PKIX Working Group A. Rundgren (X-OBI)
Internet Draft January 2003
Document: draft-rundgren-pkix-pnppki4ws-00.txt
Expires: July 2003

Internet X.509 Public Key Infrastructure
Plug-and-Play PKI for Web Services

Status of this Memo

Please send comments on this document to the ietf-pkix@imc.org mailing list.

Abstract

"Web Services" [2, 3] is the collective name of a set of emerging technologies targeted for ultimately supporting Plug-and-Play (PnP) integration of business- and information-systems, within and between organizations. This specification covers an X.509 certificate extension, designed to enable PnP-support for the Public Key Infrastructure (PKI) part of a Web Service. In addition to supporting Web Services, the extension is also intended to be useable for general-purpose PKI-enabled applications. A PowerPoint presentation highlighting the core of this specification is also available [4]

Conventions used in this document

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 RFC-2119 [5].

Table of Contents

1. Introduction and rationale....................................3
   1.1 Globally unique subject DNs................................3
   1.2 Automatic sharing of naming domains........................4
   1.3 Unified EE-certificate-type per CA-certificate & key.......5
   1.4 End-entity type-indicator..................................5
   1.5 Relying-party trust-administration processes...............6
      1.5.1 Current trust-administration process..................6
      1.5.2 Enhanced trust-administration process.................6
   1.6 Permanent identifier option................................7
   1.7 Migration to plug-and-play support.........................8
   1.8 X.500 directory conformance................................8
2. Formal definition..9
   2.1 Naming domain and naming authority.........................9
   2.2 ASN.1 definition of the PnP-descriptor.....................9
   2.3 Detailed element description..............................10
   2.4 Summary of CA issuing requirements........................13
ISSUES NOT YET RESOLVED..13
Security Considerations..14
Examples...14
   E.1 SQL database sample interface.............................14
   E.2 Sample PnP-enabled CA-certificate.........................16
   E.3 Sample PnP-enabled CA-certificate with PI support.........19
Appendix...22
   A.1 1993 ASN.1 Module...22
References...22
Acknowledgments..23
Author's Address...23

Last modifications performed: 2003-01-14 12:22

1. Introduction and rationale

To combine PKI with "Web Services" on a global scale presents a challenge, as it requires Relying Parties (RPs) to process signed messages possibly emanating from many different PKIs, while preferably using "shrink-wrapped" PKI software and generic, easy-to-manage PKI trust-administration procedures.

In the web-browser environment, global interoperability is only achieved due to the fact that web-server certificates supporting HTTPS [6], are based on a static ("hard-coded") profile [7], which is a prerequisite by browsers to correctly interpret such certificates. To further simplify usage, most commercial CAs' root-certificates are already pre-installed in leading browsers.

However, "Web Services" can unlike web-browsers, not depend on static PKI-schemes and pre-installed root-certificates, as this would severely limit the kind of entity-types and certificate-profiles that would be possible for RPs to accept.

This specification introduces a CA-certificate-based, non-critical X.509 v3 extension, from now on referred to as a "PnP-descriptor", that works like an additional "specification" for associated End Entity (EE)-certificates. The following introductory sections describe how this extension can support a more dynamic PKI-based ecosystem, by removing some major hurdles to wide-scale PKI usage.

After the introduction, a formal definition of the extension is featured.

1.1 Globally unique subject DNs

The aforementioned web-server certificates, contain globally unique subject Distinguished Names (DNs) [8] due to the fact that they certify Domain Name System (DNS) [9] host-names.

However, for non-DNS-based entities, few existing certificate-profiles as well as RFC3280 [10] and RFC3039 [11] require subject DNs to be globally unique. This could possibly lead to name-clashes when multiple non-coordinated PKIs are to be handled by RPs. One way to cope with this is to associate each CA-certificate with a unique "virtual" name-space. This complicates CA-certificate renewals with respect to RPs, as well as making it more difficult to efficiently explore common certificate-profiles and associated naming-domains shared by multiple CAs (exemplified by many national ID-schemes), as both these scenarios require manual and error-prone RP configuration to work. Requiring CAs to deploy globally unique subject DNs by for example adding Domain Components (DCs) [8] is likely to be less popular, as well as breaking some existing RP software.

The PnP-descriptor therefore supports an explicit naming-domain in the form of a Universal Resource Identifier (URI) [12], which due to the two-level naming structure, provides global uniqueness to any existing or future non-empty subject DN scheme. Below is a figure, illustrating the two-level naming system.

```
           ______
          /      \
         | PnP-CA | Naming Domain: http://sample-registry.org/members
          \______/
           /    \
          /      \
         |       _\___
         |      /     \
         |     |  EE   | Subject: CN=John Doe, serialNumber=43155, C=US
         |      \_____/
       __|__
      /     \
     |  EE   | Subject: CN=Marion Anderson, serialNumber=43566, C=US
      \_____/
```

That is, Marion's fully canonicalized name could be expressed like:

    "http://sample-registry.org/members" :
    "CN=Marion Anderson, serialNumber=43566, C=US"

Note: Canonicalization syntax is outside of this specification as it is mostly a disadvantage to merge naming domain and subject DN in a real application.

In addition to forming a naming domain, HTTP-based naming domain URIs may also support dereferencing, enabling CAs to publish information concerning the naming domain, for easy access by RP "trust administrators" using standard web-browsers.

1.2 Automatic sharing of naming domains

Due to the explicit naming domain URI, and associated issuance, sharing identical naming domains between CAs is transparent, needing no configuration. As a consequence, an entity certified by one CA, can get a new certificate from another CA certifying the same naming domain, that for an RP may (depending on strictness of the issuance and certificate-profile), be authenticated as being identical, here assuming that both CAs are accepted by the RP. Also see section 1.7 "Permanent identifier option".

1.3 Unified EE-certificate-type per CA-certificate & key

A key-motive behind this specification was to simplify the adoption of PKI for relying parties by reducing the number of options that are not absolutely needed. One such option which was "sacrificed", is the ability for a CA to use the same CA-certificate and key for generating entirely different kinds of EE-certificates. Professionally run CAs seldom exploit this possibility, in order to keep different PKI-systems properly separated. Anyway, this restriction, which is to be enforced for compliance with this specification, implies that a PnP-descriptor is intended to be applicable for all EE-certificates generated by a specific PnP-enabled CA-certificate and key.

The net result is a "normalized data model" [13] applied to PKI, enabling EE-certificates to be linked to most existing business- and information-systems in a simple, secure, robust, and partially automated manner as shown in section E.1, "SQL database sample interface".

1.4 End-entity type-indicator

Since the proposed extension limits the number of EE-types per CA-certificate and key to just one, the PnP-descriptor also includes support for an EE type-indicator. The EE type-indicator gives RP system and trust administrators, a basic information of about what EE-certificates generated by a particular CA-certificate vouch for. This includes "citizen", "individual", "employee", "organization", "device", "service" or "DNS-host".

1.5 Relying-party trust-administration processes

The next section describes how an RP trust administration process has to be designed to cope with completely unprofiled but still conforming X.509 certificates. One could object to this description and claim that it is purely hypothetical, but this rather painful exercise should be seen in the light of a desire to be able to create shrink-wrapped PKI software, that is neither depending on "assumptions", nor on "best practices".

1.5.1 Current trust-administration process

Current PKI systems potentially require multiple setups, as there may be any number of different kinds of EE certificates-types associated with a certain CA as illustrated by the figure below:

```
 ______
|      |--> Web-server certificates
|  CA  |--> Low assurance e-mail certificates
|______|--> Class 3 identity certificates
```

To actually separate different kinds of EE-certificates (in order to for example only accept a specific kind), is an arbitrarily complex process ranging from reading various CA-documents to studying the contents of actual EE-certificates. To cope with possible DN name-clashes and to facilitate mapping to business systems by creating "virtual" naming-domains, CA-certificates do not only have to be stored in PKI trust-stores, but in parallel tables supporting external mapping schemes. Usually minor software "fixes" must be added as well to cope with the characteristics of certain CAs. Knowledgeable readers do probably not recognize the scenario above, but this is due to the fact that few (if any) existing PKI-enabled applications, actually support more than a few agreed-upon similar certificate-profiles.

1.5.2 Enhanced trust-administration process

Now repeating the RP trust administration process for a new CA supporting the PnP-descriptor, the following can be observed:
Due to guaranteed globally unique DNs, and single EE certificate-type, this process is technically (not to be confused with trust-wise), reduced to performing a single "OK" or "Cancel" operation. An "OK" typically leads to the addition of the CA-path to a PKI trust-store, but no other PKI-related data-structures need necessarily to be created or maintained.

Note: Both procedures described, also need to perform CA path-validation according to RFC3280 [10].

1.6 Permanent identifier option

The PnP-descriptor allows the optional inclusion of a Permanent Identifier (PI) descriptor object. This option associates a naming domain URI with a subject DN attribute like serialNumber (OID 2.5.4.5) or to a PKIX PI-object [14], residing in an X.509 "subjectAltName" extension. There are currently several large-scale national certificate-profiles that use permanent identifiers for holding citizen registration-codes. Other uses for PIs include organizational identifiers based on national registries or commercial registries like EAN and D&B. Such certificate-profiles would likely match the needs of business-to-business e-commerce. The advantage for RPs to use PIs as "handles" to entities rather than subject DNs, is that PIs are likely to be more stable as both individuals and organizations often are allowed to change name, without getting new registration-codes. Applied to the previous example a PI-enabled scheme could be like the following.

```
           ______
          /      \
         | PnP-CA | Naming Domain: http://sample-registry.org/members
          \______/  PI Attribute: serialNumber
           /    \
          /      \
         |       _\__
         |      /    \
         |     |  EE  | Subject: CN=John Doe, serialNumber=43155, C=US
         |      \____/
        _|__
       /    \
      |  EE  | Subject: CN=Marion Anderson, serialNumber=43566, C=US
       \____/
```

That is, Marion's ID could using PI be expressed like:

"http://sample-registry.org/members" : "43566"

1.7 Migration to plug-and-play support

In addition to being compatible with most existing EE-certificate profiles, this specification is designed in such a way that it allows CAs to deploy the described scheme without necessarily recalling existing EE-certificates. Depending on CA policies and software, CA-certificates can optionally be regenerated using old private keys and validity data but with the PnP-descriptor added. RPs supporting the PnP-descriptor can subsequently at their discretion migrate by utilizing regenerated CA-certificates. As the PnP-descriptor is a non-critical extension, it should not break certificate-processing software regardless if it is "understood" or not.

In case PKIs use permanent identifiers (see section 1.6, "Permanent identifier option"), PnP provides enhanced support for many existing DN-based schemes as well as those who use the PKIX PI-object [14].

1.8 X.500 directory conformance

As this specification does not in any way change, or recommend any particular DN scheme, this specification is essentially X.500 directory [15] agnostic. Long-term however, it could be a future LDAP [16] feature to support the PnP-descriptor's naming domain URI.

2. Formal definition

2.1 Naming domain and naming authority

"Naming Domain" is in this specification to be regarded as equivalent to the term name-space. Naming domains MUST be given as globally unique URIs (e.g. "http://sample-registry.org/members"). Note that different CAs MAY share naming domains, I.e. a CA does not have to be authoritative of the naming domain it certifies.

"Naming Authority" on the other hand, is the organization (or other entity) registered as owner of the domain-part of the naming domain URI (e.g. "sample-registry.org"). As a consequence, a single naming authority MAY control any number of distinct naming domains.

2.2 ASN.1 definition of the PnP-descriptor

Below follows the ASN.1 definition of the PnP-descriptor extension. Note that the PnP-descriptor is defined under the PKIX private extension arc.

```
id-pe-pnpDescriptor OBJECT IDENTIFIER ::= { id-pe <<TDB>> }

pnpDescriptor ::= SEQUENCE {

   namingDomainID UTF8String,           -- DN naming-domain URI

   entityVerboseDescription UTF8String, -- Verbose description

   entityType EntityType,   -- Basic EE-certificate type

-- If the following element is defined, all EE-certificates
-- MUST contain a conforming permanent identifier (PI)

   permanentIdentifierDescriptor PermanentIdentifierType OPTIONAL }


EntityType ::= ENUMERATION {
   Organization(0), Department(1), Individual(2), Customer(3),
   Account(4), Service(5), Device(6), Member(7), Citizen(8),
   Licensee(9), Employee(10), DNSHostName(11), CA(12), Other(13) }
```

```
PermanentIdentifierType ::= SEQUENCE {

-- If the following element is undefined, the
-- "namingDomainID" governs the PI as well

   piNamingDomainID UTF8String OPTIONAL, -- PI naming-domain URI


-- Subject RDN attribute holding the PI-data.
-- If not specified the value is to be found in a PKIX PI-object

   attributeID [0] IMPLICIT OBJECT IDENTIFIER,


-- In case there are multiple elements of the same type as
-- indicated by "attributeID", "instance" selects the right
-- one, indexing from low to high memory

   instance [1] IMPLICIT INTEGER OPTIONAL }
```

2.3 Detailed element description

namingDomainID UTF8String

"namingDomainID" is a URI holding the naming domain of the subject DN. For interoperability reasons it is RECOMMENDED to limit the length of this object to 250 characters. For international use it is RECOMMENDED to only use US-ASCII [17] characters in naming domain URIs. The rationale behind this recommendation is that US-ASCII is close to universally known by system and trust administrators and is therefore easier to communicate, regardless if using verbal or various electronic means. To aid system and trust administrators, it is RECOMMENDED to supply the naming domain in the form of a web-browser-accessible, HTTP or HTTPS URI pointing to applicable information regarding the associated naming domain.

entityVerboseDescription UTF8String

"entityVerboseDescription" is a verbose description of the purpose of associated EE-certificates. It is RECOMMENDED to limit this text to 250 characters. It is RECOMMENDED to exclude control-characters from this text with the exception of linefeed (0x0A). Due to the fact that this text is primarily intended to be read by system and trust administrators, it is RECOMMENDED for international use, to provide this text in English, only using the ISO 8859-1 [18] character-set.

entityType EntityType

"entityType" is an enumerated value giving system and trust administrators a basic information of what associated EE-certificates vouch for according to the following table, where the values in "()" represent the applicable enumeration constant:

Organization(0)	The entity represented MUST be an organizational-only entity. An organization MAY be a legal entity. In case a certificate also defines the name of a representative, the name etc. of the representative MUST be put outside of the subject DN string, preferably in a SubjectAltName extension.
Department(1)	The entity represented MUST be a departmental-only-entity. In case a certificate also defines the name of a representative, the name etc. of the representative MUST be put outside of the subject DN string, preferably in a SubjectAltName extension.
Individual(2)	The entity represented MUST be an individual.
Customer(3)	The entity represented MUST be a customer. This MAY be an individual, organization, or other entity.
Account(4)	The entity represented MUST be an account owner.
Service(5)	The entity represented MUST be a specific software service (e.g. an OCSP [19] provider).
Device(6)	The entity represented MUST be a specific hardware device like a smart card, mobile phone, car, or router.
Member(7)	The entity represented MUST be a member.
Citizen(8)	The entity represented MUST be a citizen.
Licensee(9)	The entity represented MUST be a licensee.
Employee(10)	The entity represented MUST be an employee.

DNSHostName(11) The entity represented MUST be the registered owner of the associated DNS domain. The owner MAY be an organization or individual. Due to current practice, DNS host-names SHOULD be stored in subject Common Name (CN) attributes. Note that although DNS-names belong to the "Internet" naming domain, the registered owners represented by the CA usually do not.

CA(12) The entity represented MUST be a CA.

Other(13) To use when nothing of the above applies.

permanentIdentifierDescriptor PermanentIdentifierType OPTIONAL

"permanentIdentifierDescriptor" is an OPTIONAL element which if defined, indicates that associated EE-certificates contain PI-values, matching the declarations of "attributeID" and "instance".

piNamingDomainID UTF8String OPTIONAL

"piNamingDomainID" is an OPTIONAL URI defining the naming domain of associated PI values. If undefined, "namingDomainID" MUST be assumed to valid for PI-values as well. For RECOMMENDED guidelines: see "namingDomainID".

attributeID [0] IMPLICIT OBJECT IDENTIFIER OPTIONAL

"attributeID", if defined, denotes the Object Identifier (OID) of the subject Relative Distinguished Name (RDN) attribute holding the PI-value of associated EE-certificates. If on the other hand "attributeID" is undefined, the associated PI-value is supposed to be residing in a PKIX PI-object [14]. In case PKIX PI-objects are used as PnP PIs, any defined PKIX PI "IdentifierType" MUST match the PI naming domain as given by the PnP-descriptor. It is RECOMMENDED in a PnP-configuration to not define any associated PKIX PI elements except "identifierValue". Note: Regardless of PI-method used, a CA MAY deploy any number of self-contained "secondary" PKIX PI-objects.

instance [1] IMPLICIT INTEGER OPTIONAL

"instance" is an OPTIONAL integer which MUST be defined if there are multiple instances of "attributeID" in subject DNs, or if "attributeID" is undefined, if there are multiple PKIX PI-objects. The "instance" value tells which one, indexing from low to high memory is actually holding the PI-value of associated EE-certificates.

2.4 Summary of CA issuing requirements

The following is required by a CA with respect to issued EE-certificates for a specific PnP-enabled CA-certificate, in order to be compatible with this specification:

1. All EE-certificates MUST belong to the naming domain specified in "namingDomainID".
2. All EE-certificates MUST vouch for the same kind of entity as specified by "entityType".
3. In case the PnP-specifier indicates that PIs are supported, all EE-certificates MUST have a valid PI-value object.
4. Subject DNs MUST be at least unique in the specified naming domain.
5. Subject DNs MUST NOT be reused for another entity during the entire lifetime of the issuance.
6. In case PIs are supported, PI values MUST be unique in the applicable naming domain.
7. In case PIs are supported, PI values MUST NOT be reused for another entity during the entire lifetime of the issuance.
8. CAs MAY, depending on issuing policy, assign new subject DN or PI values to a given entity over time.
9. CAs MAY, depending on issuing policy, allow entities to have multiple identities (residing in different EE-certificates).
10. CAs SHOULD NOT use canonicalized DN or PI strings, in excess of 250 characters (in order to achieve interoperability).

However, data that does not violate the requirements above MAY differ between EE-certificates. To this category belongs X.509 extensions, subject DN attributes used (with some logical restrictions), key lengths, key algorithms, and validity intervals

ISSUES NOT YET RESOLVED

I1 (A.R.) Should we support other non-DN-based PIs (except for PKIX-PI) as well? Is it actually possible to generalize the system in that the private schemes currently used can be supported with entirely generic (shrink-wrapped) software? VeriSign's DUNS-extension is only optional so it does not fit anyway.

I2 (A.R.) Wouldn't this be a good opportunity to add other things that in many (most?) cases are CA-wide and support the "CA acceptance phase"? To this category belongs Policy-extensions, Warranty-Liability extensions, Logotype extensions. The idea is that the existing definitions could be "reused" and added as element options.

I3 (A.R. initiated by T.G.) Do we need a matchingRule for PIs?

Security Considerations

T.B.D.

Examples

E.1 SQL database sample interface

The following is short example given in SQL syntax showing how plug-and-play PKI could be applied to a general-purpose business application, typically receiving signed business messages from external customers.

```
/*###########################################################*/
/*     Extremely minimalistic "business" database scheme      */
/*###########################################################*/

-- Sample customer table
CREATE TABLE Customers (
  Customer_ID       int            NOT NULL,
  CustName          nvarchar(80)  NOT NULL,
  Address           nvarchar(80)  NOT NULL)


-- Links between PnP identities and customers
CREATE TABLE PNPMappings (
  NamingAuthority   nvarchar(250) NOT NULL,  -- URI
  SubjectUniqueData nvarchar(250) NOT NULL,  -- DN or PI data
  Customer_ID       int           NOT NULL)  -- Key in "Customers"
```

```
/*==============================================================*/
/*     Create a "plain vanilla" database customer entry       */
/*==============================================================*/

INSERT INTO Customers VALUES (1, 'ACME Corp', 'L.A.')


/*==============================================================*/
/*  Simulate adding a certificate link to the customer entry  */
/*==============================================================*/
-- Assumptions:
-- 1. The certificate path is trusted and has been validated etc.
-- 2. The CA-certificate has been identified as "PnP-compliant"
-- 3. The naming authority URI has been read from the CA-certificate
-- 4. The subject DN string has been read from the EE-certificate

INSERT INTO PNPMappings
   VALUES ('http://www.sample-bizreg.org',                -- CA
           'CN=ACME, C=US',                               -- EE
           1)


/*==============================================================*/
/*     Simulate a certificate & customer lookup operation     */
/*==============================================================*/
-- Assumptions:
-- 1. The certificate path is trusted and has been validated etc.
-- 2. The CA-certificate has been identified as "PnP-compliant"
-- 3. The naming authority URI has been read from the CA-certificate
-- 4. The subject DN string has been read from the EE-certificate

SELECT Customers.Customer_ID, Customers.CustName
  FROM PNPMappings, Customers WHERE
    NamingAuthority = 'http://www.sample-bizreg.org' AND   -- CA
    SubjectUniqueData = 'CN=ACME, C=US' AND                -- EE
    PNPMappings.Customer_ID = Customers.Customer_ID

-- >> Which should generate "1" and "ACME Corp" << --
```

Note: The PnP-extension eliminates any need for RPs to store EE-certificates of external parties' for signature and authentication purposes, as EEs are identity-wise, fully qualified by the PnP-CA, and the subject-DN of a received EE-certificate. If the PnP-CA also supports permanent identifiers, the very same database schema can optionally use these to enable more robust "handles" by extracting PI-data from subject DNs (or PKIX PI-objects), before feeding it to "SubjectUniqueData".

E.2 Sample PnP-enabled CA-certificate

The following listing using DumpASN1 [20], shows a possible CA-certificate designed to support organizations, separated by a suitable subject DN scheme.

```
SEQUENCE {
  SEQUENCE {
    [0] {
      INTEGER 2
      }
    INTEGER 0
    SEQUENCE {
      OBJECT IDENTIFIER
        sha1withRSAEncryption (1 2 840 113549 1 1 5)
      NULL
      }
    SEQUENCE {
      SET {
        SEQUENCE {
          OBJECT IDENTIFIER emailAddress (1 2 840 113549 1 9 1)
          IA5String 'info@x-obi.com'
          }
        }
      SET {
        SEQUENCE {
          OBJECT IDENTIFIER organizationName (2 5 4 10)
          PrintableString 'www.x-obi.com'
          }
        }
      SET {
        SEQUENCE {
          OBJECT IDENTIFIER commonName (2 5 4 3)
          PrintableString 'X-OBI PnP CA-1'
          }
        }
      }
    SEQUENCE {
      UTCTime '020806232746Z'
      UTCTime '080127232746Z'
      }
    SEQUENCE {
      SET {
        SEQUENCE {
          OBJECT IDENTIFIER emailAddress (1 2 840 113549 1 9 1)
          IA5String 'info@x-obi.com'
          }
        }
      SET {
```

```
           SEQUENCE {
             OBJECT IDENTIFIER organizationName (2 5 4 10)
             PrintableString 'www.x-obi.com'
             }
           }
         SET {
           SEQUENCE {
             OBJECT IDENTIFIER commonName (2 5 4 3)
             PrintableString 'X-OBI PnP CA-1'
             }
           }
         }
       SEQUENCE {
         SEQUENCE {
           OBJECT IDENTIFIER rsaEncryption (1 2 840 113549 1 1 1)
           NULL
           }
         BIT STRING 0 unused bits, encapsulates {
             SEQUENCE {
               INTEGER
                 00 EA 07 F9 EF 03 63 6E 1F 50 76 BB 6F 4E D2 7E
                 2D 2A 2A C3 71 55 E6 1D BA F9 4E AE 8D AB 61 70
                 8A C2 DC 95 FC EB C6 21 78 0E BF 1F FA A6 32 15
                 CE E8 8D DC 26 46 FD 26 9D B4 0D 50 E9 AA 5C 46
                 35 54 AE 88 DD 72 26 DB 55 C8 49 09 29 80 C8 F9
                 58 70 87 09 8D DD FA 2C 96 2A A5 A4 FB 99 65 C6
                 E5 CD 19 E8 B7 E8 FD 53 EF C1 C6 3D 6A C7 B8 29
                 EA DF 64 E7 05 BA F0 62 4F 4C E6 AD FB B1 DD E9
                         [ Another 1 bytes skipped ]
               INTEGER 65537
               }
             }
         }
       [3] {
         SEQUENCE {
           SEQUENCE {
             OBJECT IDENTIFIER basicConstraints (2 5 29 19)
             BOOLEAN TRUE
             OCTET STRING, encapsulates {
                 SEQUENCE {
                   BOOLEAN TRUE
                   }
                 }
             }
           SEQUENCE {
             OBJECT IDENTIFIER keyUsage (2 5 29 15)
             OCTET STRING, encapsulates {
                 BIT STRING 1 unused bits
                   '1100011'B
```

```
                 }
             }
           SEQUENCE {
             OBJECT IDENTIFIER subjectKeyIdentifier (2 5 29 14)
             OCTET STRING, encapsulates {
                 OCTET STRING
                   9D A7 9E 84 1B 82 6B 7C 27 DE DA 3F 2F 67 3C 2E
                   2C B5 69 7D
                 }
             }
           SEQUENCE {
             OBJECT IDENTIFIER pnpDescriptor (1 3 6 1 5 5 7 1 <<TBD>>)
             OCTET STRING, encapsulates {
                 SEQUENCE {
                   UTF8String
                     'http://www.sample-bizreg.org'
                   UTF8String
                     'B2B organizational certificate'
                   ENUMERATED 0
                   }
                 }
             }
           }
         }
       }
     SEQUENCE {
       OBJECT IDENTIFIER
         sha1withRSAEncryption (1 2 840 113549 1 1 5)
       NULL
       }
     BIT STRING 0 unused bits
       37 93 96 D1 25 73 DA 16 EB 25 18 C0 14 1B B5 C6
       77 0A 05 20 3D F7 06 29 9A 79 A0 F7 33 79 51 6B
       85 FB D4 6E 06 57 22 77 24 54 6C B7 4C 3E 92 3B
       09 83 EF 9C 33 E6 FF 1E 32 CC 8B C9 CC 62 D9 29
       BE 26 D1 B1 B3 9C 37 AE F6 29 41 12 B6 51 37 02
       4B C8 FE 6C 53 D8 4A ED 58 F3 87 09 B3 0A 0A A1
       7B 87 DD C4 D2 90 4B 90 6C 32 CF 2D 3D 86 0D 92
       4D CD A7 00 05 41 38 63 EE FE C7 BE C9 C6 24 6C
     }
```

E.3 Sample PnP-enabled CA-certificate with PI support

The following listing using DumpASN1 [20], shows a possible CA-certificate designed to support members, distinguished by a subject DN-based permanent identifier in a serialNumber (OID 2.5.4.5) attribute. An example of a subject DN-string matching the listed CA-certificate would be:
"CN=Marion Anderson, serialNumber=43566, C=US".

```
SEQUENCE {
  SEQUENCE {
    [0] {
      INTEGER 2
      }
    INTEGER 0
    SEQUENCE {
      OBJECT IDENTIFIER
        sha1withRSAEncryption (1 2 840 113549 1 1 5)
      NULL
      }
    SEQUENCE {
      SET {
        SEQUENCE {
          OBJECT IDENTIFIER emailAddress (1 2 840 113549 1 9 1)
          IA5String 'info@x-obi.com'
          }
        }
      SET {
        SEQUENCE {
          OBJECT IDENTIFIER organizationName (2 5 4 10)
          PrintableString 'www.x-obi.com'
          }
        }
      SET {
        SEQUENCE {
          OBJECT IDENTIFIER commonName (2 5 4 3)
          PrintableString 'X-OBI PnP CA-2'
          }
        }
      }
    SEQUENCE {
      UTCTime '020806232746Z'
      UTCTime '080127232746Z'
      }
    SEQUENCE {
      SET {
        SEQUENCE {
          OBJECT IDENTIFIER emailAddress (1 2 840 113549 1 9 1)
          IA5String 'info@x-obi.com'
```

```
              }
            }
          SET {
            SEQUENCE {
              OBJECT IDENTIFIER organizationName (2 5 4 10)
              PrintableString 'www.x-obi.com'
              }
            }
          SET {
            SEQUENCE {
              OBJECT IDENTIFIER commonName (2 5 4 3)
              PrintableString 'X-OBI PnP CA-2'
              }
            }
          }
        SEQUENCE {
          SEQUENCE {
            OBJECT IDENTIFIER rsaEncryption (1 2 840 113549 1 1 1)
            NULL
            }
          BIT STRING 0 unused bits, encapsulates {
              SEQUENCE {
                INTEGER
                  00 EA 07 F9 EF 03 63 6E 1F 50 76 BB 6F 4E D2 7E
                  2D 2A 2A C3 71 55 E6 1D BA F9 4E AE 8D AB 61 70
                  8A C2 DC 95 FC EB C6 21 78 0E BF 1F FA A6 32 15
                  CE E8 8D DC 26 46 FD 26 9D B4 0D 50 E9 AA 5C 46
                  35 54 AE 88 DD 72 26 DB 55 C8 49 09 29 80 C8 F9
                  58 70 87 09 8D DD FA 2C 96 2A A5 A4 FB 99 65 C6
                  E5 CD 19 E8 B7 E8 FD 53 EF C1 C6 3D 6A C7 B8 29
                  EA DF 64 E7 05 BA F0 62 4F 4C E6 AD FB B1 DD E9
                          [ Another 1 bytes skipped ]
                INTEGER 65537
                }
              }
          }
        [3] {
          SEQUENCE {
            SEQUENCE {
              OBJECT IDENTIFIER basicConstraints (2 5 29 19)
              BOOLEAN TRUE
              OCTET STRING, encapsulates {
                  SEQUENCE {
                    BOOLEAN TRUE
                    }
                  }
              }
            SEQUENCE {
              OBJECT IDENTIFIER keyUsage (2 5 29 15)
```

```
            OCTET STRING, encapsulates {
                BIT STRING 1 unused bits
                  '1100011'B
                }
            }
          SEQUENCE {
            OBJECT IDENTIFIER subjectKeyIdentifier (2 5 29 14)
            OCTET STRING, encapsulates {
                OCTET STRING
                  9D A7 9E 84 1B 82 6B 7C 27 DE DA 3F 2F 67 3C 2E
                  2C B5 69 7D
                }
            }
          SEQUENCE {
            OBJECT IDENTIFIER pnpDescriptor (1 3 6 1 5 5 7 1 <<TBD>>)
            OCTET STRING, encapsulates {
                SEQUENCE {
                  UTF8String
                    'http://sample-registry.org/members'
                  UTF8String
                    'Member certificates using an internal'
                    '.unique permanent identifier'
                  ENUMERATED 7
                  SEQUENCE {
                     [0] OBJECT IDENTIFIER serialNumber (2 5 4 5)
                    }
                  }
                }
            }
          }
        }
      }
    SEQUENCE {
      OBJECT IDENTIFIER
        sha1withRSAEncryption (1 2 840 113549 1 1 5)
      NULL
      }
    BIT STRING 0 unused bits
      37 93 96 D1 25 73 DA 16 EB 25 18 C0 14 1B B5 C6
      77 0A 05 20 3D F7 06 29 9A 79 A0 F7 33 79 51 6B
      85 FB D4 6E 06 57 22 77 24 54 6C B7 4C 3E 92 3B
      09 83 EF 9C 33 E6 FF 1E 32 CC 8B C9 CC 62 D9 29
      BE 26 D1 B1 B3 9C 37 AE F6 29 41 12 B6 51 37 02
      4B C8 FE 6C 53 D8 4A ED 58 F3 87 09 B3 0A 0A A1
      7B 87 DD C4 D2 90 4B 90 6C 32 CF 2D 3D 86 0D 92
      4D CD A7 00 05 41 38 63 EE FE C7 BE C9 C6 24 6C
    }
```

Appendix

A.1 1993 ASN.1 Module

T.B.D.

References

1 Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.

2 Word Wide Web Consortium, A center for Web Services developments, http://www.w3.org

3 OASIS, A center for Web Services developments, http://www.oasis-open.org

4 Rundgren, A., "Plug-and-Play PKI for Web Services", V0.44, January 2003, http://<TDB>/PnPPKI4WS.ppt.

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

6 E. Rescorla, "HTTP Over TLS", RFC 2818, May 2000.

7 HTTPS Certificate Profile. Largely undocumented, "de-facto" system created by Netscape Corporation around 1994.

8 ITU-T Recommendation X.520: Information Technology Open Systems Interconnection The Directory: Selected Attribute Types, 1993.

9 Mockapetris, P.V., "Domain names - concepts and facilities", RFC 1034, November 1987.

10 R. Housley, W. Ford, W. Polk, and D. Solo, "Internet X.509 Public Key Infrastructure: Certificate and CRL Profile", RFC 3280, April 2002.

11 S. Santesson, W. Polk, P. Barzin, M. Nystrom, "Qualified Certificates Profile", RFC 3039, January 2001.

12 Berners-Lee, T., R. Fielding, L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax", RFC 2396, August 1998.

13 Codd, E. F., "The Relational Model for Database Management", Version 2, Addison-Wesley, 1990.

14 Pinkas, D., T. Gindin, "Permanent Identifier", Internet Draft, draft-ietf-pkix-pi-06.txt, June 2002.

15 ITU-T Recommendation X.501 (1997 E): Information Technology - Open Systems Interconnection - The Directory: Models, June 1997.

16 Wahl, M., T. Howes, and S. Kille, "Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions", RFC 2252, December 1997.

17 ANSI, Coded Character Set, 7-Bit American Standard Code for Information Interchange, ANSI X3.4-1986.

18 ISO, International Standard, Information Processing, 8-bit Single-Byte Coded Graphic Character Sets, Part 1: Latin Alphabet No. 1, ISO 8859-1:1987.

19 Myers, M., R. Ankney, A. Malpani, S. Galperin, and C. Adams, "Online Certificate Status Protocol - OCSP", RFC 2560, June 1999.

20 Guttman, P., DumpASN1 - ASN.1 object dump/syntax check program, November 2001.

Acknowledgments

The author would like to thank Tom Gindin of IBM for valuable input on the ASN.1 definition

Author's Address

Anders Rundgren
Flottiljgatan 22
75337 Uppsala
SWEDEN
Phone: +46 70 - 627 74 37
Email: anders.rundgren@telia.com