PKI Resource Query Protocol

The **PKI Resource Query Protocol (PRQP)** is an Internet protocol used for obtaining information about services associated with a X.509 Certification Authority. It is described in a draft document from the PKIX working group and is on the Experimental track. It was created by Massimiliano Pala to solve Interoperability and Usabilities issues among PKIs, specifically addressing certain problems associated with finding services and data repositories associated with a CA. Messages communicated via PRQP are encoded in ASN.1 and are usually communicated over HTTP.

**Background**

At present, ever more services and protocols are being defined to address different needs of users and administrators in PKIs. With the deployment of new applications and services, the need to access PKI resources provided by different organizations is critical. Each application needs to be told about how to find these services for each new certificate it encounters. Therefore, each application needs to be properly configured by filling in complex configuration options whose meaning is mostly unknown to the average user (and likely to the administrator as well).

The **PRQP Protocol** addresses these interoperability and trust building issues among separate PKI islands. In fact, it provides a dynamic method capable to provide more timely information about provided services and available PKI resources. It can also be used to help in painless rollover between services, e.g. switching from CRLs to OCSP for certificate validation. Moreover, PRQP allows PKI management to dynamically choose which services are to be provided based on the faced challenges during infrastructure deployment. In other words, PRQP can be thought to be similar (in concept) to a DNS for PKI resources.

**Related Methods**

In PKIs there are three other primary methods for clients to obtain pointers to PKI data: adopting specific certificate extensions; looking at easily accessible repositories (e.g. DNS, local database, etc.); and adapting existing protocols (e.g. Web Services).

**Certificate Extensions**

To provide pointers to published data, a CA could use the **Authority Information Access (AIA)** and **Subject Information Access (SIA)** extensions as detailed in RFC-3280. The former can provide information about the issuer of the certificate while the latter carries information (inside CA certificates) about offered services. The **Subject Information Access** extension can carry a URI to point to certificate repositories and timestamping services. Hence this extension allows to access services by several different protocols (e.g. HTTP, FTP, LDAP or SMTP).

Although encouraged, usage of the AIA and SIA extension is still not widely deployed. There are two main reasons for this. The first is the lack of support for such extensions in available clients. The second reason is that extensions are static, i.e. not modifiable. Indeed to modify or add new extensions, in order to have users and applications to be aware of new services or their dismissal, the certificate must be re-issued.

This would not be feasible for End Entities (EE) certificates, except during periodic reissuing, but it would be feasible for the CA certificate itself. The CA could retain the same public key and name and just add new values to the AIA extension in the new certificate. If users fetch the CA cert regularly, rather than caching it, this would enable them to become aware of the new services. Although this is possible, almost every available clients do not look for CAs certificates if they are already stored in clients' local database.

In any case, since URLs tend to change quite often while certificates persist for longer time frames, experience suggests that these extensions invariably point to URLs that no longer exist. Moreover considering the fact that the
entity that issues the certificates and the one who runs the services may not be the same, it is infeasible that the issuing CA will reissue all of its certificate in case a server URL’s changes. Therefore it is not wise to depend on the usage of AIA or SIA extensions for available services and repositories lookup.

DNS Service Records
The SRV record or DNS Service record technique is thought to provide pointers to servers directly in the DNS (RFC 1035). As defined in RFC 2782, the introduction of this type of record allows administrators to perform operations rather similar to the ones needed to solve the problem PRQP addresses, i.e. an easily configurable PKI discovery service.

The basic idea is to have the client query the DNS for a specific SRV record. For example if an SRV-aware LDAP client wants to discover an LDAP server for a certain domain, it performs a DNS lookup for _ldap._tcp.example.com (the _tcp means the client requesting a TCP enabled LDAP server). The returned record contains information on the priority, the weight, the port and the target for the service in that domain.

The problem in the adoption of this mechanism is that in PKIs (unlike DNS) there is usually no fixed requirement for the name space used. Most of the time, there is no correspondence between DNS structure and data contained in the certificates. The only exception is when the Domain Component (DC) attributes are used in the certificate's Subject.

The DC attributes are used to specify domain components of a DNS name, for example the domain name example.com could be represented by using the dc=com, dc=example format. If the CA's subject field would make use of such a format, the Issuer field would allow client applications to perform DNS lookups for the provided domain where the information about repositories and services could be stored.

However, currently, the practice is very different. In fact it is extremely difficult for a client to map digital certificates to DNS records because the DC format is not widely adopted by existing CAs. For example, only one certificate from IE7/Outlook certificates store uses the domain components to provide a mapping between the certificate and an Internet Domain.

External links
• PKI Resource Query Protocol [1]
• PKIX Working Group [2]

References
Article Sources and Contributors


License

Creative Commons Attribution-Share Alike 3.0 Unported
creativecommons.org/licenses/by-sa/3.0/