptedValue>, должны использовать ключ одним способом для обеспечения возможности взаимодействия. Для проверки этого требуется определить соответствие между OCTET STRING и нотацией в описании стандартного алгоритма, определяющего применяемый ключ. В последующих параграфах описано соответствие между алгоритмом AES [FIPS197] и TDEA[2] (или Triple DES) [SP800-67]. Если для алгоритма не указано иное, кодирование OCTET STRING **должно** следовать AES Key Value Encoding.

4.2.1. Представление ключа AES

В параграфе 5.2 [FIPS197] (Key Expansion) в качест входного ключа применяется массив байтов, индексируемых с 0. Первому октету OCTET STRING **нужно** стать байтом ключа с AES, помеченным индексом 0 в [FIPS197], последующим октетам OCTET STRING **нужно** стать байтами в AES в порядке роста индекса.

Корректный разбор и загрузку ключа из содержимого OCTET STRING для AES **нужно** определять с использованием показанного ниже значения элемента <PlainValue> (binaryBase64-encoded) для создания и сопоставления тестовых векторов [FIPS197] (приложение A) для AES.

```
Cipher Key: 2b 7e 15 16 28 ae d2 a6 ab f7 15 88 09 cf 4f 3c
```

[1]Message Authentication Code.
[2]Triple Data Encryption Algorithm - алгоритм тройного шифрования данных.

```
...
 <PlainValue>K34VFiiu0qar9xWICc9PPA==</PlainValue>
...
```

4.2.2. Представление ключа Triple-DES

Ключ Triple-DES состоит из трех ключей для криптографической машины (Key1, Key2, Key3), имеющих размер по 64 бита (56 битов ключа и 8 битов четности). Эти три ключа вместе называют пакетом ключей (key bundle) [SP800-67]. Пакет может реализовать два или три независимых ключа. При использовании лишь двух ключей (Triple DES с двумя ключами) Key1 и Key3 имеют одинаковое значение.

Каждый ключ пакета Triple-DES извлекается в расписание ключей по процедуре, определенной в приложении A к [SP800-67]. Эта процедура нумерует биты ключей от 1 до 64,присваивая номер 1 левому (старшему) биту (MSB). В первый октет OCTET STRING **нужно** помещать биты 1 - 8 ключа Key1 (бит 1 является старшим - MSB). Во второй октет OCTET STRING **нужно** помещать биты 9 - 16 из Key1 и т. д. До финального октета OCTET STRING, в который **нужно** помещать биты 57 - 64 из Key3 (или Key2 для Triple DES с двумя ключами).

Подходящий анализ (разбор) и загрузку содержимого OCTET STRING для Triple DES **нужно** определять с использованием элемента <PlainValue> (кодирование binaryBase64) для генерации и сопоставления тестовых векторов извлечения ключа из приложения B к [SP800-67].

```
Key1 = 0123456789ABCDEF
Key2 = 23456789ABCDEF01
Key3 = 456789ABCDEF0123

...
 <PlainValue>ASNFZ4mrze8jRWeJq83vAUVniavN7wEj</PlainValue>
...
```

4.3. Передача дополнительной информации

Документ PSKC может содержать дополнительную информацию, относящуюся к идентификации устройства, клиптографического модуля и пользователя, а также параметры для использования с алгоритмами OTP и CR. Пример на рисунке 3 служит иллюстрацией для последующих параграфов.

```
<?xml version="1.0" encoding="UTF-8"?>
<KeyContainer Version="1.0"
    Id="exampleID1"
    xmlns="urn:ietf:params:xml:ns:keyprov:pskc">
    <KeyPackage>
        <DeviceInfo>
            <Manufacturer>Manufacturer</Manufacturer>
            <SerialNo>987654321</SerialNo>
            <UserId>DC=example-bank,DC=net</UserId>
        </DeviceInfo>
        <CryptoModuleInfo>
            <Id>CM_ID_001</Id>
        </CryptoModuleInfo>
        <Key Id="12345678"
            Algorithm="urn:ietf:params:xml:ns:keyprov:pskc:hotp">
            <Issuer>Issuer</Issuer>
            <AlgorithmParameters>
                <ResponseFormat Length="8" Encoding="DECIMAL"/>
            </AlgorithmParameters>
            <Data>
                <Secret>
                    <PlainValue>MTIzNDU2Nzg5MDEyMzQ1Njc4OTA=
                    </PlainValue>
                </Secret>
                <Counter>
                    <PlainValue>0</PlainValue>
                </Counter>
            </Data>
            <UserId>UID=jsmith,DC=example-bank,DC=net</UserId>
        </Key>
    </KeyPackage>
</KeyContainer>
```

Рисунок 3. Пример контейнера PSKC Key с дополнительными данными.

4.3.1. Элемент <DeviceInfo> - уникальная идентификация устройства

Элемент <DeviceInfo> однозначно указывает устройство, для которого предоставляется <KeyPackage>. Поскольку устройства могут иметь разный форм-фактор (например, аппаратный маркер, смарт-карта, программный маркер в мобильном телефоне или ПК), этот элемент позволяет применять различные комбинации дочерних элементов. При использовании комбинации она **должна** обеспечивать однозначное указание устройства. Например, для аппаратных маркеров комбинация элементов <SerialNo> и <Manufacturer> указывает устройство однозначно, а отдельный элемент <SerialNo> не обеспечивает, поскольку серийные номера могут совпадать у разных производителей (аналогично для Issuer Distinguished Name и серийного номера сертификата).

Дочерние элементы <DeviceInfo> описаны ниже.

<Manufacturer>

Этот элемент указывает производителя устройства. Значения для <Manufacturer> **должны** браться из префиксов [OATHMAN] (левая колонка) или реестра IANA Private Enterprise Number Registry [IANAPENREG] (Organization). При использовании [OATHMAN] **должен** указываться префикс "oath." (например, "oath.<значение из [OATHMAN]>"), а при использовании [IANAPENREG] **должен** указываться префикс "iana." (например "iana.<Organization из [IANAPENREG]>").

<SerialNo>

Этот элемент указывает серийный номер устройства.

<Model>

Этот элемент описывает модель устройства (например, one-button-HOTP-token-V1).

<IssueNo>

Этот элемент указывает номер выпуска и служит для того, чтобы различать устройства с одинаковым номером.

<DeviceBinding>

Этот элемент позволяет предоставляющему серверу гарантировать, что ключ будет загружен в устройство, для которого был одобрен запрос на предоставления. Устройство связывается с запросом по идентификатору (например, IMEI[1] для мобильного телефона) или классу идентификаторов (например, устройства, для которых ключи защищены TPM[2]).

<StartDate> и <ExpiryDate>

Эти два элемента указывают срок действия устройства (например, для платежной карты, когда номер выпуска на карте не напечатан). Даты **должны** указываться в формате dateTime с «каноническим представлением» [W3C.REC-xmlschema-2-20041028]. Реализациям **не следует** полагаться на точность указания времени лучше миллисекунды и **недопустимо** создавать значения с «високосными» секундами. Ключи, находящиеся на устройстве, **следует** применять только в интервале времени между <StartDate> и <ExpiryDate>. Отметим, что учет дат для ключей **возможен** лишь на сервере проверки, поскольку некоторые устройства (например, смарт-карты) не имеют встроенных часов. Поэтому системам **не следует** полагаться на устройство для контроля срока действия.

В зависимости от типа устройства некоторые дочерние элементы <DeviceInfo> **должны** включаться для отождествления устройства. Этот документ не рассматривает различные типы устройств и поэтому не задает обязательные элементы.

4.3.2. Элемент <CryptoModuleInfo> - идентификация криптомодуля

Элемент <CryptoModuleInfo> указывает криптографический модуль, для которого предоставлены или будут предоставлены симметричеые ключи. Это позволяет указать конкретные варианты в тех случаях, когда устройство **может** содержать более одного криптомодуля (например, ПК с TPM и подключенным маркером).

Элемент <CryptoModuleInfo> имеет один дочерний элемент, который **должен** включаться всегда.

<Id>

Этот элемент содержит уникальный идентификатор для CryptoModule и зависит от реализации, поэтому помогает лишь для идентификации конкретного CryptoModule которому ключ предоставлен или будет предоставлен.

4.3.3. Элемент <UserId> - идентификация пользователя

Элемент <UserId> указывает пользователя отличительного имени, как определено в [RFC4514] (например, UID=jsmith,DC=example,DC=net).

Хотя синтаксис идентификатора определен, с этим элементом не связано конкретной семантики, т. е. нет проверки применения этого ключа лишь определенным пользователем. Поэтому элемент служит лишь для информации.

Этот элемент может указываться в двух местах, а именно в качестве дочернего элемента <Key>, где он указывает связанного с ключом пользователя, или как дочерний элемент <DeviceInfo> для указания пользователя, с которым связано устройство.

4.3.4. Элемент <AlgorithmParameters> - дополнительные данные для OTP и CR

Элемент <AlgorithmParameters> element is a child element of the <Key> element, and this document defines three child elements: <Suite>, <ChallengeFormat>, and <ResponseFormat>.

<Suite>

Необязательный элемент <Suite> определяет дополнительные характеристики используемого алгоритма, зависимые от него. Например, для алгоритма на основе OTP HMAC (Hashed MAC) этот элемент может указывать «силу» применяемого алгоритма хэширования (SHA1, SHA256 и т. п.). Точная семантика значения для каждого профила алгоритма рассматривается в разделе 10.

<ChallengeFormat>

Элемент <ChallengeFormat> определяет харакетристики запроса (challenge) в сценарии использования CR, с помощью которых определяются перечисленные ниже атрибуты.

'Encoding'

Этот атрибут **должен** быть включен и указывает кодирование вызова, воспринимаемое устройством. Атрибут **должен** иметь одно из приведенных ниже значений.

DECIMAL - только цифры.
HEXADECIMAL - шестнадцатеричный отклик.
ALPHANUMERIC - все буквы и цифры с учетом регистра.
BASE64 - кодирование Base-64 в соответствии с разделом 4 [RFC4648].
BINARY - двоичные данные.

'CheckDigits'

Этот атрибут показывает, нужно ли устройству проверять добавленную в конце запроса контрольную цифру Luhn, как указано в [ISOIEC7812]. Атрибут применим только при кодировании DECIMAL. Значение TRUE указывает, что устройство должно проверить контрольную цифру Luhn в конце запроса, FALSE говорит, что устройство не будет проверять контрольную цифру Luhn.

[1]International Mobile Equipment Identity - международное отождествление мобильного оборудования.
[2]Trusted Platform Module - модуль доверенной платформы.

'Min'

Этот атрибут определяет минимальный размер запроса (challenge), воспринимаемого устройством и режиме CR, и **должен** быть включен. При кодировании DECIMAL, HEXADECIMAL или ALPHANUMERIC это значение указывает минимальное число символов, при кодировании BASE64 или 'BINARY - минимальное число октетов кодированного значения.

'Max'

Этот атрибут определяет максимальный размер запроса (challenge), воспринимаемого устройством и режиме CR, и **должен** быть включен. При кодировании DECIMAL, HEXADECIMAL или ALPHANUMERIC это значение указывает минимальное число символов, при кодировании BASE64 или 'BINARY - максимальное число октетов кодированного значения.

<ResponseFormat>

Элемент <ResponseFormat> определяет характеристики результата расчетов и формат OTP или отклика на запрос. Если ключ является значением PIN, этот элемент содержит формат самого PIN (например, DECIMAL размером 4 для PIN из 4 цифр). Определенные для элемента атрибуты перечислены ниже.

'Encoding'

Этот атрибут **должен** быть включен и указывает кодирование отклика, создаваемого устройством. Атрибут **должен** присутствовать и иметь значение DECIMAL, HEXADECIMAL, ALPHANUMERIC, BASE64 или BINARY.

'CheckDigits'

Этот атрибут показывает, нужно ли устройству добавлять в конце отклика контрольную цифру Luhn, как указано в [ISOIEC7812]. Атрибут примени только при кодировании DECIMAL. Значение TRUE указывает, что устройство должно добавить контрольную цифру Luhn в конце отклика, FALSE говорит, что устройство не будет добвлять контрольную цифру Luhn.

'Length'

Этот атрибут определяет размер отклика, создаваемого устройством, и **должен** присутствовать. При кодировании DECIMAL, HEXADECIMAL или ALPHANUMERIC это значение указывает число символов, при кодировании BASE64 или 'BINARY - число октетов кодированного значения.

4.4. Передача значений для вывода ключей

Элемент <KeyProfileId>, являющийся дочерним для элемента <Key>, передает уникальный идентификатор, применяемый между передающей и приемной стороной для организации набора значений атрибутов ключа, которые не передаются внутри контейнера, но могут согласовываться между сторонами по отдельному каналу (out of band). Этот элемент будет представлять уникальную ссылку на набор значений атрибутов ключей (например, идентификатор профиля персонализации смарт-карты, связанный с конкретными значениями атрибутов, присутствующими в приложении смарт-карты и влияющими на рсчет отклика).

Например, в случае MasterCard CAP[1] [CAP] передающая и приемная стороны согласуют, что KeyProfileId='1' представляет некий набор значений (например, флаг Internet Authentication Flag - IAF имеет определенное значение). В процессе передачи <KeyContainer> эти значения не будут передаваться как атрибуты ключа и будет включена лишь ссылка через элемент <KeyProfileId> на конкретный профиль, согласованный ранее (в данном случае '1'). Принимающая сторона затем свяжет все относящиеся к делу атрибуты из профиля, которые были согласованы по отдельному каналу, с импортированными ключами. Такой подход часто применяется между проиводством, управляемым компанией A, и службой проверки, управляемой компанией B, для предотвращения повторяющейся передачи одного и того же набора значений атрибутов ключа.

```
<?xml version="1.0" encoding="UTF-8"?>
<KeyContainer Version="1.0" Id="exampleID1"
     xmlns="urn:ietf:params:xml:ns:keyprov:pskc">
    <KeyPackage>
        <DeviceInfo>
            <Manufacturer>Manufacturer</Manufacturer>
            <SerialNo>987654321</SerialNo>
        </DeviceInfo>
        <CryptoModuleInfo>
            <Id>CM_ID_001</Id>
        </CryptoModuleInfo>
        <Key Id="12345678"
         Algorithm="urn:ietf:params:xml:ns:keyprov:pskc:hotp">
            <Issuer>Issuer</Issuer>
            <AlgorithmParameters>
                <ResponseFormat Length="8" Encoding="DECIMAL"/>
            </AlgorithmParameters>
            <KeyProfileId>keyProfile1</KeyProfileId>
            <KeyReference>MasterKeyLabel
            </KeyReference>
            <Data>
                <Counter>
                    <PlainValue>0</PlainValue>
                </Counter>
            </Data>
            <Policy>
                <KeyUsage>OTP</KeyUsage>
            </Policy>
        </Key>
    </KeyPackage>
</KeyContainer>
```

Рисунок 4. Пример документа PSKC, передающего ключ HOTP в Key Derivation.

[1]Chip Authentication Program - программа аутентификации микросхемы.

Элемент <KeyReference> содержит ссылку на внешний ключ для использования схемой вывода ключа. В этом случае родительский элемент <Key> не будет включать субэлемент <Secret> в <Data>, где передается значение (секретного) ключа, а указывается лишь ссылка на внешний первичный ключ (например, метка ключа PKCS #11).

Значение ключа будет выводиться с использованием элемента <SerialNo>, значений, согласованных между передающей и приемной стороной, указанных <KeyProfile> keyProfile1, и согласованного извне ключа, указанного меткой MasterKeyLabel.

5. Политика ключа

Этот раздел иллюстрирует функциональность элемента <Policy> в PSKC, позволяющего присоединить правила использования ключа и защиты PIN к определенному ключу и связанным с ним метаданным. Этот элемент является дочерним для элемента <Key> .

Если <Policy> содержит дочерние элементы или значение в элементах/атрибутах, которые непонятны получателю документа PSKC, получатель **должен** считать, что использование этого ключа не разрешено. Это гарантирует , что непонимание тех или иных расширений не приведет к непредусмотренному использованию ключа.

Начнем рассмотрение с примера, расширяющего пример на рисунке 3.

```
<?xml version="1.0" encoding="UTF-8"?>
<KeyContainer
    Version="1.0" Id="exampleID1"
    xmlns="urn:ietf:params:xml:ns:keyprov:pskc">
    <KeyPackage>
        <DeviceInfo>
            <Manufacturer>Manufacturer</Manufacturer>
            <SerialNo>987654321</SerialNo>
        </DeviceInfo>
        <CryptoModuleInfo>
            <Id>CM_ID_001</Id>
        </CryptoModuleInfo>
        <Key Id="12345678"
            Algorithm="urn:ietf:params:xml:ns:keyprov:pskc:hotp">
            <Issuer>Issuer</Issuer>
            <AlgorithmParameters>
                <ResponseFormat Length="8" Encoding="DECIMAL"/>
            </AlgorithmParameters>
            <Data>
                <Secret>
                    <PlainValue>MTIzNDU2Nzg5MDEyMzQ1Njc4OTA=</PlainValue>
                </Secret>
                <Counter>
                    <PlainValue>0</PlainValue>
                </Counter>
            </Data>
            <Policy>
                <PINPolicy MinLength="4" MaxLength="4"
                    PINKeyId="123456781" PINEncoding="DECIMAL" PINUsageMode="Local"/>
                <KeyUsage>OTP</KeyUsage>
            </Policy>
        </Key>
    </KeyPackage>
    <KeyPackage>
        <DeviceInfo>
            <Manufacturer>Manufacturer</Manufacturer>
            <SerialNo>987654321</SerialNo>
        </DeviceInfo>
        <CryptoModuleInfo>
            <Id>CM_ID_001</Id>
        </CryptoModuleInfo>
        <Key Id="123456781"
            Algorithm="urn:ietf:params:xml:ns:keyprov:pskc:pin">
            <Issuer>Issuer</Issuer>
            <AlgorithmParameters>
                <ResponseFormat Length="4" Encoding="DECIMAL"/>
            </AlgorithmParameters>
            <Data>
                <Secret>
                    <PlainValue>MTIzNA==</PlainValue>
                </Secret>
            </Data>
        </Key>
    </KeyPackage>
</KeyContainer>
```

Рисунок 5. Нешифрованный HOTP Secret Key, защищенный PIN.

Этот документ определяет два дочерних элемента для <Policy>, описанных ниже.

<StartDate> и <ExpiryDate>

Эти два элемента указывают срок действия ключа. Это **должно** гарантировать, что ключ будет применяться лишь в интервале между этими датами (включительно). Даты **должны** указываться в формате dateTime с «каноническим представлением» [W3C.REC-xmlschema-2-20041028]. Реализациям **не следует** полагаться на точность указания времени лучше миллисекунды и **недопустимо** создавать значения с «високосными» секундами. При отсутствии этого элемента начальным моментом считается текущее время.

<NumberOfTransactions>

Значение этого элемента задает максимальное число случаев использования ключа приложением после получения документа PSKC. При отсутствии элемента ограничений на число применений ключа на задается.

<KeyUsage>

Элемент <KeyUsage> задает ограничения на использование ключа, которым **должен** следовать получатель документа PSKC. Этот документ регистрирует приведенные ниже маркеры.

OTP

Ключ **должен** применяться только для создания OTP.

CR

Ключ **должен** применяться только для Challenge/Response.

Encrypt

Ключ **должен** применяться только для шифрования.

Integrity

Ключ **должен** применяться только для генерации дайджестов сообщений с целью защиты целостности или аутентификации.

Verify

Ключ **должен** применяться только для проверки дайджестов, служащих целям защиты целостности или аутентификации (т. е. обратно по отношению к Integrity).

Unlock

Ключ **должен** применяться только для инверсных Challenge/Response в случаях, когда пользователь заблокировал устройство в результате многократного ввода некорректных значений PIN (для устройств с поддержкой ввода PIN).

Decrypt

Ключ **должен** применяться только для расшифровки.

KeyWrap

Ключ **должен** применяться только для «заворачивания» ключей.

Unwrap

Ключ **должен** применяться только для «разворачивания» ключей.

Derive

Ключ **должен** применяться только с функцией вывода ключей для создания нового ключа (см. также параграф 8.2.4 в [NIST800-57]).

Generate

Ключ **должен** применяться только для создания нового ключа на основе случайного значения и прежнего ключа (см. также параграф 8.1.5.2.1 в [NIST800-57]).

Элемент **может** повторяться для указания нескольких способов применения ключа. При отсутствии элемента предполагается, что ограничения не заданы и ключ **можно** применять для любых целей.

<PINPolicy>

Элемент <PINPolicy> позволяет связать с ключом политику использования PIN. Атрибуты элемента указаны ниже.

'PINKeyId'

Этот атрибут передает уникальный идентификатор элемента <Key>, содержащегося в этом <KeyContainer>, который указывает значение PIN для защиты ключа.

'PINUsageMode'

Этот обязательный атрибут упазывает способ применения PIN при использовании ключа.

Local

PIN проверяется локально на устройстве до разрешения использовать ключ в работе алгоритма.

Prepend

PIN помещается перед откликом алгоритма (prepend), поэтому значение PIN **должно** проверяться любой стороной, проверяющей отклик.

Append

PIN помещается после отклика алгоритма (append), поэтому значение PIN **должно** проверяться любой стороной, проверяющей отклик.

Algorithmic

PIN используется как часть расчетов алгоритма.

'MaxFailedAttempts'

Этот атрибут задает максимальное число неудачных попыток ввести PIN, прежде чем использование ключа станет **недопустимо** (разумными значениями являются положительные целые числа от 2 до 10).

'MinLength'

Этот атрибут задает минимальный размер PIN, который может быть установлен для защиты связанного ключа. **Недопустимо** задавать PIN, размер которого меньше этого значения. Если атрибут PINFormat имеет значение DECIMAL, HEXADECIMAL или ALPHANUMERIC, этот атрибут задает число символов, а для форматов BASE64 и BINARY - число байтов кодированного значения.

'MaxLength'

Этот атрибут задает максимальный размер PIN, который может быть установлен для защиты связанного ключа. **Недопустимо** задавать PIN, размер которого больше этого значения. Если атрибут PINFormat имеет значение DECIMAL, HEXADECIMAL или ALPHANUMERIC, этот атрибут задает число символов, а для форматов BASE64 и BINARY - число байтов кодированного значения.

'PINEncoding'

Этот атрибут задает кодирование PIN и **должен** принимать одно из значений DECIMAL, HEXADECIMAL, ALPHANUMERIC, BASE64 или BINARY.

При PinUsageMode = Local ограничения, заданные в MaxFailedAttempts, MinLength, MaxLength и PINEncoding, **должно** применять устройство, в остальных случаях они **должны** применяться на стороне сервера.

5.1. Определение алгоритма PIN

Алгоритм PIN определяется как

```
boolean = comparePIN(K,P)
```

где

K - сохраненное симметричное свидетельство (PIN) в двоичном формате;

P - предложенное для сравения значение PIN в двоичном формате.

Функция comparePIN выполняет непосредственное сравнение октетов K и P. Такое сравнение **должно** давать TRUE (совпадение), когда размеры K и P одинаковы, а каждый октет K совпадает с соответствующим октетом P.

6. Методы защиты ключей

С описанной выше функциональностью информация, связанная с ключами, передается в открытом виде. С помощью элемента <EncryptionKey>, который является дочерним для <KeyContainer>, можно зашифровать ключи и связанную с ними информацию. Уровень шифрования применяется к значениям отдельных элементов, а используемый алгоритм **должен** совпадать для всех шифруемых элементов. Ключи защищаются с использованием заранее известных общих ключей (pre-shared), ключей на основе парольной фразы (passphrase-based) и асимметричных ключей. При использовании алгоритмов шифрования с вектором инициализации (IV[1]), например, AES-128-CBC, **должно** создаваться случайное значение IV для каждого шифруемого элемента и этот вектор **должен** помещаться перед зашифрованнысм значением (prepend), как указано в [XMLENC].

6.1. Шифрование на основе заранее распределенных ключей

Рисунок 6 иллюстрирует шифрование элемента <Secret> с применением AES-128-CBC и заполнения PKCS #5. Открытое значение <Secret> равно 31323334353637383930313233343536373839 30. Заранее известных общий секрет имеет значение Pre-shared-key, как указано в элементе <KeyName> (дочерний для <EncryptionKey>). Используемый ключ шифрования имеет значение 12345678901234567890123456789012.

IV для MAC имеет значение 11223344556677889900112233445566, для HOTP - 000102030405060708090a0b0c0d0e0f.

```
<?xml version="1.0" encoding="UTF-8"?>
<KeyContainer Version="1.0"
    xmlns="urn:ietf:params:xml:ns:keyprov:pskc"
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
    xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">
    <EncryptionKey>
        <ds:KeyName>Pre-shared-key</ds:KeyName>
    </EncryptionKey>
    <MACMethod Algorithm="http://www.w3.org/2000/09/xmldsig#hmac-sha1">
        <MACKey>
            <xenc:EncryptionMethod
            Algorithm="http://www.w3.org/2001/04/xmlenc#aes128-cbc"/>
            <xenc:CipherData>
                <xenc:CipherValue>
    ESIzRFVmd4iZABEiM0RVZgKn6WjLaTC1sbeBMSvIhRejN9vJa2BOlSaMrR7I5wSX
                </xenc:CipherValue>
            </xenc:CipherData>
        </MACKey>
    </MACMethod>
    <KeyPackage>
        <DeviceInfo>
            <Manufacturer>Manufacturer</Manufacturer>
            <SerialNo>987654321</SerialNo>
        </DeviceInfo>
        <CryptoModuleInfo>
            <Id>CM_ID_001</Id>
        </CryptoModuleInfo>
        <Key Id="12345678"
            Algorithm="urn:ietf:params:xml:ns:keyprov:pskc:hotp">
            <Issuer>Issuer</Issuer>
            <AlgorithmParameters>
                <ResponseFormat Length="8" Encoding="DECIMAL"/>
            </AlgorithmParameters>
            <Data>
                <Secret>
                    <EncryptedValue>
                        <xenc:EncryptionMethod
            Algorithm="http://www.w3.org/2001/04/xmlenc#aes128-cbc"/>
                        <xenc:CipherData>
                            <xenc:CipherValue>
    AAECAwQFBgcICQoLDA0OD+cIHItlB3Wra1DUpxVvOx2lef1VmNPCMl8jwZqIUqGv
                            </xenc:CipherValue>
                        </xenc:CipherData>
                    </EncryptedValue>
                    <ValueMAC>Su+NvtQfmvfJzF6bmQiJqoLRExc=
```

[1]Initialization Vector.

```
                    </ValueMAC>
                </Secret>
                <Counter>
                    <PlainValue>0</PlainValue>
                </Counter>
            </Data>
        </Key>
    </KeyPackage>
</KeyContainer>
```

Рисунок 6. Шифрование с заранее известным ключом и AES-128-CBC с HMAC-SHA1.

Поскольку AES-128-CBC не обеспечивает защиты целостности, для зашифрованного значения применяется MAC с ключом и алгоритмом, указанными в элементе <MACMethod> (http://www.w3.org/2000/09/xmldsig#hmac-sha1 в приведенном примере служит алгоритмом, ключ MAC (11223344556677889900112233445566778899 00) создается случайным образом и шифруется с указанным ключом). Результат расчета MAC помещается в <ValueMAC> - дочерний элемент <Secret>.

При защите данных с заранее известным ключом реализация **должна** указать имя конкретного ключа, распределенного заранее, в элементе <KeyName> внутри <EncryptionKey>. При шифровании в режиме CBC, который требует явного вектора инициализации, значение IV **должно** указываться перед шифрованным значением (prepend).

Для систем, реализующих PSKC, **рекомендуется** поддержка AES-128-CBC (http://www.w3.org/2001/04/xmlenc#aes128-cbc) и KW-AES128 (http://www.w3.org/2001/04/xmlenc#kw-aes128). Следует отметить, что KW-AES128 требует размера защищаемого ключа, кратного 8 байтам. Поэтому при необходимости защиты ключей иного размера **рекомендуется** алгоритм «заворачивания» ключа (key-wrap) с заполнением, как описано в [RFC5649]. Некоторые из алгоритмов шифрования, которые могут быть реализованы, перечислены в таблице.

Алгоритм	*Идентификатор URL*
AES192-CBC	http://www.w3.org/2001/04/xmlenc#aes192-cbc
AES256-CBC	http://www.w3.org/2001/04/xmlenc#aes256-cbc
TripleDES-CBC	http://www.w3.org/2001/04/xmlenc#tripledes-cbc
Camellia128	http://www.w3.org/2001/04/xmldsig-more#camellia128-cbc
Camellia192	http://www.w3.org/2001/04/xmldsig-more#camellia192-cbc
Camellia256	http://www.w3.org/2001/04/xmldsig-more#camellia256-cbc
KW-AES128	http://www.w3.org/2001/04/xmlenc#kw-aes128
KW-AES192	http://www.w3.org/2001/04/xmlenc#kw-aes192
KW-AES256	http://www.w3.org/2001/04/xmlenc#kw-aes256
KW-TripleDES	http://www.w3.org/2001/04/xmlenc#kw-tripledes
KW-Camellia128	http://www.w3.org/2001/04/xmldsig-more#kw-camellia128
KW-Camellia192	http://www.w3.org/2001/04/xmldsig-more#kw-camellia192
KW-Camellia256	http://www.w3.org/2001/04/xmldsig-more#kw-camellia256

6.1.1. Метод MAC

При использовании алгоритмов без контроля целостности, таких как AES-128-CBC, значение MAC с ключом **должно** помещаться в элемент <ValueMAC> внутри <Data>. В этом случае алгоритм MAC **должен** быть указан элементом <MACMethod> в <KeyContainer>. Ключ MAC **должен** создаваться отправителем случайным образом, быть заранее согласованным с получателем и устанавливаться приклатным протоколом, который передает документ PSKC. Отправителю **рекомендуется** генерировать случайный ключ MAC. При случайном создании ключевой материал MAC **должен** шифроваться с ключом, указанным в элементе <EncryptionKey> ключевого контейнера. Метод шифрования и шифрованное значение **должны** помещаться в элемент <EncryptionMethod> и <CipherData> (соответственно) элемента <MACKey> в <MACMethod>. **Можно** применять элемент <MACKeyReference> в <MACMethod> для указания заранее распространенного ключа MAC или протокола вывода ключа MAC. Системам, поддерживающим PSKC, **рекомендуется** реализовать HMAC-SHA1 (http://www.w3.org/2000/09/xmldsig#hmac-sha1). Некоторые из алгоритмов MAC, которые могут быть реализованы, перечислены в таблице.

Алгоритм	*Идентификатор URL*
HMAC-SHA224	http://www.w3.org/2001/04/xmldsig-more#hmac-sha224
HMAC-SHA256	http://www.w3.org/2001/04/xmldsig-more#hmac-sha256
HMAC-SHA384	http://www.w3.org/2001/04/xmldsig-more#hmac-sha384
HMAC-SHA512	http://www.w3.org/2001/04/xmldsig-more#hmac-sha512

6.2. Шифрование с ключами на основе парольной фразы

На рисунке 7 дан пример шифрования содержимого элемента <Secret> с использованием выведенного из парольной фразы ключа (PBKDF2[1]) в соответствии с [PKCS5]. При выводе ключа из парольной фразы **должен** применяться элемент <DerivedKey>, определенный в XML Encryption Version 1.1 [XMLENC11], для указания ключа на основе парольной фразы. <DerivedKey> указывается как дочерний элемент <EncryptionKey> в ключевом контейнере.

Элемент <DerivedKey> служит для задания функции вывода ключа и связанных параметров. Алгоритм шифрования(в примере AES-128-CBC, http://www.w3.org/2001/04/xmlenc#aes128-cbc) **должен** быть указан в атрибуте Algorithm элемента <EncryptionMethod>, используемого в зашифрованных элементах данных.

При использовании PBKDF2 атрибут Algorithm элемента <xenc11: KeyDerivationMethod> **должен** содержать URI http://www.rsasecurity.com/rsalabs/pkcs/schemas/pkcs-5#pbkdf2'. Элемент <xenc11:KeyDerivationMethod> **должен** включать дочерний элемент <PBKDF2-params> с параметрми PBKDF2, такими как «затравка» и число итераций.

При шифровании с режиме CBC с явным вектором инициализации (IV), отличным от выведенного, значение IV **должно** передаваться путем указания перед зашифрованным значением (prepend).

Ниже перечислены значения, используемые в примере.

[1]Passphrase-based key derivation.

Password - qwerty;

Salt - 0x123eff3c4a72129c;

Iteration Count - 1000;

MAC Key - 0xbdaab8d648e850d25a3289364f7d7eaaf53ce581;

OTP Secret - 12345678901234567890.

Выведенный ключ шифрования имеет значение 0x651e63cd57008476af1ff6422cd02e41, вектор инициализации (IV) - 0xa13be8f92db69ec992d99fd1b5ca05f0. Этот ключ также применяется для шифрования случайно выбранного ключа MAC. Можно применять другой IV (в примере 0xd864d39cbc0cdc8e1ee483b9164b9fa0). Шифрование по алгоритму AES-128-CBC соответствует [XMLENC].

```
<?xml version="1.0" encoding="UTF-8"?>
<pskc:KeyContainer
  xmlns:pskc="urn:ietf:params:xml:ns:keyprov:pskc"
  xmlns:xenc11="http://www.w3.org/2009/xmlenc11#"
  xmlns:pkcs5=
  "http://www.rsasecurity.com/rsalabs/pkcs/schemas/pkcs-5v2-0#"
  xmlns:xenc="http://www.w3.org/2001/04/xmlenc#" Version="1.0">
    <pskc:EncryptionKey>
        <xenc11:DerivedKey>
            <xenc11:KeyDerivationMethod
              Algorithm=
 "http://www.rsasecurity.com/rsalabs/pkcs/schemas/pkcs-5v2-0#pbkdf2">
                <pkcs5:PBKDF2-params>
                    <Salt>
                        <Specified>Ej7/PEpyEpw=</Specified>
                    </Salt>
                    <IterationCount>1000</IterationCount>
                    <KeyLength>16</KeyLength>
                    <PRF/>
                </pkcs5:PBKDF2-params>
            </xenc11:KeyDerivationMethod>
            <xenc:ReferenceList>
                <xenc:DataReference URI="#ED"/>
            </xenc:ReferenceList>
            <xenc11:MasterKeyName>My Password 1</xenc11:MasterKeyName>
        </xenc11:DerivedKey>
    </pskc:EncryptionKey>
    <pskc:MACMethod
        Algorithm="http://www.w3.org/2000/09/xmldsig#hmac-sha1">
        <pskc:MACKey>
            <xenc:EncryptionMethod
            Algorithm="http://www.w3.org/2001/04/xmlenc#aes128-cbc"/>
            <xenc:CipherData>
                <xenc:CipherValue>
2GTTnLwM3I4e5IO5FkufoOEiOhNj91fhKRQBtBJYluUDsPOLTfUvoU2dStyOwYZx
                </xenc:CipherValue>
            </xenc:CipherData>
        </pskc:MACKey>
    </pskc:MACMethod>
    <pskc:KeyPackage>
        <pskc:DeviceInfo>
            <pskc:Manufacturer>TokenVendorAcme</pskc:Manufacturer>
            <pskc:SerialNo>987654321</pskc:SerialNo>
        </pskc:DeviceInfo>
        <pskc:CryptoModuleInfo>
            <pskc:Id>CM_ID_001</pskc:Id>
        </pskc:CryptoModuleInfo>
        <pskc:Key Algorithm="urn:ietf:params:xml:ns:keyprov:pskc:hotp" Id="123456">
            <pskc:Issuer>Example-Issuer</pskc:Issuer>
            <pskc:AlgorithmParameters>
                <pskc:ResponseFormat Length="8" Encoding="DECIMAL"/>
            </pskc:AlgorithmParameters>
            <pskc:Data>
                <pskc:Secret>
                <pskc:EncryptedValue Id="ED">
                    <xenc:EncryptionMethod
                        Algorithm="http://www.w3.org/2001/04/xmlenc#aes128-cbc"/>
                        <xenc:CipherData>
                            <xenc:CipherValue>
      oTvo+S22nsmS2Z/RtcoF8Hfh+jzMe0RkiafpoDpnoZTjPYZu6V+A4aEn032yCr4f
                        </xenc:CipherValue>
                    </xenc:CipherData>
                    </pskc:EncryptedValue>
                    <pskc:ValueMAC>LP6xMvjtypbfT9PdkJhBZ+D6O4w=
```

```
                    </pskc:ValueMAC>
                </pskc:Secret>
            </pskc:Data>
        </pskc:Key>
    </pskc:KeyPackage>
</pskc:KeyContainer>
```

Рисунок 7. Пример документа PSKC с шифрованием ключом на основе парольной фразы.

6.3. Шифрование на базе асимметричных ключей

При использовании асимметричных ключей для шифрования дочерних элементов элемента <Data> информация о сертификате **должна** представляться элементом <X509Data>, который является дочерним для <EncryptionKey>. Алгоритм шифрования **должен** указываться атрибутом Algorithm элемента <EncryptionMethod>. В примере на рисунке 8 используется алгоритм http://www.w3.org/2001/04/xmlenc#rsa_1_5.

```
<?xml version="1.0" encoding="UTF-8" ?>
<KeyContainer
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
    xmlns="urn:ietf:params:xml:ns:keyprov:pskc"
    xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
    Id="KC0001"
    Version="1.0">
    <EncryptionKey>
        <ds:X509Data>
<ds:X509Certificate>MIIB5zCCAVCgAwIBAgIESZp/vDANBgkqhkiG9w0BAQUFADA4M
Q0wCwYDVQQKEwRJRVRGMRMwEQYDVQQLEwpLZXlQcm92IFdHMRIwEAYDVQQDEwlQU0tDIF
Rlc3QwHhcNMDkwMjE3MDkxMzMyWhcNMTEwMjE3MDkxMzMyWjA4MQ0wCwYDVQQKEwRJRVR
GMRMwEQYDVQQLEwpLZXlQcm92IFdHMRIwEAYDVQQDEwlQU0tDIFRlc3QwgZ8wDQYJKoZI
hvcNAQEBBQADgY0AMIGJAoGBALCWLDa2ItYJ6su80hd1gL4cggQYdyyKK17btt/aS6Q/e
DsKjsPyFIODsxeKVV/uA3wLT4jQJM5euKJXkDajzGGOy92+ypfzTX4zDJMkh61SZwlHNJ
xBKilAM5aW7C+BQ0RvCxvdYtzx2LTdB+X/KMEBA7uIYxLfXH2Mnub3WIh1AgMBAAEwDQY
JKoZIhvcNAQEFBQADgYEAe875m84sYUJ8qPeZ+NG7REgTvlHTmoCdoByU0LBBLotUKuqf
rnRuXJRMeZXaaEGmzY1kLonVjQGzjAkU4dJ+RPmiDlYuHLZS41Pg6VMwY+03lhk6I5A/w
4rnqdkmwZX/NgXg06alnc2pBsXWhL4O7nk0S2ZrLMsQZ6HcsXgdmHo=
</ds:X509Certificate>
        </ds:X509Data>
    </EncryptionKey>
    <KeyPackage>
        <DeviceInfo>
            <Manufacturer>TokenVendorAcme</Manufacturer>
            <SerialNo>987654321</SerialNo>
        </DeviceInfo>
        <Key
            Id="MBK000000001"
            Algorithm="urn:ietf:params:xml:ns:keyprov:pskc:hotp">
            <Issuer>Example-Issuer</Issuer>
            <AlgorithmParameters>
                <ResponseFormat Length="6" Encoding="DECIMAL"/>
            </AlgorithmParameters>
            <Data>
                <Secret>
                    <EncryptedValue>
                        <xenc:EncryptionMethod
             Algorithm="http://www.w3.org/2001/04/xmlenc#rsa_1_5"/>
                        <xenc:CipherData>
<xenc:CipherValue>hJ+fvpoMPMO9BYpK2rdyQYGIxiATYHTHC7e/sPLKYo5/r1v+4
xTYG3gJolCWuVMydJ7Ta0GaiBPHcWa8ctCVYmHKfSz5fdeV5nqbZApe6dofTqhRwZK6
Yx4ufevi91cjN2vBpSxYafvN3c3+xIgk0EnTV4iVPRCR0rBwyfFrPc4=
</xenc:CipherValue>
                        </xenc:CipherData>
                    </EncryptedValue>
                </Secret>
                <Counter>
                    <PlainValue>0</PlainValue>
                </Counter>
            </Data>
        </Key>
    </KeyPackage>
</KeyContainer>
```

Рисунок 8. Пример документа PSKC с шифрованием на основе открытого ключа.

Системам PSKC **рекомендуется** реализовать алгоритм RSA-1.5 (http://www.w3.org/2001/04/xmlenc#rsa-1_5).

В таблице указан дополнительный алгоритм асимметричного шифрования, который может быть реализован.

Алгоритм	*Идентификатор URL*
RSA-OAEP-MGF1P	http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p

6.4. Дополнение шифрованных значений для алгоритмов без дополнения

Дополнение зашифрованных значений (например, секретного ключа) требуется в тех случаях, когда используются механизмы защиты ключей, не поддерживающие встроенного дополнения, а размер шифруемого ключа не кратен размеру блока шифрования в применяемом алгоритме.

Например, при передаче ключа HOTP (20 байтов), защищенного алгоритмом AES в режиме CBC (8-байтовый блочный шифр), заполнение требуется, поскольку размер ключа не кратен размеру 8-байтового блока.

В таких случаях реализующим PSKC системам **рекомендуется** дополнять значение перед шифрованием с использованием механизма PKCS #5, как описано в [PKCS5].

7. Цифровая подпись

PSKC позволяет добавлять цифровую подпись в документ XML как дочерний элемент в <KeyContainer>. Описание цифровой подписи XML можно найти в [XMLDSIG].

```
<?xml version="1.0" encoding="UTF-8"?>
<KeyContainer
    xmlns="urn:ietf:params:xml:ns:keyprov:pskc"
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
    xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
    Version="1.0">
    <KeyPackage>
        <DeviceInfo>
            <Manufacturer>TokenVendorAcme</Manufacturer>
            <SerialNo>0755225266</SerialNo>
        </DeviceInfo>
        <Key Id="123"
        Algorithm="urn:ietf:params:xml:ns:keyprov:pskc:hotp">
            <Issuer>Example-Issuer</Issuer>
            <AlgorithmParameters>
                <ResponseFormat Length="6" Encoding="DECIMAL"/>
            </AlgorithmParameters>
            <Data>
                <Secret>
                    <PlainValue>
                        MTIzNDU2Nzg5MDEyMzQ1Njc4OTA=
                    </PlainValue>
                </Secret>
                <Counter>
                    <PlainValue>0</PlainValue>
                </Counter>
            </Data>
        </Key>
    </KeyPackage>
    <ds:Signature>
        <ds:SignedInfo>
            <ds:CanonicalizationMethod
             Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
            <ds:SignatureMethod
             Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
            <ds:Reference URI="#Device">
                <ds:DigestMethod
             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                <ds:DigestValue>
                    j6lwx3rvEPO0vKtMup4NbeVu8nk=
                </ds:DigestValue>
            </ds:Reference>
        </ds:SignedInfo>
        <ds:SignatureValue>
            j6lwx3rvEPO0vKtMup4NbeVu8nk=
        </ds:SignatureValue>
        <ds:KeyInfo>
            <ds:X509Data>
                <ds:X509IssuerSerial>
                    <ds:X509IssuerName>
                        CN=Example.com,C=US
                    </ds:X509IssuerName>
                    <ds:X509SerialNumber>
                        12345678
                    </ds:X509SerialNumber>
                </ds:X509IssuerSerial>
            </ds:X509Data>
        </ds:KeyInfo>
    </Signature>
</KeyContainer>
```

Рисунок 9. Пример цифровой подписи.

8. Массовое представление

Функциональность массового представления может быть реализована путем повтора элемента <KeyPackage> внутри <KeyContainer>, указывающего представление множества ключей для разных устройств или криптографических модулей. Элемент <EncryptionKey> в таком случае применяет все элементы <KeyPackage>. При представлении множества ключей одному устройству элемент <KeyPackage> повторяется, а вложенный в него элемент <DeviceInfo> всегда содержит одни субэлементы, однозначно указывающие устройство (например, ниже представлены ключи для устройства с SerialNo=9999999).

На рисунке 10 приведен пример использования описанной возможности.

```
<?xml version="1.0" encoding="UTF-8"?>
<KeyContainer Version="1.0"
    xmlns="urn:ietf:params:xml:ns:keyprov:pskc">
    <KeyPackage>
        <DeviceInfo>
            <Manufacturer>TokenVendorAcme</Manufacturer>
            <SerialNo>654321</SerialNo>
        </DeviceInfo>
        <Key Id="1"
        Algorithm="urn:ietf:params:xml:ns:keyprov:pskc:hotp">
            <Issuer>Issuer</Issuer>
            <AlgorithmParameters>
                <ResponseFormat Length="8" Encoding="DECIMAL"/>
            </AlgorithmParameters>
            <Data>
                <Secret>
                    <PlainValue>MTIzNDU2Nzg5MDEyMzQ1Njc4OTA=</PlainValue>
                </Secret>
                <Counter>
                    <PlainValue>0</PlainValue>
                </Counter>
            </Data>
            <Policy>
                <StartDate>2006-05-01T00:00:00Z</StartDate>
                <ExpiryDate>2006-05-31T00:00:00Z</ExpiryDate>
            </Policy>
        </Key>
    </KeyPackage>
    <KeyPackage>
        <DeviceInfo>
            <Manufacturer>TokenVendorAcme</Manufacturer>
            <SerialNo>123456</SerialNo>
        </DeviceInfo>
        <Key Id="2"
        Algorithm="urn:ietf:params:xml:ns:keyprov:pskc:hotp">
            <Issuer>Issuer</Issuer>
            <AlgorithmParameters>
                <ResponseFormat Length="8" Encoding="DECIMAL"/>
            </AlgorithmParameters>
            <Data>
                <Secret>
                    <PlainValue>MTIzNDU2Nzg5MDEyMzQ1Njc4OTA=</PlainValue>
                </Secret>
                <Counter>
                    <PlainValue>0</PlainValue>
                </Counter>
            </Data>
            <Policy>
                <StartDate>2006-05-01T00:00:00Z</StartDate>
                <ExpiryDate>2006-05-31T00:00:00Z</ExpiryDate>
            </Policy>
        </Key>
    </KeyPackage>
    <KeyPackage>
        <DeviceInfo>
            <Manufacturer>TokenVendorAcme</Manufacturer>
            <SerialNo>9999999</SerialNo>
        </DeviceInfo>
        <Key Id="3"
        Algorithm="urn:ietf:params:xml:ns:keyprov:pskc:hotp">
            <Issuer>Issuer</Issuer>
            <AlgorithmParameters>
                <ResponseFormat Length="8" Encoding="DECIMAL"/>
            </AlgorithmParameters>
            <Data>
                <Secret>
                    <PlainValue>MTIzNDU2Nzg5MDEyMzQ1Njc4OTA=</PlainValue>
```

```
                </Secret>
                <Counter>
                    <PlainValue>0</PlainValue>
                </Counter>
            </Data>
            <Policy>
                <StartDate>2006-03-01T00:00:00Z</StartDate>
                <ExpiryDate>2006-03-31T00:00:00Z</ExpiryDate>
            </Policy>
        </Key>
    </KeyPackage>
    <KeyPackage>
        <DeviceInfo>
            <Manufacturer>TokenVendorAcme</Manufacturer>
            <SerialNo>9999999</SerialNo>
        </DeviceInfo>
        <Key Id="4"
        Algorithm="urn:ietf:params:xml:ns:keyprov:pskc:hotp">
            <Issuer>Issuer</Issuer>
            <AlgorithmParameters>
                <ResponseFormat Length="8" Encoding="DECIMAL"/>
            </AlgorithmParameters>
            <Data>
                <Secret>
                    <PlainValue>MTIzNDU2Nzg5MDEyMzQ1Njc4OTA=</PlainValue>
                </Secret>
                <Counter>
                    <PlainValue>0</PlainValue>
                </Counter>
            </Data>
            <Policy>
                <StartDate>2006-04-01T00:00:00Z</StartDate>
                <ExpiryDate>2006-04-30T00:00:00Z</ExpiryDate>
            </Policy>
        </Key>
    </KeyPackage>
</KeyContainer>
```

Рисунок 10. Пример массового представления.

9. Расширяемость

В этом разделе перечислены некоторые базовые точки расширения, обеспечиваемые PSKC.

Новая версия PSKC

При необходимости определить новую версию этого документа выделяется новый номер версии, указывающий обновленную спецификацию. Номер версии создается в атрибуте Version, как описано в разделе 4, а схема нумерации **должна** соответствовать параграфу 1.2 и правилам расширения из раздела 12.

Новые элементы XML

Использование схемы XML и доступные точки расширения позволяют добавлять новые элементы XML. В зависимости от элемента XML могут применяться разные способы расширения. В одних местах можно искользовать элемент <Extensions>, в других - точку расширения XML <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>.

Новые атрибуты XML

Схема XML позволяет добавлять новые атрибуты XML там, где определены точки расширения XML (см. "<xs: anyAttribute namespace="##other"/>" в разделе 11).

Новые профили алгоритмов PSKC

Этот документ определяет в разделе 10 два профиля алгоритмов PSKC. В информационном документе [PSKC-ALGORITHM-PROFILES] описаны дополнительные профили. Могут быть зарегистрированы дополнительные профили алгоритмов PSKC в соответствии с параграфом 12.4.

URI алгоритмов

В разделе 6 описана защита ключей и связанных с ними данных, для чего применяется множество алгоритмов. Новые алгоритмы можно применять путем указания соответствующих идентификаторов URI.

Правила

В разделе 5 определены правила, которые могут быть привязаны к ключу и относящимся к нему данным. Элемент <Policy> является одним из элементов, позволяющих разработчикам ограничивать применение ключа неким набором функций (например, только для OTP). Дополнительные значения могут регистрироваться в соответствии с разделом 12.

10. Профили алгоритмов PSKC

10.1. HOTP

Базовое имя - HOTP

Класс - OTP

URI - urn:ietf:params:xml:ns:keyprov:pskc:hotp

Определение алгоритма - [HOTP]

Определение идентификатора - (этот RFC)

Регистрационный контакт - IESG

Устарел - FALSE

Профиль

Элемент <KeyPackage> **должен** присутствовать, а элемент <ResponseFormat>, являющийся дочерним для <AlgorithmParameters>, **должен** использоваться для указания размера и формата значения OTP.

Элемент <Counter> (параграф 4.1) **должен** представляться как метаданные для ключа.

Применяются также перечисленные ниже добавочные ограничения.

- Значение элемента <Secret> при его наличии **должно** содержать не менее 16 октетов (128 битов) ключевого материала.
- Элемент <ResponseFormat> **должен** иметь атрибут Format = DECIMAL, а атрибут Length **должен** указывать размер от 6 до 9 (включительно).
- Элемент <PINPolicy> **может** присутствовать, но PINUsageMode не может иметь значения Algorithmic.

Пример можно видеть на рисунке 3.

10.2. PIN

Базовое имя - PIN

Класс - сравнение симметричных статических свидетельств (удостоверений)

URI - urn:ietf:params:xml:ns:keyprov:pskc:pin

Определение алгоритма - параграф 5.1 (этот RFC)

Определение идентификатора - (этот RFC)

Регистрационный контакт - IESG

Устарел - FALSE

Профиль

Элемент <AlgorithmParameters> **может** присутствовать, но атрибут <AlgorithmParameters> не требуется. Элемент <ResponseFormat> **можно** использовать для указания формата значения PIN.

Элемент <Secret> (параграф 4.1) **должен** представляться.

Пример можно видеть на рисунке 5.

11. Схема XML

В этом разделе определена схема XML для PSKC.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
     xmlns:pskc="urn:ietf:params:xml:ns:keyprov:pskc"
     xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
     xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
     targetNamespace="urn:ietf:params:xml:ns:keyprov:pskc"
     elementFormDefault="qualified"
     attributeFormDefault="unqualified">
     <xs:import namespace="http://www.w3.org/2000/09/xmldsig#"
          schemaLocation=
             "http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/xmldsig-core-schema.xsd"/>
     <xs:import namespace="http://www.w3.org/2001/04/xmlenc#"
          schemaLocation=
             "http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/xenc-schema.xsd"/>
     <xs:import namespace="http://www.w3.org/XML/1998/namespace"/>
     <xs:complexType name="KeyContainerType">
          <xs:sequence>
               <xs:element name="EncryptionKey"
                    type="ds:KeyInfoType" minOccurs="0"/>
               <xs:element name="MACMethod"
                    type="pskc:MACMethodType" minOccurs="0"/>
               <xs:element name="KeyPackage"
                    type="pskc:KeyPackageType" maxOccurs="unbounded"/>
               <xs:element ref="ds:Signature" minOccurs="0"/>
               <xs:element name="Extensions"
                    type="pskc:ExtensionsType"
                    minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:attribute name="Version"
               type="pskc:VersionType" use="required"/>
          <xs:attribute name="Id"
               type="xs:ID" use="optional"/>
```

```
    </xs:complexType>
    <xs:simpleType name="VersionType" final="restriction">
        <xs:restriction base="xs:string">
            <xs:pattern value="\d{1,2}\.\d{1,3}"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:complexType name="KeyType">
        <xs:sequence>
            <xs:element name="Issuer"
                type="xs:string" minOccurs="0"/>
            <xs:element name="AlgorithmParameters"
                type="pskc:AlgorithmParametersType"
                minOccurs="0"/>
            <xs:element name="KeyProfileId"
                type="xs:string" minOccurs="0"/>
            <xs:element name="KeyReference"
                type="xs:string" minOccurs="0"/>
            <xs:element name="FriendlyName"
                type="xs:string" minOccurs="0"/>
            <xs:element name="Data"
                type="pskc:KeyDataType" minOccurs="0"/>
            <xs:element name="UserId"
                type="xs:string" minOccurs="0"/>
            <xs:element name="Policy"
                type="pskc:PolicyType" minOccurs="0"/>
            <xs:element name="Extensions"
                type="pskc:ExtensionsType" minOccurs="0"
                maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="Id"
            type="xs:string" use="required"/>
        <xs:attribute name="Algorithm"
            type="pskc:KeyAlgorithmType" use="optional"/>
    </xs:complexType>
    <xs:complexType name="PolicyType">
        <xs:sequence>
            <xs:element name="StartDate"
                type="xs:dateTime" minOccurs="0"/>
            <xs:element name="ExpiryDate"
                type="xs:dateTime" minOccurs="0"/>
            <xs:element name="PINPolicy"
                type="pskc:PINPolicyType" minOccurs="0"/>
            <xs:element name="KeyUsage"
                type="pskc:KeyUsageType"
                minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="NumberOfTransactions"
                type="xs:nonNegativeInteger" minOccurs="0"/>
            <xs:any namespace="##other"
                minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="KeyDataType">
        <xs:sequence>
            <xs:element name="Secret" type="pskc:binaryDataType" minOccurs="0"/>
            <xs:element name="Counter" type="pskc:longDataType" minOccurs="0"/>
            <xs:element name="Time" type="pskc:intDataType" minOccurs="0"/>
            <xs:element name="TimeInterval" type="pskc:intDataType" minOccurs="0"/>
            <xs:element name="TimeDrift" type="pskc:intDataType" minOccurs="0"/>
            <xs:any namespace="##other" processContents="lax"
                minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="binaryDataType">
        <xs:sequence>
            <xs:choice>
                <xs:element name="PlainValue" type="xs:base64Binary"/>
                <xs:element name="EncryptedValue" type="xenc:EncryptedDataType"/>
            </xs:choice>
            <xs:element name="ValueMAC" type="xs:base64Binary" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="intDataType">
        <xs:sequence>
            <xs:choice>
                <xs:element name="PlainValue" type="xs:int"/>
                <xs:element name="EncryptedValue" type="xenc:EncryptedDataType"/>
```

```
            </xs:choice>
            <xs:element name="ValueMAC" type="xs:base64Binary" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="stringDataType">
        <xs:sequence>
            <xs:choice>
                <xs:element name="PlainValue" type="xs:string"/>
                <xs:element name="EncryptedValue" type="xenc:EncryptedDataType"/>
            </xs:choice>
            <xs:element name="ValueMAC" type="xs:base64Binary" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="longDataType">
        <xs:sequence>
            <xs:choice>
                <xs:element name="PlainValue" type="xs:long"/>
                <xs:element name="EncryptedValue" type="xenc:EncryptedDataType"/>
            </xs:choice>
            <xs:element name="ValueMAC" type="xs:base64Binary" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="PINPolicyType">
        <xs:attribute name="PINKeyId" type="xs:string" use="optional"/>
        <xs:attribute name="PINUsageMode" type="pskc:PINUsageModeType"/>
        <xs:attribute name="MaxFailedAttempts" type="xs:unsignedInt" use="optional"/>
        <xs:attribute name="MinLength" type="xs:unsignedInt" use="optional"/>
        <xs:attribute name="MaxLength" type="xs:unsignedInt" use="optional"/>
        <xs:attribute name="PINEncoding" type="pskc:ValueFormatType" use="optional"/>
        <xs:anyAttribute namespace="##other"/>
    </xs:complexType>
    <xs:simpleType name="PINUsageModeType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="Local"/>
            <xs:enumeration value="Prepend"/>
            <xs:enumeration value="Append"/>
            <xs:enumeration value="Algorithmic"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="KeyUsageType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="OTP"/>
            <xs:enumeration value="CR"/>
            <xs:enumeration value="Encrypt"/>
            <xs:enumeration value="Integrity"/>
            <xs:enumeration value="Verify"/>
            <xs:enumeration value="Unlock"/>
            <xs:enumeration value="Decrypt"/>
            <xs:enumeration value="KeyWrap"/>
            <xs:enumeration value="Unwrap"/>
            <xs:enumeration value="Derive"/>
            <xs:enumeration value="Generate"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:complexType name="DeviceInfoType">
        <xs:sequence>
            <xs:element name="Manufacturer" type="xs:string" minOccurs="0"/>
            <xs:element name="SerialNo" type="xs:string" minOccurs="0"/>
            <xs:element name="Model" type="xs:string" minOccurs="0"/>
            <xs:element name="IssueNo" type="xs:string" minOccurs="0"/>
            <xs:element name="DeviceBinding" type="xs:string" minOccurs="0"/>
            <xs:element name="StartDate" type="xs:dateTime" minOccurs="0"/>
            <xs:element name="ExpiryDate" type="xs:dateTime" minOccurs="0"/>
            <xs:element name="UserId" type="xs:string" minOccurs="0"/>
            <xs:element name="Extensions" type="pskc:ExtensionsType" minOccurs="0"
                maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="CryptoModuleInfoType">
        <xs:sequence>
            <xs:element name="Id" type="xs:string"/>
            <xs:element name="Extensions" type="pskc:ExtensionsType" minOccurs="0"
                maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="KeyPackageType">
```

```
          <xs:sequence>
               <xs:element name="DeviceInfo" type="pskc:DeviceInfoType" minOccurs="0"/>
               <xs:element name="CryptoModuleInfo"
                    type="pskc:CryptoModuleInfoType" minOccurs="0"/>
               <xs:element name="Key" type="pskc:KeyType" minOccurs="0"/>
               <xs:element name="Extensions" type="pskc:ExtensionsType" minOccurs="0"
                    maxOccurs="unbounded"/>
          </xs:sequence>
     </xs:complexType>
     <xs:complexType name="AlgorithmParametersType">
          <xs:choice>
               <xs:element name="Suite" type="xs:string" minOccurs="0"/>
               <xs:element name="ChallengeFormat" minOccurs="0">
                    <xs:complexType>
                         <xs:attribute name="Encoding"
                              type="pskc:ValueFormatType" use="required"/>
                         <xs:attribute name="Min" type="xs:unsignedInt" use="required"/>
                         <xs:attribute name="Max" type="xs:unsignedInt" use="required"/>
                         <xs:attribute name="CheckDigits" type="xs:boolean" default="false"/>
                    </xs:complexType>
               </xs:element>
               <xs:element name="ResponseFormat" minOccurs="0">
                    <xs:complexType>
                         <xs:attribute name="Encoding"
                              type="pskc:ValueFormatType" use="required"/>
                         <xs:attribute name="Length" type="xs:unsignedInt" use="required"/>
                         <xs:attribute name="CheckDigits" type="xs:boolean" default="false"/>
                    </xs:complexType>
               </xs:element>
               <xs:element name="Extensions" type="pskc:ExtensionsType" minOccurs="0"
                    maxOccurs="unbounded"/>
          </xs:choice>
     </xs:complexType>
     <xs:complexType name="ExtensionsType">
          <xs:sequence>
               <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:attribute name="definition" type="xs:anyURI" use="optional"/>
     </xs:complexType>
     <xs:simpleType name="KeyAlgorithmType">
          <xs:restriction base="xs:anyURI"/>
     </xs:simpleType>
     <xs:simpleType name="ValueFormatType">
          <xs:restriction base="xs:string">
               <xs:enumeration value="DECIMAL"/>
               <xs:enumeration value="HEXADECIMAL"/>
               <xs:enumeration value="ALPHANUMERIC"/>
               <xs:enumeration value="BASE64"/>
               <xs:enumeration value="BINARY"/>
          </xs:restriction>
     </xs:simpleType>
     <xs:complexType name="MACMethodType">
           <xs:sequence>
                  <xs:choice>
                        <xs:element name="MACKey"
                           type="xenc:EncryptedDataType" minOccurs="0"/>
                        <xs:element name="MACKeyReference" type="xs:string" minOccurs="0"/>
                        </xs:choice>
                        <xs:any namespace="##other"
           processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
       </xs:sequence>
       <xs:attribute name="Algorithm" type="xs:anyURI" use="required"/>
        </xs:complexType>
     <xs:element name="KeyContainer" type="pskc:KeyContainerType"/>
</xs:schema>
```

12. Взаимодействие с IANA

12.1. Регистрация Content-Type для application/pskc+xml

Данная спецификация регистрирует новый тип носителя в соответствии с процедурами RFC 4288 [RFC4288] и рекомендациями RFC 3023 [RFC3023].

Имя типа носителя MIME - application

Имя субтипа MIME - pskc+xml

Требуемые параметры - нет.

Необязательные параметры - набор символов

Указывает кодировку символов во вложенном XML.

Вопросы кодирования - используется XML, где могут применяться 8-битовые символы в зависимости от применяемой кодировки (см. прараграф 3.2 в RFC 3023 [RFC3023]).

Вопросы безопасности - см. раздел 13 в RFC 6030.

Проблемы взаимодействия - нет.

Опубликованная спецификация - RFC 6030.

Приложения, использующие этот тип носителя - этот тип может служить контейнером для симметричных ключей при их передаче и представлении (общие секреты OTP или симметричные ключи шифрования) разным типам устройств строгой аутентификации. В таком качестве он применяется в системах представления ключей.

Дополнительная информация

Magic Number - нет.

Расширение имен файлов - .pskcxml

Код типа файлов Macintosh - TEXT

Лицо и почтовый адрес для дополнительной информации - Philip Hoyer, Philip.Hoyer@actividentity.com

Предусмотренное применение - ограниченное использование

Ограничения на использование - нет

Автор - данная спецификация создана рабочей группой IETF KEYPROV, имеющей список рассылки <keyprov@ietf.org>.

Контроль изменений - IESG <iesg@ietf.org>

12.2. Регистрация схемы XML

This section registers an XML schema as per the guidelines in [RFC3688].

URI: urn:ietf:params:xml:schema:keyprov:pskc

Регистрационный контакт - IETF KEYPROV Working Group, Philip Hoyer (Philip.Hoyer@actividentity.com).

XML Schema: The XML schema to be registered is contained in Section 11. Its first line is

```
<?xml version="1.0" encoding="UTF-8"?>
```

and its last line is

```
</xs:schema>
```

12.3. Регистрация субпространства имен URN

This section registers a new XML namespace, "urn:ietf:params:xml:ns:keyprov:pskc", per the guidelines in [RFC3688].

URI: urn:ietf:params:xml:ns:keyprov:pskc

Регистрационный контакт - IETF KEYPROV Working Group, Philip Hoyer (Philip.Hoyer@actividentity.com).

XML:

```
BEGIN
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML Basic 1.0//EN"
  "http://www.w3.org/TR/xhtml-basic/xhtml-basic10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <meta http-equiv="content-type"
        content="text/html;charset=iso-8859-1"/>
  <title>PSKC Namespace</title>
</head>
<body>
  <h1>Namespace for PSKC</h1>
  <h2>urn:ietf:params:xml:ns:keyprov:pskc</h2>
<p>See <a href="http://www.rfc-editor.org/rfc/rfc6030.txt">
 RFC 6030</a>.</p>
</body>
</html>
END
```

12.4. Реестр профилей для алгоритма PSKC

Агентство IANA создало реестр для профилей алгоритмов PSKC в соответствии с RFC 5226 [RFC5226].

Как часть этого реестра IANA поддерживает приведенную ниже информацию.

Базовое имя - имя, которым обычно обозначают профиль алгоритма PSKC.

Класс - тип записи в реестре профилей алгоритмов PSKC (например, шифрование - encryption, MAC, одноразовый пароль - OTP, подпись - digest).

URI - идентификатор профиля.

Определение идентификатора - IANA будет добавлять указатель на спецификацию, содержащую сведения о регистрации профиля алгоритма PSKC.

Определение алгоритма - ссылка на стабильный документ, в котором определен алгоритм, используемый с PSKC.

Регистрационный контакт - контактные данные стороны, подавшей запрос на регистрацию.

Устарел - TRUE, если запись была отменена на основе одобрения экспертов и профиль не **следует** применять в новых реализациях; в остальных случаях FALSE.

Профиль PSKC - информация об элементах PSKC XML и атрибутах, которые будут (или не будут) применяться с этим профилем PSKC.

Регистрация профиля алгоритма PSKC выполняется по процедуре Specification Required в соответствии с RFC 5226 [RFC5226], для обновления достаточно одобрения экспертов. На основе такого одобрения можно указать запись как устаревшую (deprecated). Экспертов назначает IESG.

Агентство IANA добавило в реестр две записи для профилей алгоритмов, описанных в разделе10.

12.5. Реестр версий PSKC

Агентство IANA создало реестр номеров версий PSKC с показанной в таблице структурой.

Версия PSKC	***Спецификация***
1.0	RFC 6030

Для добавления новой версии PSKC требуется стандартизация. Старение, удаление или изменение имеющихся версий PSKC не предполагаются.

12.6. Реестр использования ключей

Агентство IANA создало реестр использования ключей. Описание элемента <KeyUsage> приведено в разделе 5.

Как часть этого реестра IANA будет поддерживать указанную ниже информацию.

Использование ключа - идентификатор искоользования ключа.

Спецификация - IANA будет добавлять указатель на документ, содержащий информацию о семантике новой регистрации использования ключей.

Устарел - TRUE, если запись была отменена на основе одобрения экспертов и профиль не **следует** применять в новых реализациях; в остальных случаях FALSE.

Агентство IANA включило в реестр начальные значения, представленные ниже.

Применение ключа	***Спецификация***	***Устарел***
OTP	[раздел 5]	FALSE
CR	[раздел 5]	FALSE
Encrypt	[раздел 5]	FALSE
Integrity	[раздел 5]	FALSE
Verify	[раздел 5]	FALSE
Unlock	[раздел 5]	FALSE
Decrypt	[раздел 5]	FALSE
KeyWrap	[раздел 5]	FALSE
Unwrap	[раздел 5]	FALSE
Derive	[раздел 5]	FALSE
Generate	[раздел 5]	FALSE

Регистрация в реестре Key Usage выполняется по процедуре Specification Required в соответствии с RFC 5226 [RFC5226]. Для определения новых значений Key Usage применяется процедура Expert Review. На основе такого одобрения можно указать запись как устаревшую (deprecated). Экспертов назначает IESG.

13. Вопросы безопасности

Переносимый контейнер с симметричным ключом (PSKC) содержит конфиденциальную информацию (например, криптографические ключи) и может передаваться из одной защищенной области в другую. Например, контейнер, находящийся в защищенной области внутреннего сервера обеспечения, может доставляться через Internet на устройство конечного пользователя, подключенное к персональному компьютеру. Это означает, что **должны** приниматься особые меры по защите конфиденциальности, целостности и подлинности информации в контейнере.

13.1. Конфиденциальность PSKC

Конструкция контейнера позволяет применять два основных подхода для сохранения конфиденциальности содержащейся в контейнере информации при ее транспортировке.

Во-первых, содержимое контейнера (payload) может быть зашифровано.

В этом случае не требуется защиты на транспортном уровне. При выборе криптографического алгоритма для шифрования данных контейнера ключей применяются стандартные рекомендации по защите. **Следует** использовать симметричный криптографический шифр и увеличение размера ключа повышает уровень защищенности. В параграфе 6.1 даны рекомендации по защите ключевых данных с использованием симметричных криптографических шифров. В тех случаях, когда обмен ключами шифрования между отправителем и получателем невозможен, может применяться асимметричное шифрование ключевых данных, описанное в параграфе 6.3. Как и при симметричном шифровании увеличение размера ключа повышает уровень защиты.

Если ключевые данные зашифрованы с помощью метода, использующего пароль (PBE[1]), как описано в параграфе 6.2, эти данные могут быть подвержены атакам по словарю для расшифровки данных. Здесь применяются стандартные методы выбора стойких паролей.

В дополнение к этому настоятельно **рекомендуется** в практических реализациях в шифрованием PBE применять PBESalt и PBEIterationCount. Для лучшей защиты **следует** использовать свое значение PBESalt для каждого PSKC.

Другой подход к защите конфиденциальности ключевых данных основан на использовании механизмов защиты нижележащий уровней (например, [TLS], [IPsec]). Организуется защищенное соединение между защищенной областью источника (сервер обеспечения в приведенном выше примере) и целевой областью (подключенное к компьютеру конечного пользователя устройство) с шифрованием проходящих через соединение сообщений. В этом режиме сами ключевые данные не шифруются, а защищенное соединение шифрует и подписывает каждое сообщение.

Поскольку нешифрованые данные PSKC защищаются лишь транспортным уровнем, практическая реализация **должна** обеспечивать защиту от MITM[2]-атак. Аутентификация конечных точек соединения крайне важна для устранения злоумышленников, которые могут нарушить конфиденциальность PSKC.

13.2. Целостность PSKC

PSKC обеспечивает средства защиты целостности информации с помощью цифровых подписей. Для эффективной защиты **рекомендуется** включать в цифровую подпись все содержимое контейнера PSKC. Это гарантирует целостность всех атрибутов, а также позволит проверить PSKC после доставки контейнера по каналу связи.

13.3. Подлинность PSKC

Цифровая подпись PSKC является основным способом проверки подлинности. Получателю контейнера **следует** использовать связанный с подписью открытый ключ для проверки подлинности отправителя путем отслеживания пути к предварительно загруженному открытому ключу или сертификату. Отметим, что цифровую подпись PSKC можно проверять даже после доставки по защищенному каналу связи.

Гарантию подлинности можно обеспечить с помощью [TLS] или [IPsec]. Однако в этом случае невозможна будет проверка подлинности после доставки контейнера и разрыва соединения. Поскольку конечные точки TLS могут не совпадать с конечными точками представления ключей, это решение слабее цифровой подписи PSKC.

14. Участники работы

Спасибо Hannes Tschofenig за его вклад в текст этого документа.

15. Благодарности

Авторы благодарны Apostol Vassilev, Shuh Chang, Jon Martinson, Siddhart Bajaj, Stu Vaeth, Kevin Lewis, Philip Hallam-Baker, Andrea Doherty, Magnus Nystrom, Tim Moses, Anders Rundgren, Sean Turner и особенно Robert Philpott за их отклики.

Спасибо Sean Turner за рецензию в январе 2009 г. Спасибо также Anders Rundgren за инициирование обсуждения выбора алгоритмов шифрования (KW-AES-128 в сравнении с AES-128-CBC) и вклад в расчет цифровых подписей с ключами.

Эта работа основана на более ранних работах членов группы OATH[3] [OATH] по заданию формата, который можно свободно распространять в техническом сообществе.

16. Литература

16.1. Нормативные документы

[FIPS197] National Institute of Standards, "FIPS Pub 197: Advanced Encryption Standard (AES)", November 2001.

[HOTP] M'Raihi, D., Bellare, M., Hoornaert, F., Naccache, D., and O. Ranen, "HOTP: An HMAC-Based One-Time Password Algorithm", RFC 4226, December 2005.

[IANAPENREG] IANA, "Private Enterprise Numbers", <http://www.iana.org>.

[ISOIEC7812] ISO, "ISO/IEC 7812-1:2006 Identification cards -- Identification of issuers -- Part 1: Numbering system", October 2006, <http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=39698>.

[OATHMAN] OATH, "List of OATH Manufacturer Prefixes (omp)", April 2009, <http://www.openauthentication.org/oath-id/prefixes/>.

[PKCS5] RSA Laboratories, "PKCS #5: Password-Based Cryptography Standard", Version 2.0, March 1999, <http://www.rsasecurity.com/rsalabs/pkcs/>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3023] Murata, M., St. Laurent, S., and D. Kohn, "XML Media Types", RFC 3023, January 2001.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[1]Password-based encryption - шифрование на основе пароля.
[2]Man-in-the-middle - перехват и изменение данных с участием человека.
[3]Initiative for Open AuTHentication - инициатива открытой аутентификации.

[RFC4288] Freed, N. and J. Klensin, "Media Type Specifications and Registration Procedures", BCP 13, RFC 4288, December 2005.

[RFC4514] Zeilenga, K., "Lightweight Directory Access Protocol (LDAP): String Representation of Distinguished Names", RFC 4514, June 2006.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, October 2006.

[RFC5646] Phillips, A. and M. Davis, "Tags for Identifying Languages", BCP 47, RFC 5646, September 2009.

[RFC5649] Housley, R. and M. Dworkin, "Advanced Encryption Standard (AES) Key Wrap with Padding Algorithm", RFC 5649, September 2009.

[SP800-67] National Institute of Standards, "NIST Special Publication 800-67 Version 1.1: Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher", NIST Special Publication 800-67, May 2008.

[W3C.REC-xmlschema-2-20041028] Malhotra, A. and P. Biron, "XML Schema Part 2: Datatypes Second Edition", World Wide Web Consortium Recommendation REC-xmlschema-2-20041028, October 2004, <http://www.w3.org/TR/2004/REC-xmlschema-2-20041028>.

[XMLDSIG] Solo, D., Reagle, J., and D. Eastlake, "XML-Signature Syntax and Processing", World Wide Web Consortium FirstEdition REC-xmldsig-core-20020212, February 2002, <http://www.w3.org/TR/2002/REC-xmldsig-core-20020212>.

[XMLENC] Eastlake, D., "XML Encryption Syntax and Processing.", W3C Recommendation, December 2002, <http://www.w3.org/TR/xmlenc-core/>.

[XMLENC11] Reagle, J. and D. Eastlake, "XML Encryption Syntax and Processing Version 1.1", World Wide Web Consortium WD WD-xmlenc-core1-20090730, July 2009, <http://www.w3.org/TR/2009/WD-xmlenc-core1-20090730>.

16.2. Дополнительная литература

[CAP] MasterCard International, "Chip Authentication Program Functional Architecture", September 2004.

[IPsec] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[NIST800-57] Barker, E., Barker, W., Burr, W., Polk, W., and M. Smid, "NIST Special Publication 800-57, Recommendation for Key Management Part 1: General (Revised)", NIST Special Publication 800-57, March 2007.

[OATH] "Initiative for Open AuTHentication", <http://www.openauthentication.org>.

[PSKC-ALGORITHM-PROFILES] Hoyer, P., Pei, M., Machani, S., and A. Doherty, "Additional Portable Symmetric Key Container (PSKC) Algorithm Profiles", Work in Progress, May 2010.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[TLS] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[XMLNS] Hollander, D., Bray, T., and A. Layman, "Namespaces in XML", World Wide Web Consortium FirstEdition REC-xml-names-19990114, January 1999, <http://www.w3.org/TR/1999/REC-xml-names-19990114>.

Приложение A. Примеры использования

В этом приложении описан широкий набор вариантов применения, который стал мотивом разработки спецификации. Эти требования были использованы для выведения основных требований к разработке, рассмотренных ниже.

Эти варианты использования помогут также разобраться с применимостью спецификации к реальным задачам.

A.1. Интерактивное применение

В этом параграфе описаны варианты применения с предоставлением ключей на основе интерактивного протокола.

A.1.1. Доставка ключей от сервера в криптографический модуль

Например, мобильный пользователь хочет получить симметричный ключ для криптографического модуля в своем устройстве. Криптографический модуль производства компании A инициирует процесс предоставления от системы производителя B с использованием стандартного протокола предоставления. Предоставляющий элемент доставляет один или множество ключей в стандартном формате, который может быть обработан мобильным устройством.

В представленном выше варианте вместо пользовательского мобильного телефона ключ может быть предоставлен программному маркеру пользовательского приложения на мобильном компьютере с использованием сетевого протокола. Как и раньше, система предоставления доставит ключ в стандартном формате, который может быть обработан программным маркером на ПК.

Еще в одном варианте конечный пользователь или эмитент ключа хочет обновить или настроить имеющийся ключ и запрашивает контейнер замены ключа. Контейнер может включать (или не включать) новый ключ и может включать новые или измененные атрибуты ключа, такие как новое значение счетчика для применения в HOTP, измененный формат или размер отклика, новое имя и т. п.

A.1.2. Доставка ключей между криптографическими модулями

Пользователь хочет перенести ключ из одного криптографического модуля в другой. Это могут быть модули на одном компьютере или модули на компьютере и мобильном телефоне, когда пользователь хочет перенести ключ из компьютера в телефон. Пользователь может экспортировать ключ и связанные с ним данные в стандартном формате для ввода в другой криптографический модуль.

A.1.3. Доставка ключей из криптографического модуля на сервер

Пользователь хочет активировать и применять новый ключ и связанные с ним данные для системы проверки, которая не знает этот ключ. Ключ может быть встроен в криптографический модуль (например, SD-карта или диск USB), который пользователь купил в местном магазине. Вместе с криптографическим модулем пользователь может получить ключ на компакт-диске (CD) или дискете в стандартном формате. Пользователь может загрузить ключ на сервер по защищенному каналу или импортировать ключ и связанные данные в систему проверки и начать работу с ключом.

A.1.4. Массовый импорт и экспорт ключей с сервера на сервер

Время от времени системе управления ключами может требоваться массовый импорт или экспорт ключей с одного элемента на другой.

Например, вместо импорта ключей от производителя через файл сервер проверки может загрузить ключи с помощью интерактивного протокола. Ключи могут быть загружены в стандартном формате и обработаны системой проверки.

В приведенном выше варианте шлюз представления ключей OTA[1], обеспечивающий ключами мобильные телефоны, может получать ключевой материал от эмитента ключей по интерактивному протоколу. Ключи представляются в стандартном формате, обрабатываются шлюзом и последовательно передаются пользователям телефонов.

A.2. Автономное использование

В этом параграфе описаны варианты транспортировки ключей между системами по отдельному каналу (offline) с использованием модели экспорта-импорта.

A.2.1. Массовый экспорт и импорт ключей между серверами

Например, криптографические модули (такие как маркеры аутентификации OTP) могут иметь симметричные ключи, инициализируемые в процессе производства, что требует массового копирования ключей и данных алгоритма для загрузки в систему аутентификации через файл на подключаемом носителе. Производитель представляет ключи и связанные с ними данные в форме файла, содержащего записи в стандартном формате (обычно на CD). Отметим, что производители маркеров и системы проверки могут быть разными. Некоторые криптомодули позволяют локально управлять PIN-кодом (устройство имеет PIN-клавиатуру), поэтому исходные случайные значения PIN, заданные при производстве, следует передавать вместе с ключами, которые они защищают.

Предприятие может переносить ключи и связанные данные из имеющейся системы проверки A в другую систему B. Существующая система обеспечивает предприятию функциональностью, позволяющей экспортировать ключи и связанные данные (например, для маркеров аутентификации OTP) в стандартном формате. Поскольку маркеры OTP имеют стандартный формат, предприятие может импортировать записи маркеров в новую систему проверки B и начать использование имеющихся маркеров. Отметим, что производители двух систем проверки могут быть разными.

Приложение B. Требования

В этом приложении описаны наиболее важные требования, ставшие основой работы. Некоторые требование были выведены из рассмотренных выше вариантов применения.

R1

Формат **должен** поддерживать передачу множества симметричных ключей и связанных с ними атрибутов для разных алгоритмов, включая HOTP, другие OTP, Challenge/Response и т. п.

R2

Формат **должен** обслуживать сами ключи, а также атрибуты, обычно связываемые с симметричными ключами. Некоторые из таких атрибутов перечислены ниже.

- Уникальный идентификатор ключа.
- Данные об эмитенте.
- Идентификатор алгоритма.
- Режим алгоритма.
- Имя эмитента.
- Удобное имя ключа.
- Значение счетчика событий (фактор перемещения для алгоритмов OTP).
- Значение времени.

R3

Формату **следует** поддерживать интерактивный и автономный режим, т. е. следует поддерживать вывод в файл, а также использовать интерактивный протокол представления.

R4

Формату **следует** поддерживать массовое представление симметричных ключей.

R5

Формату **следует** поддерживать массовое представление значений PIN, относящихся к ключам.

[1]Over-The-Air - по воздуху (через эфир)

R6

Формату **следует** поддерживать переносимость на разные платформы, а также **следует** обеспечивать эффективность расчетов.

R7

Формат **должен** обеспечивать приемлемый уровень защиты в смысле шифрования и целостности данных.

R8

В интерактивном режиме формату **не следует** полагаться на защиту транспортного уровня (например, SSL/TLS) для основных требований безопасности.

R9

Формату **следует** быть расширяемым и обеспечивать точки расширения, позволяющие производителя задавать в будущем дополнительные атрибуты.

R10

Формату **следует** разрешать распространение данных вывода ключей без самих ключей. Это нужно для поддержки схем динамического управления симметричными ключами на основе алгоритмов вывода ключей на основе предварительно установленного первичного ключа. Данные для вывода ключей обычно содержат ссылку на ключ, а не само значение ключа.

R11

Формату **следует** разрешать дополнительные операции жизненного цикла, такие как ресинхронизация счетчиков. Это требует защиты конфиденциальности между клиентом и сервером, поэтому можно применять базовый формат защищенного контейнера без передачи ключевого материала.

R12

Формат **должен** поддерживать использование заранее распространенных общих симметричных ключей для защиты конфидинциальности элементов данных.

R13

Формат **должен** поддерживать основанное на паролях шифрование (PBE) [PKCS5] для защиты конфиденциальных элементов данных. Это широко применяется в разных вариантах представления.

R14

Формату **следует** поддерживать асимметричные алгоритмы шифрования (например, RSA) для сквозной защиты конфиденциальных элементов данных. Это нужно в ситуациях, где сложно применить предустановленные общие ключи шифрования.

Адреса авторов

Philip Hoyer

ActivIdentity, Inc.

117 Waterloo Road

London, SE1 8UL

UK

Phone: +44 (0) 20 7960 0220

EMail: phoyer@actividentity.com

Mingliang Pei

VeriSign, Inc.

487 E. Middlefield Road

Mountain View, CA 94043

USA

Phone: +1 650 426 5173

EMail: mpei@verisign.com

Salah Machani

Diversinet, Inc.

2225 Sheppard Avenue East

Suite 1801

Toronto, Ontario M2J 5C2

Canada

Phone: +1 416 756 2324 Ext. 321

EMail: smachani@diversinet.com

Перевод на русский язык

Николай Малых

nmalykh@protocols.ru