

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 23, 2014

S. Weiler
SPARTA, Inc.
A. Sonalker
Battelle Memorial Institute
R. Austein
Dragon Research Labs
October 20, 2013

A Publication Protocol for the Resource Public Key Infrastructure (RPKI)
[draft-ietf-sidr-publication-04](#)

Abstract

This document defines a protocol for publishing Resource Public Key Infrastructure (RPKI) objects. Even though the RPKI will have many participants issuing certificates and creating other objects, it is operationally useful to consolidate the publication of those objects. This document provides the protocol for doing so.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 23, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.	Introduction	3
1.1.	Terminology	3
2.	Context	3
3.	Protocol Specification	3
3.1.	Common Details	4
3.1.1.	Common XML Message Format	4
3.2.	Control Sub-Protocol	5
3.2.1.	Config Object	5
3.2.2.	Client Object	5
3.3.	Publication Sub-Protocol	6
3.4.	Error handling	7
3.5.	XML Schema	7
4.	Examples	11
4.1.	<config/> Set Query	11
4.2.	<config/> Set Reply	12
4.3.	<config/> Get Query	12
4.4.	<config/> Get Reply	13
4.5.	<client/> Create Query	13
4.6.	<client/> Create Reply	14
4.7.	<client/> Set Query	14
4.8.	<client/> Set Reply	15
4.9.	<client/> Get Query	15
4.10.	<client/> Get Reply	15
4.11.	<client/> List Query	16
4.12.	<client/> List Reply	16
4.13.	<client/> Destroy Query	17
4.14.	<client/> Destroy Reply	17
4.15.	<publish/> Query	17
4.16.	<publish/> Reply	18
4.17.	<withdraw/> Query	18
4.18.	<withdraw/> Reply	19
4.19.	<report_error/> With Text	19
4.20.	<report_error/> Without Text	19
5.	Operational Considerations	19
6.	IANA Considerations	20
7.	Security Considerations	21
8.	References	21
8.1.	Normative References	21
8.2.	Informative References	22
	Authors' Addresses	22

1. Introduction

This document assumes a working knowledge of the Resource Public Key Infrastructure (RPKI), which is intended to support improved routing security on the Internet. [[RFC6480](#)]

In order to make participation in the RPKI easier, it is helpful to have a few consolidated repositories for RPKI objects, thus saving every participant from the cost of maintaining a new service. Similarly, relying parties using the RPKI objects will find it faster and more reliable to retrieve the necessary set from a smaller number of repositories.

These consolidated RPKI object repositories will in many cases be outside the administrative scope of the organization issuing a given RPKI object. Hence the need for a protocol to publish RPKI objects.

This document defines the RPKI publication protocol, including a sub-protocol for configuring the publication engine.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [[RFC2119](#)].

"Publication engine" and "publication server" are used interchangeably to refer to the server providing the service described in this document.

"Business Public Key Infrastructure" ("Business PKI" or "BPKI") refers to a PKI, separate from the RPKI, used to authenticate clients to the publication engine.

2. Context

This protocol was designed specifically for the case where an internet registry, already issuing RPKI certificates to its children, also wishes to run a publication service for its children.

We use the term "Business PKI" here because an internet registry might already have a PKI, separate from the RPKI, for authenticating its clients and might wish to reuse that PKI for this protocol. Such reuse is not a requirement.

3. Protocol Specification

In summary, the publication protocol uses XML messages wrapped in CMS, carried over HTTP transport.

The publication protocol consists of two separate subprotocols. The first is a control protocol used to configure a publication engine. The second subprotocol, which we refer to by the overloaded term "publication protocol", is used to request publication of specific objects. The publication engine operates a single HTTP server on a single port. It distinguishes between the two protocols by using different URLs for them.

3.1. Common Details

This section discusses details that the two subprotocols have in common, including the transport and CMS wrappers.

Both protocols use a simple request/response interaction. The client passes a request to the server, and the server generates a corresponding response.

A message exchange commences with the client initiating an HTTP POST with content type of "application/rpki-publication", with the message object as the body. The server's response will similarly be the body of the response with a content type of "application/rpki-publication".

The content of the POST and the server's response will be a well-formed Cryptographic Message Syntax (CMS) [[RFC5652](#)] object with OID = 1.2.840.113549.1.7.2 as described in [Section 3.1 of \[RFC6492\]](#).

3.1.1. Common XML Message Format

The XML schema for this protocol (including both subprotocols) is below in [Section 3.5](#). Both subprotocols use the same basic XML message format, which looks like:

```
<?xml version='1.0' encoding='us-ascii'?>
<msg xmlns="http://www.hactrn.net/uris/rpki/publication-spec/"
    version="2"
    type="message type">
  [one or more PDUs]
</msg>
```

version: The value of this attribute is the version of this protocol. This document describes version 2.

type: The possible values of this attribute are "reply" and "query".

A query PDU may be one of four types: `config_query`, `client_query`, `publish_query`, or `withdraw_query`. The first two are used by the control sub-protocol, the latter two by the publication sub-protocol.

A reply PDU may be one of five types: `config_reply`, `client_reply`, `publish_reply`, `withdraw_reply`, or `report_error_reply`.

Each of these PDUs may include an optional tag to facilitate bulk operation. If a tag is set in a query PDU, the corresponding reply(s) MUST have the tag attribute set to the same value.

3.2. Control Sub-Protocol

The control sub-protocol is used to configure a publication server. It can set global variables (at the moment, limited to a BPKI CRL) and manage clients who are allowed to publish data on the server.

3.2.1. Config Object

The `<config/>` object allows configuration of data that apply to the entire publication server rather than a particular client. There is exactly one `<config/>` object in the publication server, and it only supports the "set" and "get" actions -- it cannot be created or destroyed. Its use is typically restricted to the repository operator.

The `<config/>` object only has one data element that can be set: the `bpki_crl`. This is used by the publication server when authenticating clients.

3.2.2. Client Object

Unlike the `<config/>` object, the `<client/>` object represents one client authorized to use the publication server. There may be more than one `<client/>` object on each publication server. Again, its use is typically restricted to the repository operator.

The `<client/>` object supports five actions: "create", "set", "get", "list", and "destroy". Each client has a "client_handle" attribute, which is used in responses and must be specified in "create", "set", "get", or "destroy" actions. The "create" and "set" actions have an optional flag to clear CMS-timestamp-based replay protection, to allow recovery from misconfigured clocks.

Payload data which can be configured in a `<client/>` object include:

- o `base_uri` (attribute): This attribute represents the base URI below which the client will be allowed to publish data. Additional constraints may be imposed by the publication server in certain cases, for e.g., a child publishing directly under its parent.
- o `bpki_cert` (element): This represents the X.509 BPKI CA certificate for this client. This should be used as part of the certificate chain when validating incoming CMS messages. Two valid approaches exist. If the optional `bpki_glue` certificate is being used, then the `bpki_cert` certificate should be issued by the `bpki_glue` certificate; otherwise, the `bpki_cert` certificate should be issued by the publication engine's `bpki_ta` certificate.
- o `bpki_glue` (element): This is an additional (optional) type of X.509 certificate for this client. It may be used in certain pathological cross-certification cases which require a two-certificate chain due to issuer name conflicts. When being used, issuing order is that the `bpki_glue` certificate should be the issuer of the `bpki_cert` certificate. Otherwise, it should be issued by the publication engine's `bpki_ta` certificate. Since this is an optional use certificate, it may be left unset if not needed.

3.3. Publication Sub-Protocol

The publication sub-protocol requests publication or withdrawal from publication of RPKI objects.

The publication protocol uses a common message format to request publication of any RPKI object. This format was chosen specifically to allow this protocol to accommodate new types of RPKI objects without needing changes to this protocol.

Both the `<publish/>` and `<withdraw/>` objects have a payload of an optional tag and a URI. The `<publish/>` query also contains the DER object to be published, encoded in Base64.

Note that every publish and withdraw action requires a new manifest, thus every publish or withdraw action will involve at least two objects.

3.4. Error handling

Errors are handled similarly in both subprotocols, and they're handled at two levels.

Since all messages in this protocol are conveyed over HTTP connections, basic errors are indicated via the HTTP response code. 4xx and 5xx responses indicate that something bad happened. Errors that make it impossible to decode a query or encode a response are handled in this way.

Where possible, errors will result in an XML `<report_error/>` message which takes the place of the expected protocol response message. `<report_error/>` messages are CMS-signed XML messages like the rest of this protocol, and thus can be archived to provide an audit trail.

`<report_error/>` messages only appear in replies, never in queries. The `<report_error/>` message can appear in both the control and publication subprotocols.

Like all other messages in this protocol, the `<report_error/>` message includes a "tag" attribute to assist in matching the error with a particular query when using batching. It is optional to set the tag on queries but, if set on the query, it MUST be set on the reply or error.

The error itself is conveyed in the `error_code` (attribute). The value of this attribute is a token indicating the specific error that occurred.

The body of the `<report_error/>` element itself is an optional text string; if present, this is debugging information.

3.5. XML Schema

The following is a RelaxNG compact form schema describing the Publication Protocol.

```
# $Id: rpki-publication.rnc 2601 2013-10-18 19:21:28Z sra $  
# RelaxNG schema for RPKI publication protocol.
```

```
default namespace =  
    "http://www.hactrn.net/uris/rpki/publication-spec/"
```

```
# This is version 2 of the protocol.
```

```
version = "2"
```



```
# Top level PDU is either a query or a reply.

start = element msg {
  attribute version { version } ,
  ( ( attribute type { "query" }, query_elt* ) |
    ( attribute type { "reply" }, reply_elt* ) )
}

# PDUs allowed in a query.

query_elt |= config_query
query_elt |= client_query
query_elt |= publish_query
query_elt |= withdraw_query

# PDUs allowed in a reply.

reply_elt |= config_reply
reply_elt |= client_reply
reply_elt |= publish_reply
reply_elt |= withdraw_reply
reply_elt |= report_error_reply

# Tag attributes for bulk operations.

tag = attribute tag { xsd:token { maxLength="1024" } }

# Base64 encoded DER stuff.

base64 = xsd:base64Binary

# Publication URLs.

uri_t = xsd:anyURI { maxLength="4096" }
uri = attribute uri { uri_t }

# Handles on remote objects (replaces passing raw SQL IDs).

object_handle = xsd:string {
  maxLength = "255"
  pattern="[\-_A-Za-z0-9/]*"
}

# Error codes.

error = xsd:token { maxLength="1024" }

# <config/> element (use restricted to repository operator)
```



```
config_payload = (element bpki_crl { base64 }?)

config_query |= element config {
  attribute action { "set" },
  tag?,
  config_payload
}

config_reply |= element config {
  attribute action { "set" },
  tag?
}

config_query |= element config {
  attribute action { "get" },
  tag?
}

config_reply |= element config {
  attribute action { "get" },
  tag?,
  config_payload
}

# <client/> element (use restricted to repository operator)

client_handle = attribute client_handle { object_handle }

client_payload = (
  attribute base_uri { uri_t }?,
  element bpki_cert { base64 }?,
  element bpki_glue { base64 }?
)

client_clear_replay = (
  attribute clear_replay_protection { "yes" }?
)

client_query |= element client {
  attribute action { "create" },
  tag?,
  client_handle,
  client_clear_replay,
  client_payload
}

client_reply |= element client {
  attribute action { "create" },
```



```
    tag?,
    client_handle
}

client_query |= element client {
    attribute action { "set" },
    tag?,
    client_handle,
    client_clear_replay,
    client_payload
}

client_reply |= element client {
    attribute action { "set" },
    tag?,
    client_handle
}

client_query |= element client {
    attribute action { "get" },
    tag?,
    client_handle
}

client_reply |= element client {
    attribute action { "get" },
    tag?,
    client_handle,
    client_payload
}

client_query |= element client {
    attribute action { "list" },
    tag?
}

client_reply |= element client {
    attribute action { "list" },
    tag?,
    client_handle,
    client_payload
}

client_query |= element client {
    attribute action { "destroy" },
    tag?,
    client_handle
}
```



```
client_reply |= element client {
  attribute action { "destroy" },
  tag?,
  client_handle
}

# <publish/> element

publish_query |= element publish {
  tag?,
  uri,
  base64
}

publish_reply |= element publish {
  tag?,
  uri
}

# <withdraw/> element

withdraw_query |= element withdraw {
  tag?,
  uri
}

withdraw_reply |= element withdraw {
  tag?,
  uri
}

# <report_error/> element

report_error_reply = element report_error {
  tag?,
  attribute error_code { error },
  xsd:string { maxLength="512000" }?
}
```

4. Examples

Following are examples of various queries and the corresponding replies for the RPKI publication protocol

4.1. <config/> Set Query


```
<msg
  type="query"
  version="2"
  xmlns="http://www.hactrn.net/uris/rpki/publication-spec/">
  <config
    action="set">
    <bpci_crl>
      MIIBezBlAgEBMA0GCSqGSIB3DQEBCwUAMCMxITAFBgNVBAMTGFRlc3QgQ2Vy
      dGlmawNhdGUgcHVIZCBURcNMDgwNjAyMjE0OTQ1WhcNMDgwNzAyMjE0OTQ1
      WqA0MAwwCgYDVROUBAMCAQEwDQYJKoZIhvcNAQELBQADggEBAFWCWgBl4ljV
      qX/CHo+RpqYtvmKMnjPVf1MXUB7i28RGP4DAq4l7deDU7Q82xEJyE4TXMWDW
      AV6UG6uUGum0VHW0cj9ohqyiZUGf0sKg2hbwkETm8sAEN0si1yNdyKGk6jZ1
      6aF5fubxQqZa1pdGCSac1/ZYC5sLLhEz3kmz+B9z9mXFVc5TgAh4dN3Gy5ft
      F8zZAFpDgnS4biCnRVqhGv6R0Lh/5xmii+ZU6kNDhbeMsjJg+Z0mtN+wMeHS
      Ibjiy0WuuuZ3k2xSh0C94anrHBZAvvCRhbazjR0Ef50MZ5lcllw3u08IHuoi
      sHKkehy4Y0GySdj98fV+OuiRTH9vt/M=
    </bpci_crl>
  </config>
</msg>
```

[4.2.](#) **<config/> Set Reply**

```
<msg
  type="reply"
  version="2"
  xmlns="http://www.hactrn.net/uris/rpki/publication-spec/">
  <config
    action="set"/>
</msg>
```

[4.3.](#) **<config/> Get Query**

```
<msg
  type="query"
  version="2"
  xmlns="http://www.hactrn.net/uris/rpki/publication-spec/">
  <config
    action="get"/>
</msg>
```


4.4. <config/> Get Reply

```
<msg
  type="reply"
  version="2"
  xmlns="http://www.hactrn.net/uris/rpki/publication-spec/">
  <config
    action="get">
    <bpci_crl>
      MIIBezBlAgEBMA0GCSqGSIB3DQEBCwUAMCMxITAFBgNVBAMTGFRlc3QgQ2Vy
      dGlmawNhdGUgcHViZCZBUQRcNMDgwNjAyMjE0OTQ1WhcNMDgwNzAyMjE0OTQ1
      WQAOMAwWcGyDVR0UBAMCAQEwDQYJKoZIhvcNAQELBQADggEBAFWCWgBl4ljV
      qX/CHo+RpqYtvmKMnjPVf1MXUB7i28RGP4DAq4l7deDU7Q82xEJyE4TXMWDW
      AV6UG6uUGum0VHW0cj9ohqyiZUGf0sKg2hbwkETm8sAEN0si1yNdyKGk6jZ1
      6aF5fubxQqZa1pdGCSac1/ZYC5sLLhEz3kmz+B9z9mXFVc5TgAh4dN3Gy5ft
      F8zZAFpDGnS4biCnRVqhGv6R0Lh/5xmii+ZU6kNDhbeMsjJg+Z0mtN+wMeHS
      Ibjiy0Wuuaz3k2xSh0C94anrHBZAvvCRhbazjR0Ef50MZ5lcllw3u08IHuoi
      sHKkehy4Y0GySdj98fV+OuirTH9vt/M=
    </bpci_crl>
  </config>
</msg>
```

4.5. <client/> Create Query

```
<msg
  type="query"
  version="2"
  xmlns="http://www.hactrn.net/uris/rpki/publication-spec/">
  <client
    action="create"
    base_uri="rsync://wombat.invalid/"
    client_handle="3">
    <bpci_cert>
      MIIDGzCCAgOgAwIBAgIJAKi+/+wUhQ1xMA0GCSqGSIB3DQEBBQUAMCQxIjAg
      BgNVBAMTGVRlc3QgQ2VydgGlmawNhdGUgQm9iIFJvb3QwHhcNMDcwODAxMTk1
      MzEwWhcNMDcwODMxMTk1MzEwWjAkMSIwIAYDVQQDExlUZXR0IENlcnRpZmlj
      YXR1IEJvYiBSb290MIIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA
      rKYUtJaM5PH5917SG2ACc7iBYdQ02HYyu8Gb6i9Q2Gxc3cWEX7RTBvg0L79p
      Wf3GI dnoupzMN0ZVtY3GUx2G/0WkmLui2TCeDhcfXdQ4rcp8J3V/6ESj+yuE
      PPOG8UN17mUKKgujrCh6ZvgCD09AyOK/uXu+ABQXTPsn2pVe2EVh3V004ShL
      i8GKgVdqB/rW/6GTg0Xb/zLT6WwMuT++6sXTlztJdQYkRamJvKfQDU1naC8m
      AkGf79Tba0xyBGAUII0GfREY6t4/+NAP2Yyb3xNlBqcJoTov0JfNKHZcCZeP
      r79j7LK/hkZxxip+Na9xDpE+oQRV+DRukCRJdiqg+wIDAQABo1AwTjAMBgNV
      HRMEBTADAQH/MB0GA1UdDgQWBBDDESXJe6pjAQD4UL1B7+GMDBlimTAfBgNV
      HSMEGDAWgBTDESXJe6pjAQD4UL1B7+GMDBlimTANBgkqhkiG9w0BAQUFAAOCA
      QEAWWkNcW6S1tKKqtzJsdfhjJiAAPQmOXJskv0ta/8f6Acgcum1YieNdtT0
      n96P7CUH0WP8QBb91JzeewR7b6WJLwb10ffs3wNq3kk75pJe89r4XY39EZHH
    </bpci_cert>
  </client>
</msg>
```



```
MW+Dv0PhIKu2CgD4LeyH1FVTQkF/Q0bGEmkn+sHTsuzd1l2VLwcP1Smsqep
6LAlFj62qqaIJzNeQ9NVkBqtkygnYlB0kaBTHfQTux3jYNpEo8JJB5e/WFdH
YyMNRG2xM0tIC7T4+IOHgT8PgrNhaeDg9ctewj0X8Qi9nI9nXeinicLX8vj6
hdEq30Rv7RZMJNYqv1HQ3wUE2B7fCPFv7EUwzaCds1kgRQ==
</bpki_cert>
</client>
</msg>
```

[4.6.](#) **<client/> Create Reply**

```
<msg
  type="reply"
  version="2"
  xmlns="http://www.hactrn.net/uris/rpki/publication-spec/">
  <client
    action="create"
    client_handle="3"/>
</msg>
```

[4.7.](#) **<client/> Set Query**

```
<msg
  type="query"
  version="2"
  xmlns="http://www.hactrn.net/uris/rpki/publication-spec/">
  <client
    action="set"
    client_handle="3">
    <bpki_cert>
      MIIDGzCCAgoAgAwIBAgIJAKi+/+wUhQ1xMA0GCSqGSIb3DQEBBQUAMCQxIjAg
      BgNVBAMTGVRlc3QgQ2VydgGlmawNhdGUGuQm9iIFJvb3QwHhcNMDcwODAxMTk1
      MzEwWWhcNMDcwODMxMTk1MzEwWjAKMSIwIAYDVQQDExlUZXR0IENlc3RlZm1j
      YXR1IEJvYiBSb290MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA
      rKYUtJaM5PH5917SG2ACc7iBYdQ02HYyu8Gb6i9Q2Gxc3cWEX7RTBvg0L79p
      Wf3GI dnoupzMnoZVtY3GUx2G/0WkmLui2TCeDhcfXdQ4rcp8J3V/6ESj+yuE
      PPOG8UN17mUKKgujrch6ZvgCD09Ay0K/uXu+ABQXTPsn2pVe2EVh3V004ShL
      i8GKgVdqB/rW/6GTg0Xb/zLT6WwMuT++6sXTlztJdQYkRamJvKfQDU1naC8m
      AkGf79Tba0xyBGAU1I0GfREY6t4/+NAP2Yyb3xNlBqcJoTov0JfNKHZcCZeP
      r79j7LK/hkZxxip+Na9xDpE+oQRV+DRukCRJdiqg+wIDAQABo1AwTjAMBgNV
      HRMEBTADAQH/MB0GA1UdDgQWBBDDESXJe6pjAQD4UL1B7+GMDBlimTAfBgNV
      HSMGDAwGBTDESXJe6pjAQD4UL1B7+GMDBlimTANBgkqhkiG9w0BAQUFAAOCA
      AQEAWwKNcW6S1tKKqtzJsdfhjJiAAPQmOXJskv0ta/8f6Acgcum1YieNdtT0
      n96P7CUH0WP8QBb91JzeewR7b6WJLwb10ffs3wNq3kk75pJe89r4XY39EZHH
      MW+Dv0PhIKu2CgD4LeyH1FVTQkF/Q0bGEmkn+sHTsuzd1l2VLwcP1Smsqep
      6LAlFj62qqaIJzNeQ9NVkBqtkygnYlB0kaBTHfQTux3jYNpEo8JJB5e/WFdH
      YyMNRG2xM0tIC7T4+IOHgT8PgrNhaeDg9ctewj0X8Qi9nI9nXeinicLX8vj6
    </bpki_cert>
  </client>
</msg>
```



```

      hdEq30RV7RZMJNYqv1HQ3wUE2B7fCPFv7EUwzaCds1kgRQ==
    </bpki_cert>
  </client>
</msg>

```

4.8. `<client/>` Set Reply

```
<msg
  type="reply"
  version="2"
  xmlns="http://www.hactrn.net/uris/rpki/publication-spec/">
  <client
    action="set"
    client_handle="3"/>
</msg>
```

4.9. <client/> Get Query

```
<msg
  type="query"
  version="2"
  xmlns="http://www.hactrn.net/uris/rpki/publication-spec/">
  <client
    action="get"
    client_handle="3"/>
</msg>
```

4.10. <client/> Get Reply

```
<msg
  type="reply"
  version="2"
  xmlns="http://www.hactrn.net/uris/rpki/publication-spec/">
  <client
    action="get"
    base_uri="rsync://wombat.invalid/"
    client_handle="3">
    <bpki_cert>
      MIIDGzCCAgOgAwIBAgIJAKi+//wUhQ1xMA0GCSqGSib3DQEBBQUAMCQxIjAg
      BgNVBAMTGVRlc3QgQ2VydGlmawNhdGUgQm9iIFJvb3QwHhcNMMDcwODAxMTk1
      MzEwWhcNMMDcwODMxMTk1MzEwWjAKMSiWIAYDVQQDEXLUZXN0IENlcuRpZmlj
      YXRlIEJvYiBSb290MIIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA
      rKYUtJaM5PH5917SG2ACC7iBYdQ02HYyu8Gb6i9Q2Gxc3cWEX7RTBVg0L79p
      Wf3GI dnoupzMnoZVtY3GUx2G/0WkmLui2TCeDhcfXdQ4rcp8J3V/6ESj+yuE
      PPOG8UN17mUKKqujrch6ZvqCD09Ay0K/uXu+ABQXTPsn2pVe2EVh3V004ShL
```



```

i8GKgVdqB/rW/6GTg0Xb/zLT6WwMuT++6sXTlztJdQYkRamJvKfQDU1naC8m
AkGf79Tba0xyBGAUII0GfREY6t4/+NAP2Yyb3xNlBqcJoTov0JfNKHZcCZeP
r79j7LK/hkZxxip+Na9xDpE+oQRV+DRukCRJdiqg+wIDAQAB01AwTjAMBgNV
HRMEBTADAQH/MB0GA1UdDgQWBbTDEsXJe6pjAQD4ULlB7+GMDBlimTAfBgNV
HSMEGDAWgBTDEsXJe6pjAQD4ULlB7+GMDBlimTANBgkqhkiG9w0BAQUFAAOC
AQEAWwKncW6S1tKKqtzJsdfhjJiAAPQm0XJskv0ta/8f6Acgcum1YieNdtT0
n96P7CUH0WP8QBb91JzeewR7b6WJLwb10ffs3wNq3kk75pJe89r4XY39EZHH
MW+Dv0PhIKu2CgD4LeyH1FVTQkF/Q0bGEmkn+s+HTsuzd1l2VLwcP1Smsqep
6LAlFj62qqaIJzNeQ9NVkBqtkygnYlB0kaBTHfQTux3jYNpEo8JJB5e/WFdH
YyMNRg2xM0tIC7T4+IOHgT8PgrNhaeDg9ctewj0X8Qi9nI9nXeinicLX8vj6
hdEq30Rv7RZMJNYqv1HQ3wUE2B7fCPFv7EUwzaCds1kgRQ==
</bpki_cert>
</client>
</msg>

```

[4.11.](#) **<client/> List Query**

```

<msg
  type="query"
  version="2"
  xmlns="http://www.hactrn.net/uris/rpki/publication-spec/">
  <client
    action="list"/>
</msg>

```

[4.12.](#) **<client/> List Reply**

```

<msg
  type="reply"
  version="2"
  xmlns="http://www.hactrn.net/uris/rpki/publication-spec/">
  <client
    action="list"
    client_handle="3">
    <bpki_cert>
      MIIDGzCCAgOgAwIBAgIJAKi+//wUhQlXMA0GCSqGSIb3DQEBBQUAMCQXijAg
      BgNVBAMTGVRlc3QgQ2VydGlmawNhdGUgQm9iIFJvb3QwHhcNMDCwODAxMTk1
      MzEwWhcNMDCwODMxMTk1MzEwWjAKMSIwIAYDVQQDExlUZXRN0IENlcnRpZmlj
      YXRlIEJvYiBSb290MIIBIjANBgkqhkiG9w0BAQEFAAOCQA8AMIIBCgKCAQEa
      rKYUtJaM5PH5917SG2ACc7iBYdQ02HYyu8Gb6i9Q2Gxc3cWEX7RTBvgOL79p
      Wf3GI dnoupzMnoZVtY3GUx2G/0WkmLui2TCeDhcfXdQ4rcp8J3V/6ESj+yuE
      PPOG8UN17mUKKgujrch6ZvgCD09AyOK/uXu+ABQXTPsn2pVe2EVh3V004ShL
      i8GKgVdqB/rW/6GTg0Xb/zLT6WwMuT++6sXTlztJdQYkRamJvKfQDU1naC8m
      AkGf79Tba0xyBGAUII0GfREY6t4/+NAP2Yyb3xNlBqcJoTov0JfNKHZcCZeP
      r79j7LK/hkZxxip+Na9xDpE+oQRV+DRukCRJdiqg+wIDAQAB01AwTjAMBgNV
      HRMEBTADAQH/MB0GA1UdDgQWBbTDEsXJe6pjAQD4ULlB7+GMDBlimTAfBgNV
    </bpki_cert>
  </client>
</msg>

```



```
HSMEGDAWgBTDEsXJe6pjAQD4ULlB7+GMDBlmTANBgkqhkiG9w0BAQUFAAOC
AQEAWWkNcW6S1tKKqtzJsdfhjJiAAPQmOXJskv0ta/8f6Acgcum1YieNdtT0
n96P7CUH0WP8QBb91JzeewR7b6WJLwb10ffs3wNq3kk75pJe89r4XY39EZHH
MW+Dv0PhIKu2CgD4LeyH1FVTQkF/Q0bGEmkn+s+HTsuzd1l2VLwcP1Smsqep
6LAlFj62qqaIJzNeQ9NVkBqtkygnYlB0kaBTHfQTux3jYNpEo8JJB5e/WFdH
YyMNRG2xM0tIC7T4+IOHgT8PgrNhaeDg9ctewj0X8Qi9nI9nXeinicLX8vj6
hdEq30Rv7RZMJNYqv1HQ3wUE2B7fCPFv7EUwzaCds1kgRQ==
</bpki_cert>
</client>
</msg>
```

[4.13.](#) **<client/> Destroy Query**

```
<msg
  type="query"
  version="2"
  xmlns="http://www.hacitrn.net/uris/rpki/publication-spec/">
  <client
    action="destroy"
    client_handle="3"/>
</msg>
```

[4.14.](#) **<client/> Destroy Reply**

```
<msg
  type="reply"
  version="2"
  xmlns="http://www.hacitrn.net/uris/rpki/publication-spec/">
  <client
    action="destroy"
    client_handle="3"/>
</msg>
```

[4.15.](#) **<publish/> Query**

```
<msg
  type="query"
  version="2"
  xmlns="http://www.hacitrn.net/uris/rpki/publication-spec/">
  <publish
    uri="rsync://wombat.invalid/Alice/blCrcCp9ltyPDNzYKPfxc.cer">
    MIIe+jCCA+KgAwIBAgIBDTANBgkqhkiG9w0BAQsFADAzMTEwLWYDVQQDEyHE
    RjRBODAXN0U2NkE5RTkxNzJFNDYxMkQ4Q0Y0QzgzRjIzOERFMkEzMB4XDTA4
    MDUyMjE4MDUxM1oXDTA4MDUyNDE3NTQ1M1owMzExMC8GA1UEAxMoOEZC0DIX
    OEYwNkU1MEFCNzAyQTdEOTZEQzhGMENEQ0Q4MjhGN0YxNzCCASIwDQYJKoZI
```



```

hvcNAQEBBQADggEPADCCAQoCggEBAMeziKp0k5nP7v6SZoNsXIMQYRgNtC6F
r/9Xm/1yQHomiPqHUK47rHhGojYiK5AhkrwoYhkh4UjJl2iwlDYczXuaBU3
F5qrKlZ4aZnjIxdlP7+hktVpeApL6yuJTUAYeC3UIxnLDVdD6phydZ/F0Qlu
ffiNDjzteCCvoyOUatqt8WB+oND6LToHp028g1YUYLHG6mur0dPdcHOVXLsm
UDuZ1HDz1nDuYvIVKjB/MpH9aw9XeaQ6ZFIlZVPwuuvI2brR+ThH7Gv27GL/
o8qFdC300VQfoTZ+rKPGDE8K1cI906BL4kiwx9z0oiDcE96QCz+B0vsjc9mG
aA1jgAxlXwsCAwEAAa0CAhcwggITMB0GA1UdDgQWBBSpuCGPBuUKtwKn2W3I
8M3Ngo9/FzAfBgNVHSMEGDAWgBTfSoAX5mqekXLkYS2M9Mg/I43iozBVBgNV
HR8ETjBMMEqgSKBGhkRyc3luYzovL2xvY2FsaG9zdDo0NDAwL3Rlc3RiZWQv
UklSLzEvMzBxQUYtWnFucEZ5NUdFdGpQVElQeU90NHFNLMNybDBFBggrBgEF
BQcBAQQ5MDcwNQYIKwYBBQUHMAKGKXJzeW5j0i8vbG9jYWxob3N0OjQ0MDAv
dGVzdGJlZC9XT01CQVQuY2VyMBGA1UdIAEB/wQOMAwWCgYIKwYBBQUHdGJw
DwYDVR0TAQH/BAUwAwEB/zA0BgNVHQ8BAf8EBAMCAQYwgZsGCCsGAQUFBwEL
BIGOMIGLMDQGCCsGAQUFBzAFhahyc3luYzovL2xvY2FsaG9zdDo0NDAwL3Rl
c3RiZWQvUklSL1IwLzEvMFMGCCsGAQUFBzAKhkdyd3luYzovL2xvY2FsaG9z
dDo0NDAwL3Rlc3RiZWQvUklSL1IwLzEvajdnaGp3YmxDcmNDcDlzdHlQRE56
WUtQZnhjLm1uZjAaBggrBgEFBQcBCAEB/wQLMAmgBzAFAGMA/BUwPgYIKwYB
BQUHAQcBAf8ELzAtMCsEAgABMCDUAWAKAZA0AwUAwAACAMFACAAAiAwDgMF
AsAAAiWDBQDAAAJKMA0GCSqGSIB3DQEBcUAA4IBAQCehU7jtI2PJY6+zWv
306vmCuXhtu9Lr2mmRw2ZErB8EMcb5xypMrNqMoKou14K2x4a4RPJkK4yAth
M81FPNRsU5mM0acIRnAPtxjHvPME7PHN2w2nGLASRsZmaa+b8A7SS0xVcFUR
azENZtpps0lHeTpm0cpLiTK7mNpudUg1JGuFo94VLf1MnE2EqARG1vTsNhe1
/SM/Uv0ARCC0Bvf0Gz7kSuupDSZ7qx+LiDmtEsLdbGNQBiyPbLrDk41Phrxd
x28qIj7ejZkRzNFw/3pi8/XK281h8zeHoFVu6ghRPY5db0A4akX/KG6b8XIX
0iwPYdLiDbdWFbtTdPcXBauY
</publish>
</msg>

```

4.16. <publish/> Reply

```

<msg
  type="reply"
  version="2"
  xmlns="http://www.hactrn.net/uris/rpki/publication-spec/">
  <publish
    uri="rsync://wombat.invalid/Alice/blCrcCp9ltyPDNzYKPfxc.cer"/>
  </msg>

```

4.17. <withdraw/> Query

```

<msg
  type="query"
  version="2"
  xmlns="http://www.hactrn.net/uris/rpki/publication-spec/">
  <withdraw
    uri="rsync://wombat.invalid/Alice/blCrcCp9ltyPDNzYKPfxc.cer"/>
  </withdraw>
</msg>

```



```
</msg>
```

[4.18.](#) `<withdraw/>` Reply

```
<msg
  type="reply"
  version="2"
  xmlns="http://www.hactrn.net/uris/rpki/publication-spec/">
  <withdraw
    uri="rsync://wombat.invalid/Alice/blCrcCp9ltyPDNzYKPfxc.cer"/>
</msg>
```

[4.19.](#) `<report_error/>` With Text

```
<msg
  type="reply"
  version="2"
  xmlns="http://www.hactrn.net/uris/rpki/publication-spec/">
  <report_error
    error_code="your_hair_is_on_fire">
    Shampooing with sterno again, are we?
  </report_error>
</msg>
```

[4.20.](#) `<report_error/>` Without Text

```
<msg
  type="reply"
  version="2"
  xmlns="http://www.hactrn.net/uris/rpki/publication-spec/">
  <report_error
    error_code="your_hair_is_on_fire"/>
</msg>
```

[5.](#) Operational Considerations

There are two basic options open to the repository operator as to how the publication tree is laid out. The first option is simple: each publication client is given its own directory one level below the top of the rcynic module, and there is no overlap between the publication spaces used by different clients. For example:

```
rsync://example.org/rpki/Alice/
rsync://example.org/rpki/Bob/
```



```
rsync://example.org/rpki/Carol/
```

This has the advantage of being very easy for the publication operator to manage, but has the drawback of making it difficult for relying parties to fetch published objects both safely and as efficiently as possible.

Given that the mandatory-to-implement retrieval protocol for relying parties is rsync, a more efficient repository structure would be one which minimized the number of rsync fetches required. One such structure would be one in which the publication directories for subjects were placed underneath the publication directories of their issuers: since the normal synchronization tree walk is top-down, this can significantly reduce the total number of rsync connections required to synchronize. For example:

```
rsync://example.org/rpki/Alice/  
rsync://example.org/rpki/Alice/Bob/  
rsync://example.org/rpki/Alice/Bob/Carol/
```

Preliminary measurement suggests that, in the case of large numbers of small publication directories, the time needed to set up and tear down individual rsync connections becomes significant, and that a properly optimized tree structure can reduce synchronization time by an order of magnitude.

The more complex tree structure does require careful attention to the `base_uri` attribute values when setting up clients. In the example above, assuming that Alice issues to Bob who in turn issues to Carol, Alice has ceded control of a portion of her publication space to Bob, who has in turn ceded a portion of that to Carol, and the `base_uri` attributes in the `<client/>` setup messages should reflect this.

The details of how the repository operator determines that Alice has given Bob permission to nest Bob's publication directory under Alice's is outside the scope of this protocol.

6. IANA Considerations

IANA is asked to register the `application/rpki-publication` MIME media type as follows:

```
MIME media type name:  application  
MIME subtype name:    rpki-publication  
Required parameters:  None  
Optional parameters:  None  
Encoding considerations:  binary  
Security considerations:  Carries an RPKI Publication Protocol
```


Message, as defined in this document.
Interoperability considerations: None
Published specification: This document
Applications which use this media type: HTTP
Additional information:
 Magic number(s): None
 File extension(s):
 Macintosh File Type Code(s):
Person & email address to contact for further information:
 Rob Austein <sra@hacitrn.net>
Intended usage: COMMON
Author/Change controller: Rob Austein <sra@hacitrn.net>

7. Security Considerations

The RPKI publication protocol and the data it publishes use entirely separate PKIs for authentication. The published data is authenticated within the RPKI, and this protocol has nothing to do with that authentication, nor does it require that the published objects be valid in the RPKI. The publication protocol uses a separate Business PKI (BPKI) to authenticate its messages.

Each of the RPKI publication protocol messages is CMS-signed. Because of that protection at the application layer, this protocol does not require the use of HTTPS or other transport security mechanisms.

Compromise of a publication server, perhaps through mismanagement of BPKI keys, could lead to a denial-of-service attack on the RPKI. An attacker gaining access to BPKI keys could use this protocol delete (withdraw) RPKI objects, leading to routing changes or failures. Accordingly, as in most PKIs, good key management practices are important.

8. References

8.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [RFC 2119](#), [BCP 14](#), March 1997.
- [RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", [RFC 5652](#), STD 70, September 2009.
- [RFC6492] Huston, G., Loomans, R., Ellacott, B., and R. Austein, "A Protocol for Provisioning Resource Certificates", [RFC 6492](#), February 2012.

8.2. Informative References

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

Authors' Addresses

Samuel Weiler
SPARTA, Inc.
7110 Samuel Morse Drive
Columbia, Maryland 21046
US

Email: weiler@tislabs.com

Anuja Sonalker
Battelle Memorial Institute

Email: sonalkera@battelle.org

Rob Austein
Dragon Research Labs

Email: sra@hacitrn.net

