Internet Engineering Task Force (IETF)
Request for Comments: 6492
Category: Standards Track
ISSN: 2070-1721

G. Huston
R. Loomans
B. Ellacott
APNIC
R. Austein
ISC

February 2012

Протокол обеспечения сертификатами ресурсов

A Protocol for Provisioning Resource Certificates

Тезисы

Этот документ определяет модель взаимодействий при управлении сертификатами между эмитентом ресурса INR[1] (issuer) и получателем INR (subject) путем задания спецификации протокола взаимодействия между этими сторонами. Протокол поддерживает передачу запросов от субъекта и соответствующих откликов от эмитента, относящихся к выпуску сертификатов, их отзыву и отчетам о статусе сертификатов. Протокол предназначен лишь для использования приложениями для управления сертификатами INR и не рассчитан на применение в более общих моделях управления сертификатами.

Статус документа

Этот документ является проектом стандарта (Internet Standards Track).

Документ является результатом работы IETF[2] и представляет собой согласованное мнение сообщества IETF. Документ был вынесен на публичное рассмотрение и одобрен для публикации IESG[3]. Дополнительная информация о документах BCP представлена в разделе 2 документа RFC 5741.

Информация о текущем статусе этого документа, обнаруженных ошибках и способах обратной связи может быть найдена по ссылке http://www.rfc-editor.org/info/rfc6492.

Оглавление

[1]Internet Number Resource — числовые ресурсы (номера) Internet.
[2]Internet Engineering Task Force.
[3]Internet Engineering Steering Group.

1. Введение

Этот документ определяет модель взаимодействий при управлении сертификатами между эмитентом ресурса INR (issuer) и получателем INR (subject) путем задания спецификации протокола взаимодействия между этими сторонами. Протокол поддерживает передачу запросов от субъекта и соответствующих откликов от эмитента, относящихся к выпуску сертификатов, их отзыву и отчетам о статусе сертификатов. Протокол предназначен лишь для использования приложениями для управления сертификатами INR и не рассчитан на применение в более общих моделях управления сертификатами.

1.1. Терминология

Ниже приведены определения используемых в документе терминов.

Internet Number Resource (или resource) -числовой ресурс Internet (ресурс)
В контексте этого документа обозначает номера автономных систем (AS[1]), адреса IP версии 4 и IP версии 6.

issuer - эмитент
В контексте этого документа обозначает объект (сущность), играющий роль «издателя» ресурса. Эмитентами являются удостоверяющие центры (УЦ или CA[2]) и они могут выпускать сертификаты ресурсов.

Subject - субъект
В контексте этого документа обозначает элемент (сущность), играющий роль получателя ресурса, указанного в качестве субъекта сертификата. Субъект может обладать сертификатом CA, позволяющим ему выступать в качестве эмитента (издателя).

resource class — класс ресурса
Класс ресурсов указывает на группу ресурсов, которые эмитент может быть сертифицированы одним сертификатом ресурса.

Server - сервер
В контексте данной спецификации протокола «клиент-сервер» роль сервера возлагается на эмитентов.

client
В контексте данной спецификации протокола «клиент-сервер» роль клиента возлагается на субъектов.

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

2. Область действия

Этот протокол обеспечения инфраструктуры сертификатов открытых ключей ресурсов (RPKI[3]) определяет базовый набор взаимодействий, которые позволяют субъектам запросить выпуск сертификатов, отозвать сертификаты и получить информацию о статусе от эмитента, а эмитентам — поддерживать множество выпущенных сертификатов, согласованное с записями о выделении, относящимися к каждому субъекту. База данных эмитента о выделении ресурсов является полномочным источником информации о выделении ресурсов, которые эмитент может сертифицировать для субъекта.

Получатель ресурса (субъект) может также играть роль издателя ресурсов (эмитент).

Данная спецификация не охватывает:

- подписывание объектов с использованием ключей, сертифицированных сертификатами объектов, а также выпуск сертификатов конечных элементов;
- взаимодействие с базой данных эмитента о выделении ресурсов, а также протокол поддержки репозитория публикаций;

[1]Autonomous System.
[2]Certification Authority — агентство по выдаче сертификатов.
[3]Resource Public Key Infrastructure.

- взаимодействие между клиентом и сервером, обеспечивающее отождествление сторон, или обмен сертификатами и контекстом проверки пригодности PKI[1], используемым в синтаксисе обмена криптографическими сообщениями (CMS[2]) [RFC5652];
- взаимодействие между клиентом и сервером, позволяющее установить для локального времени подписи CMS взаимно согласованные значения.

3. Спецификация протокола

Данный протокол обеспечения сертификатов RPKI заключается в простом взаимодействии «запрос-отклик», где клиент передает запросы серверу, а тот генерирует соответствующие отклики.

Протокол реализуется в форме обмена сообщениями.

Сообщения передаются через сквозное соединение HTTP [RFC2616]. Обмен сообщениями начинается с того, что клиент инициирует HTTP POST с типом содержимого "application/rpki-updown" и объектом message в теле. Отклик сервера похож на отклик HTTP — объект message содержится в теле сообщения, а тип содержимого отклика указан "application/rpki-updown". Содержимое запроса POST и отклика сервера является «хорошо сформированным» объектом CMS [RFC5652], представленным с использованием правил DER[3] для ASN.1 [X.509-88] и отформатированным в соответствии с профилем CMS, заданным в следующем параграфе. CMS используется в качестве формата подписи для подписывания объекта сообщения. Сообщение CMS включает сертификат конечного элемента (EE[4]), который используется для проверки пригодности цифровой подписи CMS (см. параграф 3.1.1.4); цепочка сертификации, используемая для проверки пригодности сертификата EE, **может** включаться в сообщение CMS, а при ее отсутствии предполагается, что коммуникации между двумя элементами осуществляются и с использованием механизмов, не определенных в данной спецификации.

Взаимодействие «запрос-отклик» в рамках этого протокола предполагается надежным и на каждый запрос **должен** генерироваться отклик. Протокол требует упорядоченной обработки для каждого отдельного клиента и серверу **недопустимо** воспринимать клиентский запрос, пока не передан отклик на предшествующий запрос этого клиента. Попытки клиента инициировать множество запросов параллельно (т. е. множество одновременных запросов с одинаковым атрибутом отправителя — см. параграф 3.2) **должны** обнаруживаться сервером и отвергаться с передачей сообщения об ошибке (т. е. отклика с кодом 1101).

3.1. Профиль CMS

Ниже показан формат объекта CMS.

```
ContentInfo ::= SEQUENCE {
  contentType ContentType,
  content [0] EXPLICIT ANY DEFINED BY contentType }

ContentType ::= OBJECT IDENTIFIER
```

Поле content-type представляет собой signed-data типа id-data, а именно id-signedData, OID = 1.2.840.113549.1.7.2. [RFC5652]

3.1.1. Тип SignedData

В соответствии со стандартом CMS [RFC5652] типом содержимого signed-data является ASN.1 типа SignedData.

```
SignedData ::= SEQUENCE {
    version CMSVersion,
    digestAlgorithms DigestAlgorithmIdentifiers,
    encapContentInfo EncapsulatedContentInfo,
    certificates [0] IMPLICIT CertificateSet OPTIONAL,
    crls [1] IMPLICIT RevocationInfoChoices OPTIONAL,
    signerInfos SignerInfos }

  DigestAlgorithmIdentifiers ::= SET OF DigestAlgorithmIdentifier
  SignerInfos ::= SET OF SignerInfo
```

Кроме того, поле SignerInfos **должно** включать только один объект SignerInfo.

3.1.1.1. version

Поле version указывает номер версии синтаксиса. Оно должно иметь значение 3, соответствующее структуре signerInfo, имеющей номер версии 3.

3.1.1.2. digestAlgorithms

Множество digestAlgorithms содержит идентификаторы объектов (OID[5]) для алгоритмов подписи, используемых при подписывании инкапсулированного содержимого. Это множество **должно** включать в точности один идентификатор алгоритма подписи и значение OID **должно** выбираться из числа указанных в [RFC6485].

3.1.1.3. encapContentInfo

Поле encapContentInfo представляет собой подписанное содержимое, состоящее из идентификатора типа и самого содержимого. encapContentInfo содержит информацию (payload) протокола обеспечения сертификатами RPKI.

---

[1]Public Key Infrastructure — инфраструктура открытых ключей.
[2]Cryptographic Message Syntax — синтаксис криптографических сообщений.
[3]Distinguished Encoding Rules — правила отличительного кодирования.
[4]End-entity.
[5]Object Identifier.

```
EncapsulatedContentInfo ::= SEQUENCE {
   eContentType ContentType,
   eContent [0] EXPLICIT OCTET STRING OPTIONAL }

ContentType ::= OBJECT IDENTIFIER
```

3.1.1.3.1. eContentType

Поле eContentType для объекта RPKI Protocol Message определено как id-ct-xml и имеет числовое значение 1.2.840.113549.1.9.16.1.28.

```
id-smime OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840)
                           rsadsi(113549) pkcs(1) pkcs9(9) 16 }

id-ct OBJECT IDENTIFIER ::= { id-smime 1 }

id-ct-xml OBJECT IDENTIFIER ::= { id-ct 28 }
```

3.1.1.3.2. eContent

Содержимое RPKI XML Protocol Object включает одно протокольное сообщение, структурированное в соответствии с определенной схемой XML, как определено в последующих параграфах. Поле eContent объекта CMS формально определяется с использованием ASN.1 как

```
RPKIXMLProtocolObject ::= OCTET STRING -- сообщение в формате XML
```

3.1.1.4. certificates

Это поле **должно** присутствовать и **должно** содержать один сертификат EE для пары ключей, секретный ключ из которой был использован для подписи CMS. В это поле **недопустимо** помещать сертификат RPKI, ему **следует** содержать сертификат, который признан отождествляющим сторону, которая создала объект CMS.

Это поле **может** содержать сертификаты CA, которые зависимая сторона **может** использовать для проверки пригодности сертификата EE.

3.1.1.5. crls

Это поле **должно** присутствовать. Содержимое поля задано в [RFC5652]. В этом поле **должен** быть указан текущий список отзыва (CRL[1]), выпущенный тем же CA, который выпустил сертификат EE для ключевой пары, чей секретный ключ был использован для подписи CMS. Поле **может** содержать списки CRL, выпущенные другими CA и покрывающие сертификаты CA, включенные в поле certificates объекта CMS (см. параграф 3.1.1.4). Включение каких-либо иных CRL в это поле **недопустимо**.

3.1.1.6. SignerInfo

Структура SignerInfo определена в CMS как показано ниже.

```
SignerInfo ::= SEQUENCE {
  version CMSVersion,
  sid SignerIdentifier,
  digestAlgorithm DigestAlgorithmIdentifier,
  signedAttrs [0] IMPLICIT SignedAttributes OPTIONAL,
  signatureAlgorithm SignatureAlgorithmIdentifier,
  signature SignatureValue,
  unsignedAttrs [1] IMPLICIT UnsignedAttributes OPTIONAL }
```

3.1.1.6.1. version

Поле version **должно** иметь значение 3, соответствующее выбору SubjectKeyIdentifier для sid.

3.1.1.6.2. sid

Определение поля sid приведено ниже.

```
SignerIdentifier ::= CHOICE {
  issuerAndSerialNumber IssuerAndSerialNumber,
  subjectKeyIdentifier [0] SubjectKeyIdentifier }
```

В данном профиле поле sid **должно** иметь значение SubjectKeyIdentifier, присутствующее в сертификате EE, содержащемся в поле certificates сообщения CMS.

3.1.1.6.3. digestAlgorithm

Поле digestAlgorithm **должно** содержать идентификатор OID алгоритма хэширования (digest), который соответствует спецификации профиля RPKI для алгоритмов и размеров ключей [RFC6485].

3.1.1.6.4. signedAttrs

Определение поля signedAttrs приведено ниже.

```
SignedAttributes ::= SET SIZE (1..MAX) OF Attribute

UnsignedAttributes ::= SET SIZE (1..MAX) OF Attribute

Attribute ::= SEQUENCE {
  attrType OBJECT IDENTIFIER,
  attrValues SET OF AttributeValue }
```

[1] Certificate Revocation List — список отозванных сертификатов.

```
AttributeValue ::= ANY
```

Элемент signedAttr **должен** присутствовать и **должен** включать атрибуты content-type и message-digest [RFC5652]. Если присутствует один из двух или оба атрибута signing-time [RFC5652] и binary-signing-time [RFC6019], они также **должны** включаться в SignedAttributes. Включение других подписанных атрибутов **недопустимо**.
Поле signedAttr **должно** включать лишь единственный экземпляр любого конкретного атрибута. Кроме того, хотя синтаксис разрешает SET OF AttributeValue, в данном профиле поле attrValues **должно** содержать только одно значение AttributeValue.

3.1.1.6.4.1. Атрибут Content-Type

Атрибут content-type **должен** присутствовать. Идентификатор attrType OID для атрибута content-type имеет значение 1.2.840.113549.1.9.3.

```
id-contentType OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) rsadsi(113549) pkcs(1) pkcs9(9) 3 }

ContentType ::= OBJECT IDENTIFIER
```

Поле attrValues для атрибута content-type **должно** соответствовать eContentType в EncapsulatedContentInfo. Это значение OID определяется как id-ct-xml и имеет числовое представление 1.2.840.113549.1.9.16.1.28.

3.1.1.6.4.2. Атрибут Message-Digest

Атрибут message-digest **должен** присутствовать. Идентификатор attrType OID для атрибута message-digest имеет значение 1.2.840.113549.1.9.4.

```
id-messageDigest OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) rsadsi(113549) pkcs(1) pkcs9(9) 4 }

MessageDigest ::= OCTET STRING
```

Поле attrValues для атрибута message-digest содержит выход алгоритма хэширования, примененного к подписываемому содержимому, как указано в параграфе 5.4 [RFC5652].

3.1.1.6.4.3. Атрибут Signing-Time

Атрибут signing-time **может** присутствовать. Идентификатор attrType OID для атрибута signing-time имеет значение 1.2.840.113549.1.9.5.

```
id-signingTime OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) rsadsi(113549) pkcs(1) pkcs9(9) 5 }

SigningTime ::= Time

Time ::= CHOICE {
  utcTime UTCTime,
  generalizedTime GeneralizedTime }
```

Атрибут signing-time указывает время (по локальным часам), когда для содержимого была создана цифровая подпись.

Рекомендации по использованию UTCTime и GeneralizedTime в атрибуте signing-time даны в параграфе 11.3 [RFC5652].

Один или оба атрибута signing-time и binary-signing-time **должны** присутствовать. При наличии обоих атрибутов они **должны** указывать одинаковое время.

3.1.1.6.4.4. Атрибут Binary-Signing-Time

Атрибут binary-signing-time **может** присутствовать. Идентификатор attrType OID для атрибута binary-signing-time имеет значение 1.2.840.113549.1.9.16.2.46.

```
id-aa-binarySigningTime OBJECT IDENTIFIER ::= { iso(1)
    member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9)
    smime(16) aa(2) 46 }

BinarySigningTime ::= BinaryTime

BinaryTime ::= INTEGER (0..MAX)
```

Атрибут binary-signing-time указывает время (по локальным часам), когда для содержимого была создана цифровая подпись. Опредеелние точности для атрибута binary-signing-time приведено в [RFC6019].

Один или оба атрибута signing-time и binary-signing-time **должны** присутствовать. При наличии обоих атрибутов они **должны** указывать одинаковое время.

3.1.1.6.5. Атрибут signatureAlgorithm

Поле signatureAlgorithm **должно** соответствовать спецификации профиля алгоритмов и размеров ключей RPKI [RFC6485].

3.1.1.6.6. Атрибут signature

Определение значения подписи приведено ниже.

```
SignatureValue ::= OCTET STRING
```

Характеристики подписи определяются алгоритмами хэширования и подписи.

3.1.1.6.7. Неподписанные атрибуты

Поле unsignedAttrs **должно** быть опущено.

3.1.2. Проверка пригодности объекта CMS

До того как получатель подписанного объекта CMS сможет использовать содержимое этого объекта, он **должен** проверить пригодность подписанного объекта, убедившись, что выполняются все приведенные ниже условия (проверки могут выполняться в любом порядке).

1. Объект CMS имеет корректный формат и синтаксис подписанных объектов соответствует данной спецификации. В частности выполняются все приведенные ниже условия.
   a. Типом содержимого (content-type) в объекте CMS является объект SignedData (OID 1.2.840.113549.1.7.2).
   b. Поле version для объекта SignedData имеет значение 3.
   c. Поле certificates присутствует в объекте SignedData и содержит один сертификат EE, поле SubjectKeyIdentifier в котором соответствует полю sid объекта SignerInfo.
   d. Поле crls присутствует в объекте SignedData.
   e. Поле version в SignerInfo имеет значение 3.
   f. Поле signedAttrs присутствует в объекте SignerInfo и содержит по одному атрибуту content-type (OID 1.2.840.113549.1.9.3) и message-digest (OID 1.2.840.113549.1.9.4), а также один или оба атрибута signing-time (OID 1.2.840.113549.1.9.5) и binary-signing-time (OID 1.2.840.113549.1.9.16.2.46) с одним значением времени. Других атрибутов присутствовать не должно.
   g. Поле eContentType в EncapsulatedContentInfo представляет собой идентификатор OID, который соответствует attrValue в атрибуте content-type и имеет attrValue id-ct-xml.
   h. Поле unsignedAttrs в объекте SignerInfo опущено.
   i. При наличии обоих атрибутов signing-time и binary-signing-time их значения представляют одно время.
   j. Поля digestAlgorithm в объектах SignedData и SignerInfo соответствуют профилю алгоритмов и размеров ключей [RFC6485].
   k. Поле signatureAlgorithm в объекте SignerInfo соответствует профилю алгоритмов и размеров ключей [RFC6485].
   l. Подписанный объект использует DER-представление.
2. Открытый ключ сертификата EE (содержащийся внутри объекта CMS signed-data) может быть использован для успешной проверки подписи в подписанном объекте.
3. Сертификат EE (содержащийся внутри объекта CMS signed-data) является пригодным к использованию сертификатом EE. В частности, существует пригодный путь сертификации от доверенной привязки, выбранной получателем, до данного сертификата EE.
4. В текущий момент сертификат EE не отозван. Это можно определить, подтвердив, что список CRL, содержащийся в поле crls подписанного объекта данных CMS, является текущим пригодным к использованию списком CRL, выпущенным тем же CA, который выпустил сертификат EE, и этот CRL не включает порядкового номера сертификата EE.
5. Время, представленное атрибутом signing-time или binary-signing-time, не меньше значения времени, указанного в предыдущем пригодном объекте CMS, который был передан данному получателю от того же источника. Это значение времени подписи **может** лежать в интервале Validity Time сертификата EE, но сертификат EE **не следует** считать непригодным, если это не выполняется, а все остальные перечисленные выше проверки завершились успешно.

3.1.3. Спецификация ASN.1 для подписанного объекта CMS

Ниже приведена спецификация ASN.1 для подписанного объекта данных CMS, используемого протоколом обеспечения RPKI.

```
ContentInfo ::= SEQUENCE {
  contentType ContentType,
  content [0] EXPLICIT ANY DEFINED BY contentType }

ContentType ::= OBJECT IDENTIFIER

id-smime OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840)
                         rsadsi(113549) pkcs(1) pkcs9(9) 16 }
id-ct OBJECT IDENTIFIER ::= { id-smime 1 }

id-ct-xml OBJECT IDENTIFIER ::= { id-ct 28 }

RPKIXMLProtocolObject ::= OCTET STRING -- XML encoded message

id-signedData OBJECT IDENTIFIER ::= { iso(1) member-body(2)
                    us(840) rsadsi(113549) pkcs(1) pkcs7(7) 2 }

SignedData ::= SEQUENCE {
  version CMSVersion,
```

```
      digestAlgorithms DigestAlgorithmIdentifiers,
      encapContentInfo EncapsulatedContentInfo,
      certificates [0] IMPLICIT CertificateSet OPTIONAL,
      crls [1] IMPLICIT RevocationInfoChoices OPTIONAL,
      signerInfos SignerInfos }

    DigestAlgorithmIdentifiers ::= SET OF DigestAlgorithmIdentifier

    SignerInfos ::= SET OF SignerInfo

    SignerInfo ::= SEQUENCE {
      version CMSVersion,
      sid SignerIdentifier,
      digestAlgorithm DigestAlgorithmIdentifier,
      signedAttrs [0] IMPLICIT SignedAttributes OPTIONAL,
      signatureAlgorithm SignatureAlgorithmIdentifier,
      signature SignatureValue,
      unsignedAttrs [1] IMPLICIT UnsignedAttributes OPTIONAL }

    SignerIdentifier ::= CHOICE {
      issuerAndSerialNumber IssuerAndSerialNumber,
      subjectKeyIdentifier [0] SubjectKeyIdentifier }

    SignedAttributes ::= SET SIZE (1..MAX) OF Attribute

    Attribute ::= SEQUENCE {
    attrType OBJECT IDENTIFIER,
    attrValues SET OF AttributeValue }

    AttributeValue ::= ANY

    SignatureValue ::= OCTET STRING

    id-contentType OBJECT IDENTIFIER ::= { iso(1) member-body(2)
        us(840) rsadsi(113549) pkcs(1) pkcs9(9) 3 }

    ContentType ::= OBJECT IDENTIFIER

    id-messageDigest OBJECT IDENTIFIER ::= { iso(1) member-body(2)
        us(840) rsadsi(113549) pkcs(1) pkcs9(9) 4 }

    MessageDigest ::= OCTET STRING

    id-signingTime OBJECT IDENTIFIER ::= { iso(1) member-body(2)
        us(840) rsadsi(113549) pkcs(1) pkcs9(9) 5 }

    SigningTime ::= Time

    Time ::= CHOICE {
      utcTime UTCTime,
      generalizedTime GeneralizedTime }

    id-aa-binarySigningTime OBJECT IDENTIFIER ::= { iso(1)
        member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9)
        smime(16) aa(2) 46 }

    BinarySigningTime ::= BinaryTime

    BinaryTime ::= INTEGER (0..MAX)
```

3.2. Общий формат сообщений

Неформальное описание шаблона XML приведено ниже (компактная форма RELAX NG, формально описывающая объекты протокольных сообщений, приведена в параграфе 3.7).

```
   ------------------------------------------------------------
   <?xml version="1.0" encoding="UTF-8"?>
   <message xmlns="http://www.apnic.net/specs/rescerts/up-down/"
            version="1"
            sender="sender name"
            recipient="recipient name"
            type="message type">
   [payload]
   </message>
   ------------------------------------------------------------
```

version

Значение этого атрибута указывает версию данного протокола. Этот документ описывает версию 1.

sender
Значением этого атрибута является согласованное ранее между клиентом и сервером имя отправителя сообщения.

recipient
Значением этого атрибута является согласованное ранее между клиентом и сервером имя получателя сообщения.

type
Возможными значениями этого атрибута являются list, list_response, issue, issue_response, revoke, revoke_response и error_response.

Соответствующие спецификации анализаторы (parser) **должны** отвергать любые документы с непонятным номером версии, а также с любыми непонятными элементами или атрибутами. Серверы должны генерировать сообщение об ошибке при получении таких запросов. Клиентам следует генерировать сообщение об ошибке при получении таких откликов.

Инкапсулированное содержимое CMS представляет собой документ XML. В оставшейся части этой спецификации инкапсуляция CMS не рассматривается и обсуждаются только документы XML.

Сообщение проверяется в соответствии с приведенным ниже списком.

1. Проверяется корректность формата CMS (см. п. 1 в параграфе 3.1.2).
2. Проверяется корректность формата XML.
3. Проверяется соответствие атрибутов XML sender и recipient известному клиенту и данному серверу для запросов и ранее адресованному серверу и данному клиенту для откликов.
4. Проверяется цифровая подпись с использованием открытого ключа, содержащегося в сертификате внутри CMS (см. п. 2 в параграфе 3.1.2).
5. Проверяется пригодность сертификата, представленного CMS, с использованием инфраструктуры PKI, которая может быть определена предшествующим соглашением между клиентом и сервером (см. п. 3 в параграфе 3.1.2).
6. Проверяется что время подписи CMS не раньше времени подписи предыдущего сообщения, принятого данным получателем от того же отправителя (см. п. 4 в параграфе 3.1.2).
7. Проверяется значение номера версии сообщение (должно быть 1).

Проверки **следует** проводить в указанном выше порядке.

Любые ошибки при проверках 1 - 7 **должны** вызывать на сервере генерацию отклика "HTTP 400 Bad Request" на запрос HTTP POST. Ошибка при проверке 7 **должна** заставлять сервер генерировать отклик "Request-Not-Performed". При возникновении у клиента любой ошибки при проверках 1 — 7 ему **следует** генерировать сигнал ошибки.

Сервер **может** использовать управление потоком данных для контроля скорости обработки запросов. Если запрос не обрабатывается в результате ограничений такого контроля, сервер **может** генерировать отклик "HTTP 503 Service Unavailable". Отклик HTTP 503 **может** включать заголовок HTTP Retry-After: в качестве совета клиенту.

3.3. Управление — запрос класса ресурса

Эти запросы используются клиентами для получения от сервера списка всех ресурсов, которые выделены или назначены сервером для данного клиента. Кроме того, отклики серверов будут включать копию текущих сертификатов, выпущенных CA сервера, в которых данный клиент указан субъектом.

3.3.1. Запрос класса ресурсов

Запросное сообщение имеет вид:

```
type="list"
-----------------------------------------------------------------
Payload:
[для этого типа запросов данных не определено]
-----------------------------------------------------------------
```

3.3.2. Отклик для класса ресурсов

Сообщение с откликом имеет вид:

```
   type="list_response"
-----------------------------------------------------------------
Payload:
 <class class_name="class name"
     cert_url="url"
     resource_set_as="as resource set"
     resource_set_ipv4="ipv4 resource set"
     resource_set_ipv6="ipv6 resource set"
     resource_set_notafter="datetime"
     suggested_sia_head="[directory uri]" >
     <certificate cert_url="url"
         req_resource_set_as="as resource set"
         req_resource_set_ipv4="ipv4 resource set"
         req_resource_set_ipv6="ipv6 resource set" >
     [certificate]
     </certificate>
     ...
```

```
        (повторяется для всех действующих сертификатов, где клиент указан субъектом)
        <issuer>[сертификат эмитента]</issuer>
        </class>
    ...
    (повторяется для всех классов ресурсов эмитента, которые были выделены клиенту)
  ------------------------------------------------------------
```

Если клиенту выделены ресурсы из нескольких классов, отклик **должен** содержать множество элементов class, соответствующее полному набору классов ресурсов эмитента, в которых клиенту были выделены ресурсы. Классы ресурсов эмитента, в которых клиенту не были выделены ресурсы, **недопустимо** включать в отклик.

Если эмитент имеет множество сертификатов в каком-либо классе ресурсов, подписанных разными ключами (такое может возникать при поэтапной смене ключей эмитентом), в отклике указывается лишь последний (наиболее свежий) сертификат, выпущенный с «активным» в данный момент ключом эмитента.

Каждый элемент class описывает набор ресурсов, которые сертифицированы в области действия одного сертификата и относятся к одному классу с общим путем проверки пригодности.

class_name
Этот атрибут содержит выделенное эмитентом имя его класса ресурсов.

cert_url
В контексте элемента класса этот атрибут является указателем на сертификат CA для эмитента (т. е. указателем на непосредственный вышестоящий сертификат, в котором эмитент является субъектом). Это значение представляет собой разделенный запятыми список URI, в котором по крайней мере один **должен** быть rsync URI [RFC5781]. Символы запятой в URI **должны** представляться escape-последовательностями %2C. Список может рассматриваться клиентом как упорядоченный издателем этого сертификата по уровню предпочтения набор вариантов доступа.

resource_set_as
В контексте элемента класса этот атрибут является множеством номеров и диапазонов номеров AS, которые эмитент выделил клиенту в области действия этого класса ресурсов, представленных в форме списка разделенных запятыми строк ASCII. Элементы списка являются десятичными целыми числами или диапазонами, заданными нижним и верхним значением с символом дефиса в качестве разделителя, с использованием канонического порядка, описанного в [RFC3779], без ведущих нулей и символов пробела или знаков препинания, отличающихся от запятой и дефиса (например, resource_set_as="123,456-789,123456"). Если в этом классе ресурсов нет номеров AS, указывается пустой набор AS в виде "".

resource_set_ipv4
В контексте элемента класса этот атрибут является множеством адресов IPv4, которые эмитент выделил клиенту в области действия этого класса ресурсов. Значение представляется в форме списка строк ASCII, разделенных запятыми. Каждый элемент списка представляет собой адресный префикс в формате <адрес с разделением точками>/размер маски или диапазон, заданный младшим и старшим значением в формате десятичных чисел с разделением точками и символом дефиса между границами диапазона. Список представляется в каноническом порядке, описанном в [RFC3779]. Компоненты адреса, разделяемые точками, указываются без ведущих нулей. Кроме того, в списке не должны присутствовать пробелы и знаки препинания, отличающиеся от запятой, точки, дробной черты и дефиса (например, resource_set_ipv4="192.0.2.0/26,192.0.2.66-192.0.2.76"). Если в этом классе ресурсов нет адресов IPv4, атрибут представляется пустым адресом IPv4 в форме "".

resource_set_ipv6
В контексте элемента класса этот атрибут является множеством адресов IPv6, которые эмитент выделил клиенту в области действия этого класса ресурсов. Значение представляется в форме списка строк ASCII, разделенных запятыми. Каждый элемент списка представляет собой адресный префикс в формате <последовательность шестнадцатеричных полубайтов>/размер маски, или диапазон, заданный младшим и старшим адресом с разделением символом дефиса. Нули в конце адресов опускаются и представляются двумя символами двоеточия (::). Список представляется в каноническом порядке, описанном в [RFC3779]. Последовательность шестнадцатеричных полубайтов указывается без ведущих нулей и в списке не должно присутствовать пробелов и знаков препинания, отличающихся запятой, точки, дробной черты и дефиса, а формат списка должен соответствовать [RFC5952] (например, resource_set_ipv6="2001:db8::/48,2001:db8:2::-2001:db8:5::"). Тип данных XML Schema описан по ссылке http://www.w3.org/TR/xmlschema-2/#hexBinary, регистр символов не принимается во внимание, но канонический формат использует нижний регистр. Если в этом классе ресурсов нет адресов IPv6, атрибут представляется пустым адресом IPv6 в форме "".

resource_set_notafter
Значение этого атрибута задает дату и время, которые будут установлены в поле Validity notAfter любого нового сертификата, выпускаемого для данного клиента в области действия класса ресурсов при запросе клиентом нового сертификата. Время указывается в формате ISO 8601 [ISO.8601:2004] и **должно** использовать время часового пояса UTC, представленное как YYYY-MM-DDThh:mm:ssZ (например, 2007-11-29T04:40:00Z). Если в сертификате клиента указано другое значение notAfter, этому клиенту **следует** запросить выпуск нового сертификата для этого класса ресурсов.

suggested_sia_head (не обязательно)
При наличии этого поля оно содержит URI каталога, соответствующего точке публикации репозитория, которую сервер делает доступной для клиента с целью использования для клиентского набора публикуемой продукции. Данная спецификация не задает протоколов, которые клиент может использовать для публикации объектов в этой точке публикации оператора репозитория.

[issuer's certificate]
Значение этого поля содержит представление Base64 для DER-кодированного сертификата CA для эмитента (сертификат с поддержкой CA, в котором эмитент является субъектом).
Каждый элемент с сертификатом описывает самый свежий текущий сертификат, в котором субъект указывает на клиента для каждой активной пары ключей клиента. Текущим считается не просроченный и не отозванный сертификат. Если такого сертификата не было выпущено, отклик не будут включать элемента с сертификатом.

cert_url
: В контексте элемента с сертификатом это поле указывает место, в котором эмитент опубликовал данный сертификат. Это поле является предложением издателя в части значения поля AIA[1] для субъекта с целью использования в подчиненных сертификатах, выпускаемых этим субъектом. Согласно профилю сертификатов ресурсов [RFC6487] поле AIA является непустым списком (содержит хотя бы 1 элемент) URI, один из которых **должен** быть rsync URI [RFC5781]. Порядок URI в поле AIA может интерпретироваться как относительные предпочтения издателя в части методов доступа к этому сертификату. Поле cert_url соответствует этой спецификации AIA. Значение поля представляет собой список разделенных запятыми URI, один из которых **должен** быть rsync URI. Запятые внутри URI **должны** заменяться escape-последовательностью %2C.

req_resource_set_as
: Множество номеров AS, которые были указаны в соответствующем исходном запросе сертификата, определяющее максимальную запрошенную область действия сертифицированного множества номеров AS (синтаксис описан ниже). При наличии этого атрибута в запросе сертификата атрибут **должен** включаться в данный отклик, в остальных случаях его присутствие **недопустимо**.

req_resource_set_ipv4
: Множество адресов IPv4, которые были указаны в соответствующем исходном запросе сертификата, определяющее максимальную запрошенную область действия сертифицированного множества номеров IPv4 (синтаксис описан ниже). При наличии этого атрибута в запросе сертификата атрибут **должен** включаться в данный отклик, в остальных случаях его присутствие **недопустимо**.

req_resource_set_ipv6
: Множество адресов IPv6, которые были указаны в соответствующем исходном запросе сертификата, определяющее максимальную запрошенную область действия сертифицированного множества номеров IPv6 (синтаксис описан ниже). При наличии этого атрибута в запросе сертификата атрибут **должен** включаться в данный отклик, в остальных случаях его присутствие **недопустимо**.

[certificate]
: Значение в формате Base64 для DER-представления сертификата.

3.4. CA — выпуск сертификата

Такие сообщения используются клиентами для запроса у серверного CA выпуска сертификатов для ресурсов, которые были выделены или назначены клиенту. При успешном выполнении запроса отклик сервера включает выпущенный сертификат.

3.4.1. Запрос на выпуск сертификата

Сообщение с запросом имеет вид:

```
      type="issue"
   -----------------------------------------------------------------
   Payload:
   <request
           class_name="class name"
           req_resource_set_as="as resource set"
           req_resource_set_ipv4="ipv4 resource set"
           req_resource_set_ipv6="ipv6 resource set">
           [Certificate request]
            </request>
   -----------------------------------------------------------------
```

Клиент **должен** использовать разные пары ключей для каждого класса ресурсов.

Атрибуты req_resource_set в запросе не обязательны.

Если не задан ни один из атрибутов req_resource_set, это означает, что в выпущенный сертификат будет включен полный набор всех ресурсов, выделенных к настоящему времени запрашивающему сертификат клиенту.

Если какой-либо из атрибутов req_resource_set указан в запросе, отсутствующие атрибуты req_resource_set будут интерпретироваться, как указание полного набора ресурсов соответствующего класса для включения в запрошенный клиентом сертификат.

Если значение любого из включенных атрибутов req_resource_set пусто (""), это указывает, что ресурсы соответствующего типа не нужно включать в запрошенный сертификат.

Значения множеств ресурсов из запроса сохраняются эмитентом в локальной записи, связанной с классом ресурсов и открытым ключом клиента. Любой последующий запрос на выпуск сертификата, указывающий тот же класс ресурсов и тот же открытый ключ клиента, будут сбрасывать (создавать заново) локальную запись эмитента для запрошенного набора ресурсов с включением в новую запись значения из последнего запроса.

class_name
: Серверный идентификатор класса ресурсов.

req_resource_set_as (OPTIONAL)
: Множество номеров AS, определяющее максимальную запрашиваемую область действия сертифицированного набора номеров AS и отформатированное в соответствии с атрибутом resource_set_as в отклике.

req_resource_set_ipv4 (OPTIONAL)
: Множество адресов IPv4, определяющее максимальную запрашиваемую область действия сертифицированного набора адресов IPv4 и отформатированное в соответствии с атрибутом resource_set_ipv4 в отклике.

req_resource_set_ipv6 (OPTIONAL)
: Множество адресов IPv6 , определяющее максимальную запрашиваемую область действия сертифицированного набора адресов IPv6 и отформатированное в соответствии с атрибутом resource_set_ipv6 в отклике.

[1] Authority Information Access — доступ к информации об УЦ (агентстве)

[Certificate request]

Запрос сертификата в формате Base64 для DER-версии запроса, отформатированной с использованием PKCS#10 [RFC2986]. Запрос сертификата подписывается с использованием секретного ключа из пары, открытая часть которой служит значением ключа субъекта в запросе сертификации. Алгоритм подписи задан в [RFC6485] (эта подпись предназначена для подтверждения прав владения секретным ключом).

Ответом на этот запрос является Certificate Issuance Response, если запрос может быть выполнен в интерактивном режиме. Если запрос не может быть обработан сразу же, сервер **должен** ответить сообщением Request-Not-Performed с использованием подходящего кода ошибки (см. ниже).

- Если ресурс не определен сервером, **должен** возвращаться код ошибки 1201.
- Если клиенту не выделено ресурсов данного класса, сервер **должен** вернуть код ошибки 1202 и не обрабатывать запрос.
- Если информационные поля (payload) запроса имеют некорректный формат, сервер **должен** вернуть код ошибки 1203.
- Если использованный в запросе сертификата открытый ключ показывает попытку клиента применить одну и ту же пару ключей для нескольких классов ресурсов, сервер **должен** передать код ошибки 1204.
- если эмитент сертификат использует автономный (off-line) процесс для создания сертификата и сервер не может сразу же ответить на запрос выпуска сертификата, эмитент **должен** ответить на первый экземпляр запроса кодом ошибки 1104 для индикации асинхронной обработки запроса. На последующие повторы этого запроса, пока выполняется автономный процесс выпуска сертификата, **следует** отвечать кодом ошибки 1101. В этом контексте, где в выпуске сертификата участвует автономный процесс, если эмитент в процессе обработки запроса определит, что новый сертификат полностью идентичен выпущенному последним сертификату для того же клиента и отличается от того лишь порядковым номером, эмитент может принять решение о возврате предыдущего сертификата без вовлечения автономного процесса для создания нового.

Отметим, что клиент, получивший на запрос сертификата отклик с кодом 1104, **может** периодически повторять свой запрос и в таких случаях клиент **должен** получать отклики с кодом 1101, пока обрабатывается запрос и сообщение Certificate Issuance Response по завершении процесса выпуска сертификата. В таких обстоятельствах клиенту **следует** ограничивать частоту повторов (не более 1 запроса в течение 24 часов).

3.4.2. Отклик на запрос выпуска сертификата

Сообщение с откликом имеет вид:

```
   type="issue_response"
-----------------------------------------------------------------
   Payload:
    <class class_name="class name"
           cert_url="url"
           resource_set_as="as resource set"
           resource_set_ipv4="ipv4 resource set"
           resource_set_ipv6="ipv6 resource set" >
            <certificate cert_url="url"
                  req_resource_set_as="as resource set"
                  req_resource_set_ipv4="ipv4 resource set"
                  req_resource_set_ipv6="ipv6 resource set" >
            [certificate]
            </certificate>
            <issuer>[issuer's certificate]</issuer>
          </class>
   -----------------------------------------------------------------
```

Если эмитент в процессе обработки запроса определит, что новый сертификат полностью идентичен выпущенному последним сертификату для того же клиента и отличается от того лишь порядковым номером, эмитент может принять решение о возврате предыдущего сертификата без создания нового.

Определения атрибутов и синтаксис значений совпадают с используемыми для запроса списка классов ресурсов, но отклик указывает только (один) именованный класс ресурсов и выпускается только (один) сертификат, соответствующий открытому ключу клиента в соответствующем запросе сертификата.

3.5. Отзыв сертификата

Такой запрос «отменяет» (retires) клиентскую пару ключей путем запроса у серверного CA отзыва всех сертификатов для данного клиента (т. е., сертификатов, где субъектом является данный клиент), которые содержат соответствующий открытый ключ в рамках именованного класса ресурсов. Отдельные сертификаты в рамках данного протокола не могут быть отозваны.

3.5.1. Запрос на отзыв сертификата

Сообщение с запросом отзыва имеет вид:

```
   type="revoke"
-----------------------------------------------------------------
  Payload:
  <key class_name="class name"
       ski="[encoded hash of the subject public key]" />
-----------------------------------------------------------------
```

Это сообщение служит серверному CA командой незамедлительно пометить все действующие сертификаты, выпущенные этим эмитентом в именованном классе ресурсов с этим клиентом в качестве субъекта и представленным значением SKI, как отозванные, что влечет за собой отзыв этих сертификатов из опубликованного репозитория и включение в последующие CRL для данного класса ресурсов. Эмитент **должен** убедиться в том, что все отзываемые сертификаты были выпущены с именем запросившего их отзыв клиента в качестве субъекта сертификатов.

class_name
Выделенное эмитентом имя класса ресурсов.

ski
Кодированное хэш-значение открытого ключа клиента, который будет отозван. Кодирование представляет собой создание 160-битового хэша SHA-1 открытого ключа клиента, в соответствии с методом (1) из параграфа 4.2.1.2 в [RFC5280] и кодирование результата с использованием представления Base 64 с URL and Filename Safe Alphabet в соответствии с определением раздела 5 в [RFC4648].

3.5.2. Отклик на отзыв сертификата

Сообщение с откликом на запрос отзыва имеет вид:

```
   type="revoke_response"
-----------------------------------------------------------------
   Payload:
   <key class_name="class name"
     ski="[encoded hash of the subject public key]" />
-----------------------------------------------------------------
```

class_name
Выделенное эмитентом имя класса ресурсов.

ski
Кодированное хэш-значение открытого ключа клиента, который будет отозван. Кодирование представляет собой создание 160-битового хэша SHA-1 открытого ключа клиента, в соответствии с методом (1) из параграфа 4.2.1.2 в [RFC5280] и кодирование результата с использованием представления Base 64 с URL and Filename Safe Alphabet в соответствии с определением раздела 5 в [RFC4648].

3.6. Отклик Request-Not-Performed

Сообщение об отказе от выполнения запроса имеет вид:

```
   type="error_response"
-----------------------------------------------------------------
   Payload:
   <status>[Code]</status>
   <description xml:lang="en-US">[Readable text]</description>
-----------------------------------------------------------------
```

Во всех случаях, когда возникает ошибка в результате несогласованности или наличия ошибок в содержимом запроса, а также по причине наличия на сервере состояния, препятствующего выполнению запроса, создается отклик Request-Not-Performed.

description
Текстовое поле, которое **можно** включать в отклик. В рамках данного протокола трактовка значения этого поля не определяется и реализации могут предполагать наличие в нем некой информации, предназначенной для человека. Если запрос HTTP, вызвавший генерацию этого отклика, включал заголовок Accept-Language, как определено в параграфе 14.4 спецификации HTTP/1.1 [RFC2616], сервер **может** включить второй описательный элемент с использованием самого предпочтительного из указанных клиентом языков. Описание с использованием языка en-US **должно** включаться в текст, если он присутствует.

Коды ошибок приведены в таблице.

Код	*Описание*
1101	Уже обработанный запрос
1102	Ошибка в номере версии
1103	Не распознанный тип запроса
1104	Запрос запланирован для обработки
1201	Запрос — нет такого класса ресурсов
1202	Запрос — не выделено ресурсов этого класса
1203	Запрос — некорректно сформированный запрос сертификата
1204	Запрос — ключ уже используется
1301	Отзыв — нет такого класса ресурсов
1302	Отзыв — нет такого ключа
2001	Внутренняя ошибка сервера, запрос не выполнен

3.7. Схема XML

Ниже приведена компактная форма RELAX NG, описывающая версию 1 этого протокола.

Примечание. Как отмечено в [XML], "названию пространства имен, служащего для определенных целей, **следует** быть уникальным и достаточно постоянным. Не задается цели его непосредственной применимости для поиска схемы (если таковая существует)".

```
default namespace = "http://www.apnic.net/specs/rescerts/up-down/"

grammar {
   resource_set_as = xsd:string { maxLength="512000" pattern="[\-,0-9]*" }
   resource_set_ip4 = xsd:string { maxLength="512000" pattern="[\-,/.0-9]*" }
   resource_set_ip6 = xsd:string { maxLength="512000" pattern="[\-,/:0-9a-fA-F]*" }

   class_name = xsd:token { minLength="1" maxLength="1024" }
   ski = xsd:token { minLength="27" maxLength="1024" }

   label = xsd:token { minLength="1" maxLength="1024" }
   cert_url = xsd:string { minLength="10" maxLength="4096" }
   base64_binary = xsd:base64Binary { minLength="4" maxLength="512000" }

   start = element message {
     attribute version { xsd:positiveInteger { maxInclusive="1" } },
     attribute sender { label },
     attribute recipient { label },
     payload
   }

   payload |= attribute type { "list" }, list_request
   payload |= attribute type { "list_response"}, list_response
   payload |= attribute type { "issue" }, issue_request
   payload |= attribute type { "issue_response"}, issue_response
   payload |= attribute type { "revoke" }, revoke_request
   payload |= attribute type { "revoke_response"}, revoke_response
   payload |= attribute type { "error_response"}, error_response

   list_request = empty
   list_response = class*

   class = element class {
     attribute class_name { class_name },
     attribute cert_url { cert_url },
     attribute resource_set_as { resource_set_as },
     attribute resource_set_ipv4 { resource_set_ip4 },
     attribute resource_set_ipv6 { resource_set_ip6 },
     attribute resource_set_notafter { xsd:dateTime },
     attribute suggested_sia_head { xsd:anyURI { maxLength="1024" pattern="rsync://.+"} }?,
     element certificate {
       attribute cert_url { cert_url },
       attribute req_resource_set_as { resource_set_as }?,
       attribute req_resource_set_ipv4 { resource_set_ip4 }?,
       attribute req_resource_set_ipv6 { resource_set_ip6 }?,
       base64_binary
     }*,
     element issuer { base64_binary }
   }

   issue_request = element request {
     attribute class_name { class_name },
     attribute req_resource_set_as { resource_set_as }?,
     attribute req_resource_set_ipv4 { resource_set_ip4 }?,
     attribute req_resource_set_ipv6 { resource_set_ip6 }?,
     base64_binary
   }
   issue_response = class

   revoke_request = revocation
   revoke_response = revocation

   revocation = element key {
     attribute class_name { class_name },
     attribute ski { ski }
   }

   error_response =
     element status { xsd:positiveInteger { maxInclusive="9999" } },
     element description { attribute xml:lang { xsd:language },
                           xsd:string { maxLength="1024" } }*
   }
```

4. Вопросы безопасности

Этот протокол поддерживает обслуживание сертификатов ресурсов, которые эмитент выпускает для субъекта при сертификации ресурсов, которые были выделены или назначены эмитентом для данного субъекта [RFC6480].

Протокол предполагает, что эмитент и субъект известны один другому и обменялись представлениями для взаимного признания цифровых подписей, используемых в сообщениях CMS. Механизмы обмена такими представлениями не описаны в данной спецификации.

Протокол представляет собой минимальный протокол «запрос-отклик», в котором строго задана последовательность транзакций, что снижает вероятность потери субъектом и эмитентом синхронизации статуса выпущенных сертификатов.

Проверка пригодности протокольных объектов (параграф 3.1.2) требует, чтобы значение времени подписи CMS было не меньше значения времени, переданного в предыдущем пригодном протокольном объекте от того же инициатора к тому же получателю. Если одна из сторон по какой-то отправляет пригодное сообщение (объект протокола), подписанное в будущем времени, последующие сообщения от этой стороны в том же контексте клиент-сервер, могут использовать значение времени подписи с учетом предыдущей ошибки, чтобы время подписи каждого последующего сообщения было не раньше времени подписи предыдущего пригодного сообщения (отметим, что не задается нормативного требования по части соответствия времени подписи реальному времени, что обеспечивает устойчивость к произвольным временным сдвигам в контексте протокольного обмена сообщениями). Если клиент и сервер хотят использовать в качестве времени подписи взаимно согласованное значение, такое согласование (как было отмечено в разделе 2) не является частью данного протокола.

5. Взаимодействие с IANA

Агентство IANA зарегистрировало приведенный ниже тип среды (media type).

```
   application/rpki-updown
```

5.1. application/rpki-updown

```
Type name:  application
Subtype name:  rpki-updown
Required parameters:  None
Optional parameters:  None
Encoding considerations:  binary
Security considerations:  Carries an RPKI Provisioning Protocol Message, as defined in this
   document.
Interoperability considerations:  None
Published specification:  This document
Applications that use this media type:  HTTP [RFC5652]
Additional information:
   Magic number(s):  None
   File extension(s):
   Macintosh File Type Code(s):
Person & email address to contact for further information: Geoff Huston <gih@apnic.net>
Intended usage:  COMMON
Restrictions on usage:  Only to be used as an RPKI Provisioning Protocol message object type,
   as defined in this document.
Author:  Geoff Huston <gih@apnic.net>
Change controller:  Geoff Huston <gih@apnic.net>
```

6. Благодарности

Авторы отмечают с признательностью существенный вклад Russ Housley, Steve Kent, Randy Bush, George Michaelson, Robert Kisteleki, Tim Bruijnzeels и Carsten Bormann в подготовку протокола, описанного в этом документе.

7. Литература

7.1. Нормативные документы

[ISO.8601:2004] ISO, "ISO 8601:2004 Representation of dates and Times", 2004.

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

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[RFC2986] Nystrom, M. and B. Kaliski, "PKCS #10: Certification Request Syntax Specification Version 1.7", RFC 2986, November 2000.

[RFC3779] Lynn, C., Kent, S., and K. Seo, "X.509 Extensions for IP Addresses and AS Identifiers", RFC 3779, June 2004.

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

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.

[RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, September 2009.

[RFC5781] Weiler, S., Ward, D., and R. Housley, "The rsync URI Scheme", RFC 5781, February 2010.

[RFC5952] Kawamura, S. and M. Kawashima, "A Recommendation for IPv6 Address Text Representation", RFC 5952, August 2010.

[RFC6019] Housley, R., "BinaryTime: An Alternate Format for Representing Date and Time in ASN.1", RFC 6019, September 2010.

[RFC6485] Huston, G., "The Profile for Algorithms and Key Sizes for Use in the Resource Public Key Infrastructure (RPKI)", RFC 6485, February 2012.

[X.509-88] CCITT, "Recommendation X.509: The Directory-Authentication Framework", 1988.

7.2. Дополнительная литература

[RFC6480] Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", RFC 6480, February 2012.

[RFC6487] Huston, G., Michaelson, G., and R. Loomans, "A Profile for X.509 PKIX Resource Certificates", RFC 6487, February 2012.

[XML] Bray, T., Hollander, D., Layman, A., Tobin, R., and H. Thompson, "Namespaces in XML 1.0 (Third Edition)", World Wide Web Consortium Recommendation REC-xml-names-20091208, December 2009, <http://www.w3.org/TR/2009/REC-xml-names-20091208/>.

Адреса авторов

Geoff Huston

APNIC

EMail: gih@apnic.net

URI: http://www.apnic.net

Robert Loomans

APNIC

EMail: robertl@apnic.net

URI: http://www.apnic.net

Byron Ellacott

APNIC

EMail: bje@apnic.net

URI: http://www.apnic.net

Rob Austein

Internet Systems Consortium

EMail: sra@hactrn.net

Перевод на русский язык

Николай Малых

nmalykh@gmail.com