Plug-and-Play-Enabled PKI for Web Services

A rationale for an IETF-PKIX draft effort

Anders Rundgren, X-OBI

V0.47

10, January, 2003

The following PowerPoint slide-show, describes a possible extension for introducing support for *a separate naming-domain descriptor in X509.v3 CA-certificates*. In addition to holding a globally unique naming-domain, the descriptor also holds an *entity type-indicator*, as well as an optional part describing how to extract a *permanent identifier* from associated EE-certificates. This arrangement, which in effect is an enhanced PKI "data-model", has many implications, one is that existing EE-certificates as well as existing subject DN-attributes can continue to be utilized, as CA-certificates can often be *regenerated* (using old keys and validity data) with the new descriptor added. I.e. this is a *migration solution* which is an important factor for general acceptance. The result using specifically adapted certificate processing software, is enabling a considerable simpler way of setting up trust anchors (accepting CAs), as well as introducing robust and simple schemes for associating certificates with client-accounts using current relational database technology and tools. The latter is also known as *"business system compatible"*. In a somewhat market-oriented fashion the proposed extension has been coined PnP-descriptor, where PnP is a commonly used short form for "Plug-and-Play"

Problem Definition: How can a relying party's *software* determine if EE-certificates produced by a certain CA

- contains globally unique DNs
- supports permanent identifiers
- associated name-space of identifiers

or even understanding what the issued EE-certificates vouch for?

Answer: **It Cannot**. This knowledge must be "deciphered" from potentially huge certificate practice statements by *humans*, and then in an *error-prone* way be "configured" into software systems

This was the major reason for creating the PnP-specification, i.e. to "demystify" primarily TTP-based PKI, to the point where it almost can be plug-and-play...

The PnP-enabled “accept new CA” procedure

New CA

PnP

This is a previously unknown CA, that you may opt to trust

CA Root: CN=MegaCA Corp. Root Certificate Server, O=MegaCA Corp., L=Los Angeles, C=US

CA Description: CN=MegaCA Corp. Organization Certificate Server, O=MegaCA Corp., L=Los Angeles, C=US

Entity Certificate Type: Organization

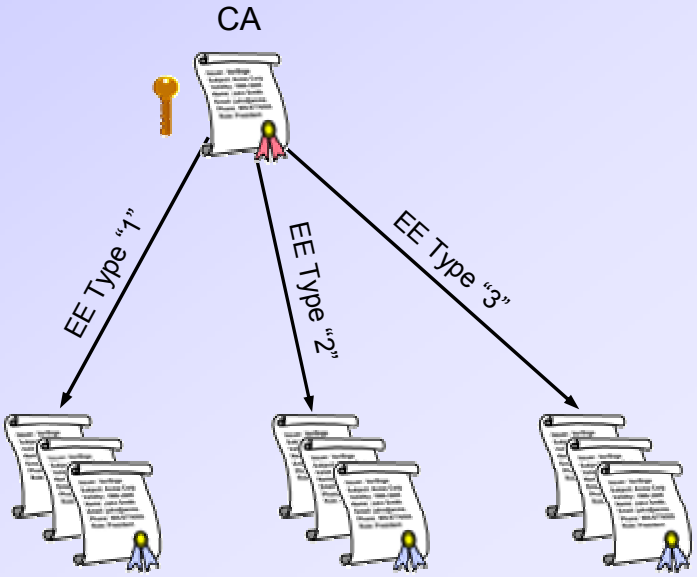
Permanent Identifier Type: DUNS

Accept... | CA Certificate details... | Cancel

Note: As the CA-certificate in the sample above contained a PnP-descriptor, very little information had to be exchanged to the relying party’s “trust administrator”. *Supporting database entries would usually be automatically configured* (see SQL sample slide for details on such) at the moment of acceptance. If the PnP-enabled CA in turn belonged to an already trusted root, the entire procedure could be automatic (assuming that the entity type matched the needs of the relying party).

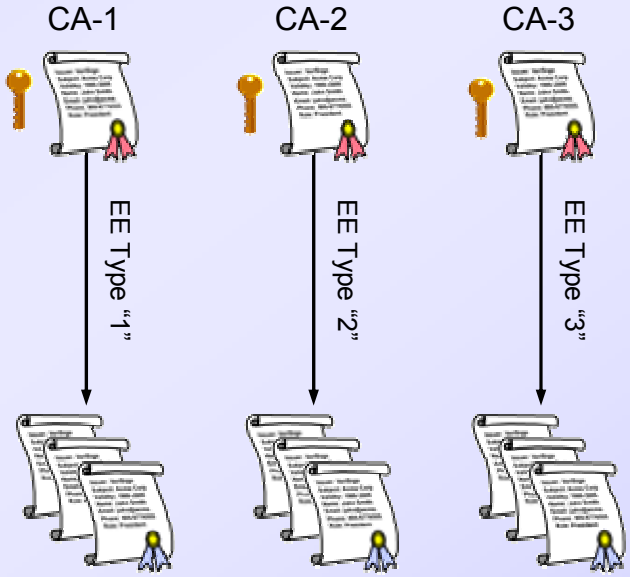
The X.500-vision versus the reality

X.500 Vision



A single CA (and key) may support any number of EE-certificates of various type (including using different policies), all having globally unique subject-DNs, based on a global registry

Common commercial practice



Each EE-certificate-type and associated policy, is supported by a single CA (and key). Depending on the type of EE-certificate, subject-DNs *may* be globally unique. Permanent identifiers are frequently deployed, currently using arbitrary representations

The reason for the current situation is not only due to a generally poor understanding of the X.500 model, but to sound operational principles as well. The reason for having separate CAs for different EE-types is twofold:
1. Different EE-types usually require different RA-software and associated procedures which is easiest to securely maintain on different machines. 2. Technical support issues, and legal issues in case of customer err. speak for separating different certificate "products". The PnP-descriptor extension exploits the current practice, while *indirectly* achieving the X.500-visionaries' globally unique identities associated with EE-certificates

Globally unique naming-domains featured in PnP-descriptors are expressed as URIs

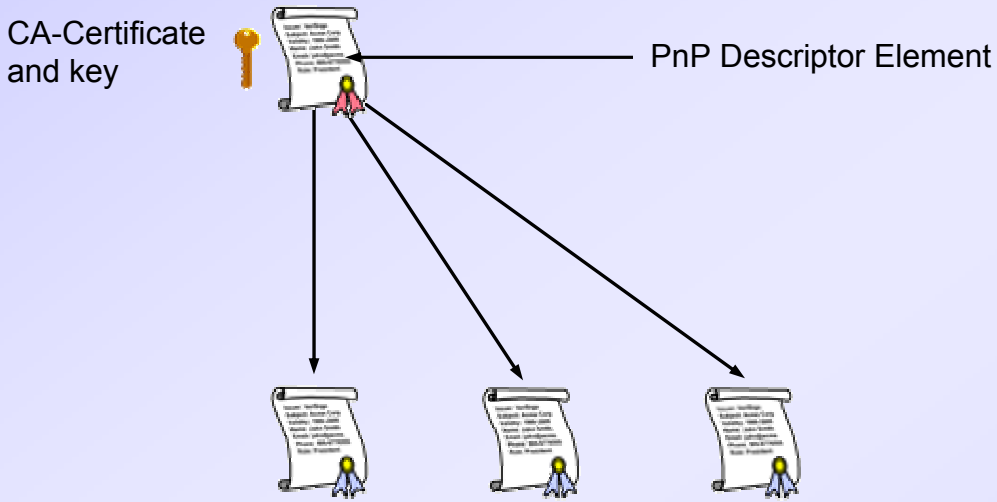
```
https://namespaces.government.se/eid_v1

http://xmlns.dnb.com/duns_numbers

http://www.acme.com/employees

http://www.icann.org/the_internet
```

The major advantage with utilizing DNS-based URIs (Universal Resource Indicators), is that a globally functioning registration-system is already in place, and used by almost every entity on the Internet. By using http-based URIs, name-space administrators can at their discretion (but are highly recommended to do so), put real documents behind URIs, containing browser-viewable descriptions of the associated naming-domains. A naming-domain URI MUST be distinct for a particular entity-type.

Note: A naming-domain is unambiguously associated to the authority owning the registered host-name of the URI

The addition of an *explicit* naming-domain is essentially formalizing current methods, that are mostly based on "conventions", where a naming-domain is *implicitly* associated to a certain CA and issuance, and then hard-coded into applications like in the case of web-server certificates and Internet-browsers. Even recent IETF-specifications like Qualified Certificates (RFC3039), also explicitly states that subject DNs only have to be unique within the CA's issuing domain.

The PnP-descriptor requires a single entity-type and naming-domain per CA-certificate and key

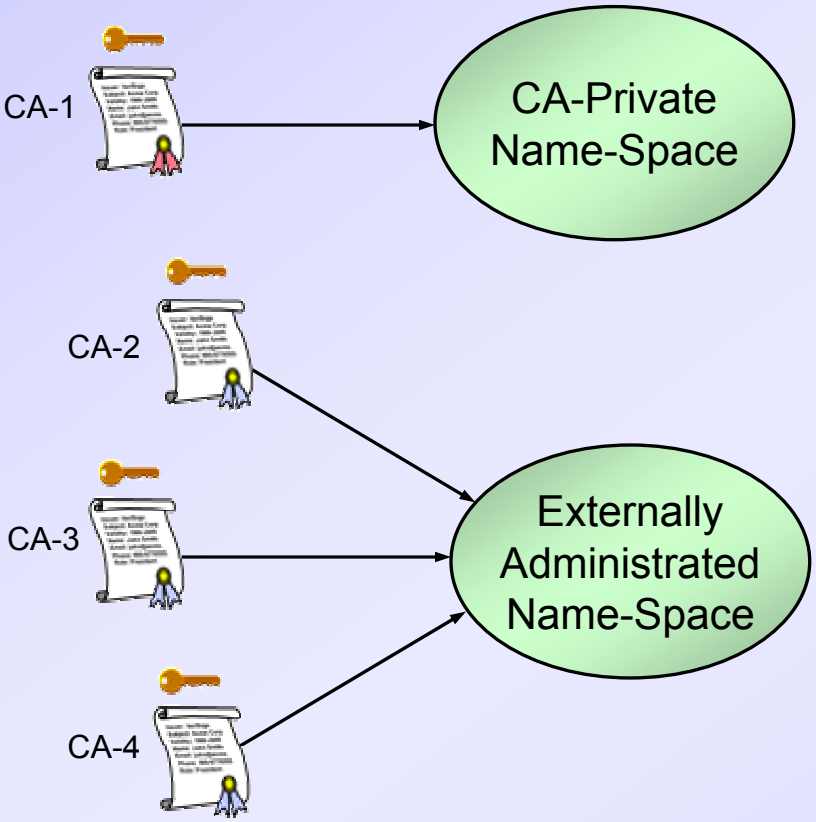


The PnP-descriptor is a non-critical extension (should not break software packages that do not understand this extension), that a CA can put in a CA-certificate. There can be only one such extension per CA-certificate. All EE-certificates produced by such a CA MUST match the descriptor which is a form of EE-certificate "specification". The resulting globally unique identity-string of an EE-certificate has two dimensions according to the following:

<*CA_GloballyUniqueNamingDomainInPnPDescriptor***>:<***EE_SubjectDistinguishedName***>**

The placement of static CA-wide information in CA-certificates, reduce data-redundancy and adhere to database-technologists' desire for having normalized data.

PnP restrict CAs to a single naming-domain per key, but naming-domains may be *shared*



- Customer IDs
- Employee IDs
- Account IDs
- Device IDs

- DNS
- DUNS
- National or regional organization registries

Note: Externally administrated name-spaces often support permanent identifiers (PIs) as well

PnP supports a permanent identifier (PI) *option*

- PIs keep the links unbroken between subjects and associated databases, regardless of certificate renewals
- PIs remain intact even if a subject changes name or other properties that may be secondary to the relying party
- PIs are very efficient for long-term, keeping track of subjects. This though reduce their applicability for privacy concerns, versus *less trusted relying parties,* if the subject is an individual or associated to an individual

Typical "classical" scheme:

EE Subject: **`CN=John Doe, OU=Marketing, O=Acme, L=New York, C=US`**

Typical PnP PI-scheme:

CA PnP naming domain: **`http://xmlns.acme.com/employees`**

CA PnP PI-OID: **`2.5.4.5`**

EE Subject: **`CN=John Doe, SerialNumber=003888390`**

Using a PnP PI-scheme, there may be any number of John Does at Acme, and they may also change position within the organization without requiring new certificates. Additional information regarding an employee must though be extracted from other sources like directories, attribute certificates, etc. Note: SerialNumber (OID.2.5.4.5), is not to be confused with certificate serial number.

And the ASN.1...

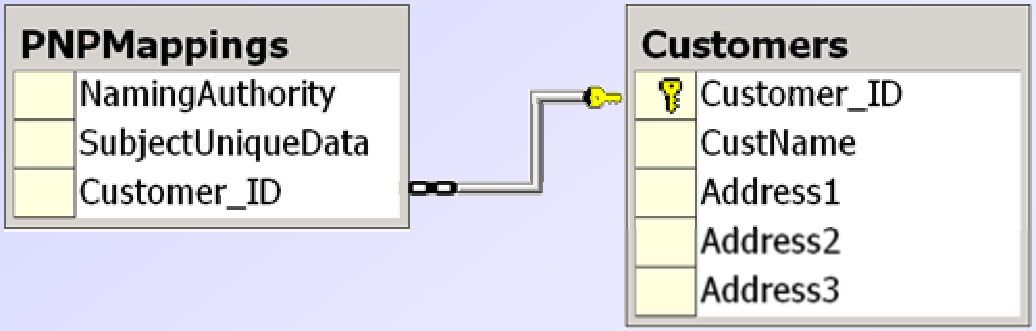
```
pnpDescriptor ::= SEQUENCE {
   namingDomainID UTF8String, -- DN naming-domain in the form of a URI
   entityVerboseDescription UTF8String, -- Verbose description of CA and entity
   entityType EntityType, -- What the associated EE-certificates represent
-- If the following element is defined, all EE-certificates MUST contain a conforming permanent identifier (PI)
   permanentIdentifierDescriptor PermanentIdentifierType OPTIONAL }

PermanentIdentifierType ::= SEQUENCE {
-- If the following element is undefined, the outer namingDomainID governs the PI as well
   piNamingDomainID UTF8String OPTIONAL, -- PI naming-domain in the form of a URI
   attributeID OBJECT IDENTIFIER,   -- The subject RDN attribute holding the PI-data
   instance INTEGER OPTIONAL }  -- In case there are multiple elements of the same type as attributeID
                                -- counting from left to right

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) }
 -- DNSHostName is an organization-certificate where CN is reserved for the DNS host
 -- In case an organization-certificate (0) or department-certificate(1), also defines the name of a
 -- representative, the name etc. of the representative MUST be put outside of the DN string,
 -- preferably in a SubjectAltName extension
```

V1.04

Notes: A PnP-descriptor which is a non-critical extension, always refer to certificates belonging to the *next level* in a certificate-path. If a PnP-descriptor also contains a permanent-identifier-descriptor, a relying party can at its discretion when handling associated EE-certificates, use the extracted PI-data and its associated naming-domain, rather than the entire subject-DN and its associated naming-domain, as a globally unique identity.

NON-NORMATIVE SAMPLE

Using PnP and relational databases - Its a snap!



Note: PnP completely eliminates the need for RPs to store EE-certificates 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 a permanent identifier, the very same scheme can optionally use these to enable more robust "links" by extracting the PI-data from the subject-DN before feeding it to the *@SubjectUniqueData* variable below. The *@NamingAuthority* variable is set by the PnP-extension of the associated CA-certificate.

PnP-enabled certificate lookup using SQL

```
SELECT Customers.Customer_ID, Customers.CustName FROM PNPMappings, Customers WHERE
   PNPMappings.NamingAuthority = @NamingAuthority AND
   PNPMappings.SubjectUniqueData = @SubjectUniqueData AND
   PNPMappings.Customer_ID = Customers.Customer_ID
```

Open questions

1. Should the PI-option be complemented with a mechanism to support non-DN-based storage of permanent identifier data?
2. Should there be a data conversion-option to PI-data as well? Some older PI-schemes put strings into octet-strings and you really want the string to be stored in databases etc.
3. Does cross-certification need some specific support to not puke on the possible enhanced path-validation algorithms?
4. Would it not be a good idea to insert other typically repeated data like policy information, logotypes, etc. in the PnP-descriptor? These would serve as defaults in the absence of such data in associated EE-certificates
5. Does the PnP-scheme need coordination with LDAP-developments to take full advantage of the CA/EE data model?

Anders.Rundgren@x-obi.com

+46 70 - 627 74 37