PKIX Working Group	M. Pala	
Internet-Draft	Dartmouth College	
Intended status: Experimental	July 01, 2008	
Expires: January 2, 2009		

TOC

PKI Resource Query Protocol (PRQP)
draft-pala-prqp-02

Status of this Memo

Abstract

One of the most strategic problems still open in PKIX is locating public data and services associated with a Certification Authority (CA). This issue impacts interoperability and usability in PKIX.

This draft describes the PKI Resource Query Protocol (PRQP), its design, definition, and its impact in already deployed PKIX protocols.

Table of Contents

---

1. Requirements notation

TOC

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] (Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," March 1997.).

---

2. Introduction

TOC

An increasing number of services and protocols are being defined to address different needs of users and administrators of PKIs. With the

deployment of new applications and services, the need to access information and services provided by Certificate Service Providers (CSPs) is critical. Currently Certification Authorities (CAs) barely publish access details on their official web sites, this includes URL of provided services and repositories.

Using the PRQP, resources provided by a CA can be automatically and securely discovered by an application.

2.1. Overview of existing solutions

TOC

Currently there are three options to find URLs providing access to PKI data:

- by including such data in certificate extensions
- by searching easily accessible repositories (e.g. DNS, local database, etc.)
- by adapting existing protocols (e.g. SLP)

2.1.1. Certificate Extensions

TOC

To provide pointers to published data it is possible to use the Authority Information Access (AIA) Subject Information Access (SIA) extensions defined by PKIX [RFC3280] (Housley, R., Polk, W., Ford, W., and D. Solo, “Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile,” April 2002.).

The former can provide information about services associated with the issuer of the certificate, while the latter carries information (inside a CA certificate) about offered CA services.

AIA and SIA extensions are static, i.e. not modifiable unless the certificate is re-issued. If a CA inserts the AIA extension into every certificate it issues, e.g., to identify the location of an OCSP responder, then changing that location would require re-issuance of all these certificates, a substantial barrier to such a change. If a CA certificate is self-signed and used as a trust anchor, then re-issuing the certificate to change the content of the SIA extension, e.g., to reflect a change in the location of a time stamping server would be very disruptive. In closed PKIs, e.g., enterprises, use of these extensions may be replaced by manual configuration and management of

this data via ad hoc means. Because of the centrally controlled nature of such environments, the static nature of SIA and AIA extensions is not a concern.

However in order to promote interoperability between PKIs, PRQP enables dynamic management of pointers to such services (e.g., adding/removing or moving) without requiring changes in the certificate contents or third parties to manually configure services in their applications. Even in closed environments, PRQP could help manage PKI services analogous the way DHCP facilitates network management.

2.1.2. DNS SRV records

TOC

The SRV record technique provides pointers to servers via the DNS [RFC1035] (Mockapetris, P., “Domain names - implementation and specification,” November 1987.).

As defined in [RFC2782] (Gulbrandsen, A., Vixie, P., and L. Esibov, “A DNS RR for specifying the location of services (DNS SRV),” February 2000.), the introduction of this type of record allows administrators to perform operations similar to what we require in order to solve the problem we are addressing in this draft, i.e., to provide URLs to services.

The problem in the adoption of this mechanism is that, in contrast to the DNS environment, usually in PKIX there is no fixed mapping between certificates and the DNS name space. The only exception is when the Domain Component (DC) attributes are used in the certificate's Subject.

Currently this approach is not widely adopted. Moreover, it is not always easy to identify the right DNS to query to, when trying to find a particular service provided by a CA, because of the lack of such information in certificates.

2.1.3. Local Network Oriented Solutions

TOC

Another approach to provide reliable information is to use existing protocols for service location such as Jini, Universal Plug and Play protocol (UPnP) or Service Location Protocol (SLP) [RFC2608] (Guttman, E., Perkins, C., Veizades, J., and M. Day, “Service Location Protocol, Version 2,” June 1999.) [RFC2609] (Guttman, E., Perkins, C., and J. Kempf, “Service Templates and Service: Schemes,” June 1999.).

The IETF defined the SLP to provide a service location mechanism that is language and technology independent. Some issues, however, make it not the right choice to solve our problem, e.g., the protocol is quite complex to implement when considering the scope of the problem we are addressing.

The definition of a specific and simple protocol for PKI service and resource location is needed to ease PKI integration into existing and future applications, especially for mobile devices which have limited computational power and communication bandwidth.

---

3. Protocol Details

TOC

The PRQP protocol is a request-response protocol, formed by the exchanging of two messages, i.e., a request and a response between a client and a server, called the Resource Query Authority (RQA).

The requesting entity (the client) may be any entity that needs to access information about repositories and services related to a certificate.

The RQA is the authority entitled to answer for a particular CA or to act as a PRQP Trusted Authority (PTA) for a set of users, e.g., users in an enterprise environment.

In the first case the RQA is directly designated by a CA to act as an RQA, by having the CA issue a certificate to the RQA with a specific value set in the extendedKeyUsage extension. In this case the RQA provides authoritative responses for requests regarding the CA that issued the RQA's certificate.

When operating as a PTA, the RQA may provide responses about multiple CAs, without the need to have been directly certified by them. To operate as such, a specific extension (prqpTrustedAuthority) should be present in RQA's certificate and its value should be set to TRUE.

3.1. The Resource Query Authority (RQA)

TOC

The Resource Query Authority is the designated authority to act as PRQP responder. The RQA's signing key needs not to be the same as that of the CA that designated it.

The CA may designate an RQA by issuing a certificate containing a unique value for the extendedKeyUsage in RQA's certificate. The RQA may

also act as a trusted responder. PRQP signing delegation SHALL be designated by the inclusion of id-kp-PRQPSigning in the extendedKeyUsage extension within the PRQP response signer's certificate.

```
id-kp-PRQPSigning OBJECT IDENTIFIER ::= {id-kp 10}
```

When operating as a PTA, the RQA may provide responses about multiple CAs, without the need to have been directly certified by them. To operate as a PTA a specific extension (prqpTrustedAuthority) should be present in RQA's certificate and its value should be set to TRUE.

```
prqpTrustedAuthority ::= BOOLEAN DEFAULT TRUE
```

We also define two new OIDs to identify the PRQP protocol and the PTA extension as follows:

```
id-prqp OBJECT IDENTIFIER ::= { id-pkix 23 }

id-prqp-pta OBJECT IDENTIFIER ::= { id-prqp 1 }
```

3.2. PRQP Overview

TOC

The protocol encompasses the exchange of a single round of messages between a client and an RQA:

1. the client requests a resource token by sending a request to the RQA

2. the RQA replies by sending a response to the client

Upon receiving the response the client MUST verify the status error returned in the response. If no error is present, the client MUST verify the various fields contained in the ResourceResponseToken and the validity of the associated digital signature (if present). A nonce MAY be used to guarantee that the response is associated with a specific request in order to avoid reply attacks.

The client also SHOULD check the validity period of the response. It SHOULD NOT, in order to minimize the load on an RQA, request again the location of the same resource within this interval to the same RQA.

If the response is signed, the client SHOULD check the RQA's certificate validity.

3.2.1. PRQP Request

TOC

A PRQP request contains the following data:

- *protocol version
- *nonce
- *MaxResponse
- *ResourceRequestToken
- *Extensions

The ASN.1 syntax imports terms defined in [RFC4210] (Adams, C., Farrell, S., Kause, T., and T. Mononen, “Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP),” September 2005.). For signature calculation, the data to be signed is encoded by using the DER format. ASN.1 EXPLICIT tagging is used as a default unless specified otherwise. The terms imported from [RFC3280] (Housley, R., Polk, W., Ford, W., and D. Solo, “Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile,” April 2002.) are: Extensions, Certificate, CertificateSerialNumber, SubjectPublicKeyInfo, Name, AlgorithmIdentifier.

3.2.1.1. Request Syntax

TOC

The PRQP request syntax is as follows:

```
PRQPRequest ::= SEQUENCE {
    requestData            TBSReqData,
    signature              [0] EXPLICIT Signature OPTIONAL }

TBSReqData ::= SEQUENCE {
    version                INTEGER { v(1) },
    nonce                  INTEGER              OPTIONAL,
          -- very large number
    maxRespEntries         INTEGER              OPTIONAL,
          -- maximum number of accepted entries in
          -- corresponding response
    serviceToken           ResourceRequestToken,
          -- token identifying the requested service
    extensions         [0] IMPLICIT Extensions  OPTIONAL }
```

The version field (currently v1) describes the version of the PRQP request. The nonce field, if present, is an integer between 80 bits and 256 bit in length.

The MaxResponse identifier is used to tell the RQA the maximum number of ResourceResponseToken that presenting can include in the response.

The ResourceRequestToken is used to identify the requested services. It carries information about the requested services. It contains a CA identifier and optionally one or more service identifiers.

```
ResourceRequestToken ::= SEQUENCE {
  ca                     CertIdentifier,
  servicesList        [0] SET OF ResourceIdentifier OPTIONAL }
```

The ca field is of type CertIdentifier. This is used to identify the certificate of the CA whose services are requested.

The CertIdentifier syntax is as follows:

```
BasicCertIdentifier ::= SEQUENCE {
  issuerNameHash              OCTET STRING,
  serialNumber                CertificateSerialNumber  }

ExtenderCertInfo ::= SEQUENCE {
  certificateHash             OCTET STRING,
  subjectKeyHash              OCTET STRING,
  subjectKeyIdentifier    [0] KeyIdentifier          OPTIONAL,
  issuerKeyIdentifier     [1] KeyIdentifier          OPTIONAL  }

CertIdentifier ::= SEQUENCE {
  hashAlgorithm               AlgorithmIdentifier,
  basicCertIdentifier         BasicCertIdentifier,
  extInfo                     [0] ExtendedCertInfo    OPTIONAL,
  caCertificate               [1] Certificate         OPTIONAL,
  issuedCertificate           [2] Certificate         OPTIONAL }
```

The resourceList specifies the resources or services being requested.

```
ResourceIdentifier ::= SEQUENCE {
  resourceId             OBJECT IDENTIFIER,
  version            [0] INTEGER             OPTIONAL
    --- version of the protocol or data format (if applicable) }
```

The ResourceIdentifier is formed by an OID that identifies the service or the data being requested (e.g. OCSP, LDAP, CRL, etc...) and an optional version number that may be used to better identify the requested resource. All fields SHOULD be used whenever applicable.

If one or more ResourceIdentifier are provided in the request, the RQA

should report back the location for each of the requested services. If no ResourceIdentifier is present in the request, the response should carry all the available service locations for the specified CA (with respect to the MaxResponse and optional parameters constrain).

The signature field is of type Signature and it is defined in [RFC2560] (Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP," June 1999.):

```
           Signature ::= SEQUENCE {
             signatureAlgorithm     AlgorithmIdentifier,
             signature              BIT STRING,
             certs              [0] EXPLICIT SEQUENCE OF Certificate
                                                          OPTIONAL }
```

Extensions can be used for future protocol enhancement.

3.2.2. PRQP Response

TOC

The PRQP response contains the following data:

- *protocol version
- *nonce
- *status
- *CA identifier
- *ResourceResponseToken
- *Extensions

3.2.2.1. Response Syntax

TOC

The response syntax is as follows:

```
PRQPResponse ::= SEQUENCE {
  respData               TBSRespData,
  signature          [0] EXPLICIT Signature OPTIONAL }

TBSRespData ::= SEQUENCE {
  version                INTEGER { v(1)},
  nonce                  INTEGER              OPTIONAL,
        -- as duplicated from the request
  producedAt             GeneralizedTime,
        -- time when the response has been generated
  nextUpdate         [0] GeneralizedTime      OPTIONAL,
        -- time till when the response should be considered valid
  pkiStatus              PKIStatusInfo,
        -- status of the response
  caCertId               CertIdentifier,
        -- identifier of the CA certificate that issued the
        -- targeted certificate
  responseToken          SEQUENCE OF ResourceResponseToken
                                                        OPTIONAL,
        -- token carrying informations about
        -- requested services
  extensions         [0] EXPLICIT Extensions  OPTIONAL }
```

The version field (currently v1) describes the version of the used PRQP response. The nonce, if present, binds the response to a specific request. The usage of the nonce is meaningful only in signed responses and its value must be copied directly from the corresponding request. If not present in the request, the nonce MUST be omitted.

The pkiStatus field is based on the definition of status in section 3.2.3 of [RFC4210] (Adams, C., Farrell, S., Kause, T., and T. Mononen, “Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP),” September 2005.). However, to limit the complexity of the field, the statusString field is of type UTF8String instead of PKIFreeText.

```
PKIStatusInfo ::= SEQUENCE {
  status        PKIStatus,
  statusString  [0] UTF8String     OPTIONAL,
  failInfo      [1] PKIFailureInfo  OPTIONAL  }
```

If status has value zero, a responseToken MUST be present in the response. When the status value is non zero, the responseToken MUST be omitted and the reason code MUST be one of the values in PKIStatus.

```
PKIStatus ::= INTEGER {
  ok                    (0),
     -- when the PKIStatus contains the value zero one or
        more responseToken is present
  badRequest            (1),
     -- the request is badly formatted
  caNotPresent          (2),
     -- the requested CA is not present
  systemFailure         (3)
     -- a system failure has occourred }
```

The signature field is of type Signature and it is defined in [RFC2560] (Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP," June 1999.):

```
Signature ::= SEQUENCE {
  signatureAlgorithm     AlgorithmIdentifier,
  signature              BIT STRING,
  certs              [0] EXPLICIT SEQUENCE OF Certificate
                                                 OPTIONAL }
```

The responseToken carries information about the services requested by the client. For each of the requested service, the RQA should include a ResourceResponseToken which bears the OID of the service and the corresponding URI.

The ResourceResponseToken syntax is described below:

```
ResourceInfo ::= SEQUENCE {
  resourceUri            IA5String,
      --- resource locator
  version            [0] INTEGER             OPTIONAL,
      --- version of the protocol or data format (if applicable)
  }

ResourceResponseToken ::= SEQUENCE {
  serviceId              OBJECT IDENTIFIER,
  resourceLocator    [0] EXPLICIT SEQUENCE OF ResourceInfo    }
```

The serviceId field value is copied from the corresponding request and it bears the OID of the service about which the client inquired. We define the following OIDs that SHOULD be used to identify the specified PKI services:

```
id-ad-prqp                    OBJECT IDENTIFIER ::= {id-ad 12 }
id-ad-prqp-ocsp               OBJECT IDENTIFIER ::= {id-ad-prqp 1}
id-ad-prqp-caIssuers          OBJECT IDENTIFIER ::= {id-ad-prqp 2}
id-ad-prqp-timestamping       OBJECT IDENTIFIER ::= {id-ad-prqp 3}
id-ad-prqp-dvcs               OBJECT IDENTIFIER ::= {id-ad-prqp 4}
id-ad-prqp-caRepository       OBJECT IDENTIFIER ::= {id-ad-prqp 5}
id-ad-prqp-http-certs         OBJECT IDENTIFIER ::= {id-ad-prqp 6}
        --- HTTP certificate repository
id-ad-prqp-http-crls          OBJECT IDENTIFIER ::= {id-ad-prqp 7}
        --- HTTP CRL download URL

id-ad-prqp-xkmsGateway        OBJECT IDENTIFIER ::= {id-ad-prqp 10}
        --- XKMS Gateway
id-ad-prqp-cmsGateway         OBJECT IDENTIFIER ::= {id-ad-prqp 11}
        --- CMS Gateway
id-ad-prqp-scepGateway        OBJECT IDENTIFIER ::= {id-ad-prqp 12}
        --- SCEP Gateway

--- Certificate Policies
id-ad-prqp-certPolicy         OBJECT IDENTIFIER ::= {id-ad-prqp 20}
        --- Certificate Policy (CP) URL
id-ad-prqp-certPracticesStatement
                              OBJECT IDENTIFIER ::= {id-ad-prqp 21}
        --- Certification Practices Statement (CPS) URL

--- Level Of Assurance
id-ad-prqp-certLOAPolicy      OBJECT IDENTIFIER ::= {id-ad-prqp 25}
        --- LOA Policy URL
id-ad-prqp-certLOALevel       OBJECT IDENTIFIER ::= {id-ad-prqp 26}
        --- Certificate LOA Modifier URL

--- HTTP (Browsers) based services
id-ad-prqp-httpRevokeCertificate
                              OBJECT IDENTIFIER ::= {id-ad-prqp 30}
        --- HTTP Based Certificate Revocation Service
id-ad-prqp-httpRequestCertificate
                              OBJECT IDENTIFIER ::= {id-ad-prqp 31}
        --- HTTP Based Certificate Request Service
id-ad-prqp-httpRenewCertificate
                              OBJECT IDENTIFIER ::= {id-ad-prqp 32}
        --- HTTP Based Certificate Renewal Service
id-ad-prqp-httpSuspendCertificate
                              OBJECT IDENTIFIER ::= {id-ad-prqp 33}
        --- Certificate Suspension Service

--- Webdav Services
id-ad-prqp-webdavCert         OBJECT IDENTIFIER ::= {id-ad-prqp 40}
```

```
        --- Webdav Certificate Validation
id-ad-prqp-webdavRev          OBJECT IDENTIFIER ::= {id-ad-prqp 41}
        --- Webdav Certificate Revocation

--- Grid Specific Services
id-ad-prqp-grid-accreditationBody
                              OBJECT IDENTIFIER ::= {id-ad-prqp 50}
        --- CA Accreditation Body(s)
id-ad-prqp-grid-accreditationPolicy
                              OBJECT IDENTIFIER ::= {id-ad-prqp 51}
        --- CA Accreditation Policy Document(s)
id-ad-prqp-grid-accreditationStatus
                              OBJECT IDENTIFIER ::= {id-ad-prqp 52}
        --- CA Accreditation Status Document(s)
id-ad-prqp-grid-commonDistributionUpdate
                              OBJECT_IDENTIFIER ::= {id-ad-prqp 53}
        --- Grid Distribution Package(s)
id-ad-prqp-grid-accreditedCACerts
                              OBJECT IDENTIFIER ::= {id-ad-prqp 54}
        --- Certificates of Currently Accredited CAs
```

The producedAt and nextUpdate define the time-frame when the response data is to be considered valid. Within the defined period, the client SHOULD NOT request for the same service. Use of wider time-frames values can help the RQA avoid duplication of requests from the same client thus potentially lowering the load of the responder. However, providing this data to a client does not ensure a lower query rate, as a server cannot rely on clients to obey the advice provided in the rersponse.

The resourceLocator bears access information for the service identified by the serviceId. The name MUST be an absolute URL, and it MUST follow the URL syntax and encoding rules specified in [RFC4248] (Hoffman, P., “The telnet URI Scheme,” October 2005.) and [RFC4266] (Hoffman, P., “The gopher URI Scheme,” November 2005.). The resourceLocator includes both a scheme (e.g., HTTP or FTP) and a scheme specific part. The scheme specific part is supposed to carry information on how to reach the requested service, this is, for example, a fully qualified domain name or IP address as the host. If the requested service is not available or it is unknown by the server, the resourceLocator value should be empty.

Optional Extensions may be added if requested.

TOC

3.3. IANA Considerations

This document has no actions for IANA.

---

4. PRQP Design Rationale

TOC

In this section we provide some considerations about the protocol design and its details.

4.1. Response Complexity

TOC

An important design consideration is the complexity of messages. Some type of services, e.g. delta CRLs, can be directly detected upon data downloading. On the contrary if a client is looking for a specific version of a protocol or data type, the definition of a fine-grained query system would allow for data downloading only when it is actually supported by the requesting client, thus reducing the server's load.

At present we think that keeping the protocol simple will encourage its adoption in current environments because the flexibility introduced by PRQP is a big enhancement over the current options.

Moreover, without requiring changes to the protocol, extensions could be defined to provide more fine grained options.

Future versions of the protocol may implement extended request and response types if required by applications.

4.2. RQA's URL distribution

TOC

The AIA and SIA extensions in certificates can be used to carry the pointer to the RQA. If no RQA address is present in the certificate, a client application could use a default configured URL.

Although this approach seems to contraddict the criticism of Certificate extensions use in Section 2.1.1 (Certificate Extensions), using only one extension to locate the RQA would provide an easy way to distribute the RQA's URL.

The usage of PRQP will provide a gateway for all the other services and data URLs.

4.3. Security Considerations

TOC

The PRQP provides URLs for PKI resources. This means that it provides locators to data and services, not the data per se. It still remains the client's job to access the provided URLs to gather the needed data.

Both NONCEs and signatures are optional in order to provide flexibility in how requests and responses are generated.

It is possible to provide pre-computed responses in case the NONCE is not provided by the client. This allows the RQA to generate off-line signatures for responses, an optimization used in OCSP.

Moreover if an authenticated secure channel is used at the transport level between the client and the RQA (e.g. HTTPS or SFTP) signatures in requests and responses can be safely omitted.

4.4. Time Validity

TOC

The time validity should reflect the frequency of updates in configured URLs. An interesting aspect to be considered is how often would users execute the protocol for a given set of data.

If the clients query the server often it could be a serious burden on the server but, if executed rarely, clients would not be able to discover changes in provided resources.

As described in more detail in Appendix A (Distribution of PRQP Responses), the adoption of a validity time frame for responses can be used as a mean to balance the trade off between this two aspects, but this is merely advisory data for clients and thus not a guarantee against DoS attacks by clients.

4.5. Message Format

TOC

Two different candidates have been considered. The first one is the Extensible Markup Language (XML), while the second one is the Distinguished Encoding Rules (DER).

The adoption of the Abstract Syntax Notation (ASN.1) to describe the data structures allows a software developer to provide either DER or XML based implementations of the protocol.

However we think that a DER based implementation of PRQP is the best choice because of compatibility considerations with existing applications and APIs. Moreover DER encoded messages are smaller in size then XML encoded ones and almost all PKI aware applications already support it.

---

5. Acknowledgments

TOC

The authors would like to thank Stephen Kent for his insightful comments about PRQP and his help in writing this document.

---

6. References

TOC

6.1. Normative References

TOC

[RFC1035]	Mockapetris, P., "Domain names - implementation and specification," STD 13, RFC 1035, November 1987 (TXT).
[RFC2119]	Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," BCP 14, RFC 2119, March 1997 (TXT, HTML, XML).
[RFC2560]	Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP," RFC 2560, June 1999 (TXT).
[RFC2608]	Guttman, E., Perkins, C., Veizades, J., and M. Day, "Service Location Protocol, Version 2," RFC 2608, June 1999 (TXT).
[RFC2609]	Guttman, E., Perkins, C., and J. Kempf, "Service Templates and Service: Schemes," RFC 2609, June 1999 (TXT).
[RFC2782]	Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)," RFC 2782, February 2000 (TXT).
[RFC3280]	Housley, R., Polk, W., Ford, W., and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," RFC 3280, April 2002 (TXT).
[RFC4210]	Adams, C., Farrell, S., Kause, T., and T. Mononen, "Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP)," RFC 4210, September 2005 (TXT).
[RFC4248]	Hoffman, P., "The telnet URI Scheme," RFC 4248, October 2005 (TXT).
[RFC4266]	Hoffman, P., "The gopher URI Scheme," RFC 4266, November 2005 (TXT).

6.2. Non-Normative References

TOC

[PEACH]	Pala, M. and S. Smith, "Peaches and Peers," LNCS 5057, June 2008.

Appendix A. Distribution of PRQP Responses

TOC

A.1. PRQP over HTTP

TOC

This section describes the formatting needed in order to route PRQP request and response over HTTP.

A.1.1. Request

TOC

HTTP based PRQP requests SHOULD use the POST method to submit their requests. Where privacy is a requirement, PRQP transactions exchanged using HTTP MAY be protected using either TLS/SSL or some other lower layer protocol.

The required HTTP headers for the request are:

- *Content-Type
- *Content-Transfer-Encoding
- *Content-Length

The Content-Type header SHOULD be set to "application/prqp-request". The Content-Transfer-Encoding SHOULD be set to "Binary", while the Content-Length SHOULD be set to the length (in bytes) of the body of the request. The body of the HTTP message MUST carry the binary value of the DER encoding of the PRQPRequest.

A.1.2. Response

TOC

An HTTP-based PRQP response is composed of the appropriate HTTP headers, followed by the binary value of the DER encoding of the PRQPPResponse.

The required HTTP headers for the response are:

- *Content-Type
- *Content-Transfer-Encoding
- *Content-Length

The Content-Type header SHOULD be set to "application/prqp-response". The Content-Transfer-Encoding SHOULD be set to "Binary", while the Content-Length SHOULD be set to the length (in bytes) of the body of

the request. The body of the HTTP message MUST carry the binary value of the DER encoding of the PRQPResponse.

A.1.3. Message Caching

TOC

To minimize bandwidth usage, clients MUST locally cache authoritative PRQP responses for the validity period of the request. To enable proxy servers to be able to cache responses as well, additional HTTP headers MAY be used in the response.

The PRQP responder MAY ease caching by setting the following headers:

- *date
- *last-modified
- *expires

In particular, the date field SHOULD carry the date at which the HTTP response has been generated. The last-modified, instead, SHOULD bear the date at which the response has been modified. This field SHOULD carry the same date as the producedAt field of the PRQPResponse. The expires field SHOULD carry the date till when the response is to be considered valid. This field SHOULD carry the same date as in the nextUpdate field of the PRQPResponse.

An example HTTP response would look like:

```
HTTP/1.0 200 OK
Content-Type: application/prqp-response
Content-Transfer-Encoding: Binary
Content-Length: 860
Date: Thu, 03 May 2007 04:43:43 GMT
Last-Modified: Thu, 03 May 2007 04:43:42 GMT
Expires: Thu, 04 May 2007 04:43:42 GMT

<...response data...>
```

PRQP clients SUOULD NOT included a no-cache header in PRQP request messages, unless the client encounters an expired response which may be a result of an intermediate proxy caching stale data.

TOC

A.2. PRQP over Peer-to-Peer Network

PRQP offers a starting point for the development of a PKI Resource Discovery Architecture where different RQAs cooperate to access data not locally available.

One technology that already provides good results in data sharing is Peer-to-Peer (P2P) networking.

Signed PRQP requests and responses can be routed also on existing P2P networks or a PRQP-specific network can be setup to provide a World Wide PKI Resources Discovery Architecture (PRDA), the definition of which is out of the scope of this document. An example of such an architecture is PEACH [PEACH] (Pala, M. and S. Smith, “Peaches and Peers,” June 2008.)

TOC

Appendix B. PRQP ASN1.1 Specification

```
PRQP DEFINITIONS EXPLICIT TAGS ::=

BEGIN

-- EXPORTS ALL --

IMPORTS

      -- Directory Authentication Framework (X.509)
            Certificate, AlgorithmIdentifier
            FROM AuthenticationFramework { joint-iso-itu-t ds(5)
                    module(1) authenticationFramework(7) 3 }


      --  PKIX Certificate Extensions
            AuthorityKeyIdentifier, SubjectKeyIdentifier, KeyIdentifier,
          FROM PKIX1Implicit88 {iso(1) identified-organization(3)
                  dod(6) internet(1) security(5) mechanisms(5) pkix(7)
                  id-mod(0) id-pkix1-implicit-88(2)}


             CertificateSerialNumber, Extensions, id-kp, id-ad-prqp
          FROM PKIX1Explicit88 {iso(1) identified-organization(3)
                  dod(6) internet(1) security(5) mechanisms(5) pkix(7)
                  id-mod(0) id-pkix1-explicit-88(1)};


    PRQPRequest ::= SEQUENCE {
        requestData            TBSReqData,
        signature              [0] EXPLICIT Signature OPTIONAL }

    TBSReqData ::= SEQUENCE {
        version                INTEGER { v(1) },
        nonce                  INTEGER              OPTIONAL,
              -- very large number
        maxRespEntries         INTEGER              OPTIONAL,
              -- maximum number of accepted entries in
              -- corresponding response
        serviceToken           ResourceRequestToken,
              -- token identifying the requested service
        extensions         [0] IMPLICIT Extensions  OPTIONAL }


    ResourceRequestToken ::= SEQUENCE {
        ca                      CertIdentifier,
        servicesList        [0] SET OF ResourceIdentifier OPTIONAL }

      BasicCertIdentifier ::= SEQUENCE {
        issuerNameHash              OCTET STRING,
```

```
    serialNumber                CertificateSerialNumber  }

  ExtenderCertInfo ::= SEQUENCE {
    certificateHash             OCTET STRING,
    subjectKeyHash              OCTET STRING,
    subjectKeyIdentifier    [0] KeyIdentifier          OPTIONAL,
    issuerKeyIdentifier     [1] KeyIdentifier          OPTIONAL  }

  CertIdentifier ::= SEQUENCE {
    hashAlgorithm               AlgorithmIdentifier,
    basicCertIdentifier         BasicCertIdentifier,
    extInfo                     [0] ExtendedCertInfo    OPTIONAL,
    caCertificate               [1] Certificate         OPTIONAL,
    issuedCertificate           [2] Certificate         OPTIONAL }


  ResourceIdentifier ::= SEQUENCE {
    resourceId             OBJECT IDENTIFIER,
    version            [0] INTEGER             OPTIONAL
      --- version of the protocol or data format (if applicable) }


PRQPResponse ::= SEQUENCE {
    respData               TBSRespData,
    signature          [0] EXPLICIT Signature OPTIONAL           }


TBSRespData ::= SEQUENCE {
    version                INTEGER { v(1)},
    nonce                  INTEGER              OPTIONAL,
          -- as duplicated from the request
    producedAt             GeneralizedTime,
          -- time when the response has been generated
    nextUpdate         [0] GeneralizedTime      OPTIONAL,
          -- time till when the response should be considered
          -- valid
    pkiStatus              PKIStatusInfo,
          -- status of the response
    caCertId               CertIdentifier,
          -- identifier of the CA the targeted certificate is
          -- issued from
    responseToken          SEQUENCE OF ResourceResponseToken
                                                          OPTIONAL,
          -- token carrying informations about
          -- requested services
    extensions         [0] EXPLICIT Extensions  OPTIONAL }


PKIStatusInfo ::= SEQUENCE {
    status        PKIStatus,
```

```
        statusString  [0] UTF8String     OPTIONAL,
        failInfo      [1] PKIFailureInfo  OPTIONAL  }


    PKIStatus ::= INTEGER {
        ok                      (0),
           -- when the PKIStatus contains the value zero one or
              more responseToken is present
        badRequest              (1),
           -- the request is badly formatted
        caNotPresent            (2),
           -- the requested CA is not present
        systemFailure           (3)
           -- a system failure has occourred }


    Signature ::= SEQUENCE {
        signatureAlgorithm     AlgorithmIdentifier,
        signature              BIT STRING,
        certs              [0] EXPLICIT SEQUENCE OF Certificate
                                                    OPTIONAL }


      ResourceInfo ::= SEQUENCE {
        resourceUri            IA5String,
            --- resource locator
        version            [0] INTEGER             OPTIONAL,
            --- version of the protocol or data format (if applicable)}


    ResourceResponseToken ::= SEQUENCE {
        serviceId              OBJECT IDENTIFIER,
        resourceLocator    [0] EXPLICIT SEQUENCE OF ResourceInfo      }


-- Object Identifiers

id-kp-PRQPSigning       OBJECT IDENTIFIER ::= { id-kp 10 }
id-prqp                 OBJECT IDENTIFIER ::= { id-pkix 23 }
id-prqp-pta             OBJECT IDENTIFIER ::= { id-prqp 1 }

id-ad-prqp                   OBJECT IDENTIFIER ::= {id-ad 12 }
id-ad-prqp-ocsp              OBJECT IDENTIFIER ::= {id-ad-prqp 1}
id-ad-prqp-caIssuers         OBJECT IDENTIFIER ::= {id-ad-prqp 2}
id-ad-prqp-timestamping      OBJECT IDENTIFIER ::= {id-ad-prqp 3}
id-ad-prqp-dvcs              OBJECT IDENTIFIER ::= {id-ad-prqp 4}
id-ad-prqp-caRepository      OBJECT IDENTIFIER ::= {id-ad-prqp 5}
id-ad-prqp-http-certs        OBJECT IDENTIFIER ::= {id-ad-prqp 6}
        --- HTTP certificate repository
id-ad-prqp-http-crls         OBJECT IDENTIFIER ::= {id-ad-prqp 7}
```

```
        --- HTTP CRL download URL

id-ad-prqp-xkmsGateway        OBJECT IDENTIFIER ::= {id-ad-prqp 10}
        --- XKMS Gateway
id-ad-prqp-cmsGateway         OBJECT IDENTIFIER ::= {id-ad-prqp 11}
        --- CMS Gateway
id-ad-prqp-scepGateway        OBJECT IDENTIFIER ::= {id-ad-prqp 12}
        --- SCEP Gateway

--- Certificate Policies
id-ad-prqp-certPolicy         OBJECT IDENTIFIER ::= {id-ad-prqp 20}
        --- Certificate Policy (CP) URL
id-ad-prqp-certPracticesStatement
                              OBJECT IDENTIFIER ::= {id-ad-prqp 21}
        --- Certification Practices Statement (CPS) URL

--- Level Of Assurance
id-ad-prqp-certLOAPolicy      OBJECT IDENTIFIER ::= {id-ad-prqp 25}
        --- LOA Policy URL
id-ad-prqp-certLOALevel       OBJECT IDENTIFIER ::= {id-ad-prqp 26}
        --- Certificate LOA Modifier URL

--- HTTP (Browsers) based services
id-ad-prqp-httpRevokeCertificate
                              OBJECT IDENTIFIER ::= {id-ad-prqp 30}
        --- HTTP Based Certificate Revocation Service
id-ad-prqp-httpRequestCertificate
                              OBJECT IDENTIFIER ::= {id-ad-prqp 31}
        --- HTTP Based Certificate Request Service
id-ad-prqp-httpRenewCertificate
                              OBJECT IDENTIFIER ::= {id-ad-prqp 32}
        --- HTTP Based Certificate Renewal Service
id-ad-prqp-httpSuspendCertificate
                              OBJECT IDENTIFIER ::= {id-ad-prqp 33}
        --- Certificate Suspension Service

--- Webdav Services
id-ad-prqp-webdavCert         OBJECT IDENTIFIER ::= {id-ad-prqp 40}
        --- Webdav Certificate Validation
id-ad-prqp-webdavRev          OBJECT IDENTIFIER ::= {id-ad-prqp 41}
        --- Webdav Certificate Revocation

--- Grid Specific Services
id-ad-prqp-grid-accreditationBody
                              OBJECT IDENTIFIER ::= {id-ad-prqp 50}
        --- CA Accreditation Body(s)
id-ad-prqp-grid-accreditationPolicy
                              OBJECT IDENTIFIER ::= {id-ad-prqp 51}
        --- CA Accreditation Policy Document(s)
```

```
id-ad-prqp-grid-accreditationStatus
                              OBJECT IDENTIFIER ::= {id-ad-prqp 52}
        --- CA Accreditation Status Document(s)
id-ad-prqp-grid-commonDistributionUpdate
                              OBJECT_IDENTIFIER ::= {id-ad-prqp 53}
        --- Grid Distribution Package(s)
id-ad-prqp-grid-accreditedCACerts
                              OBJECT IDENTIFIER ::= {id-ad-prqp 54}
        --- Certificates of Currently Accredited CAs
```

Author's Address

TOC

	Massimiliano Pala
	Dartmouth College
	6211 Sudikoff PKI/Trust Lab
	Hanover, NH 03755
	US
Email:	pala@cs.dartmouth.edu
URI:	http://www.openca.org

Full Copyright Statement

TOC

Intellectual Property

procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.