DEFECT REPORT FORM

1. Defect Report Number: 9594/273

Title: Name constraints conformance

2. Source: X.509 editor

3. Addressed to:

5. Date circulated by WG Secretariat:

6. Deadline for Response from Editor:

7. Defect Report Concerning:

ITU-T X.509 (1997 and 2000 editions) | ISO/IEC 9594-8:1997 and 2000 editions

8. Qualifier: Clarification

9. References in Document:

3rd edition, 1997 (clause 12.4.2.2, 12.4.3 and Annex A) & 4th edition, 2000 (clause 8.4.2.2, clause 10 (and subclauses) and Annex A)

10. Nature of Defect:

The text of this clause needs to be clarified with respect to qualification of certificates when tested against these constraints. Various interpretations have been made as to how these constraints are to be tested. The solution proposed attempts to satisfy all the (sometimes conflicting) requirements expressed for constraining names, while at the same time taking into account the information obtained about commonality in how industry participants have implemented the constraints to date.

11. Solution Proposed by the Source:

While most of the ASN.1 and significant amounts of the text remain unchanged, to enable the reader to more clearly understand the resulting behavior, it is simplest to rewrite the appropriate clause for this extension in its entirety.

Replace 12.4.2.2 (3rd edition) and clause 8.4.2.2 (4th edition) with the following:

8.4.2.2 Name constraints extension

This field, which shall be used only in a CA-certificate, indicates a name space within which all subject names in subsequent certificates in a certification path must be located. This field is defined as follows:

```
nameConstraints EXTENSION ::= {
        SYNTAX              NameConstraintsSyntax
        IDENTIFIED BY       id-ce-nameConstraint }

NameConstraintsSyntax ::= SEQUENCE {
        permittedSubtrees       [0]     GeneralSubtrees OPTIONAL,
        excludedSubtrees        [1]     GeneralSubtrees OPTIONAL,
        requiredNameForms       [2]     NameForms OPTIONAL  }
```

```
GeneralSubtrees ::= SEQUENCE SIZE (1..MAX) OF GeneralSubtree

GeneralSubtree ::= SEQUENCE {
        base                GeneralName,
        minimum     [0]     BaseDistance DEFAULT 0,
        maximum     [1]     BaseDistance OPTIONAL }

BaseDistance ::= INTEGER (0..MAX)


NameForms  ::= SEQUENCE {
    basicNameForms      [0]     BasicNameForms OPTIONAL,
    otherNameForms      [1]     SEQUENCE SIZE (1..MAX) OF OBJECT IDENTIFIER OPTIONAL }
(ALL EXCEPT ({ --none; i.e.:at least one component shall be present-- }))

BasicNameForms  ::= BIT STRING {
        rfc822Name                  (0),
        dNSName                     (1),
        x400Address                 (2),
        directoryName               (3),
        ediPartyName                (4),
        uniformResourceIdentifier   (5),
        iPAddress                   (6),
        registeredID                (7) }  (SIZE (1..MAX))
```

If present, the **permittedSubtrees** and **excludedSubtrees** components each specify one or more naming subtrees, each defined by the name of the root of the subtree and optionally, within that subtree, an area that is bounded by upper and/or lower levels. If **permittedSubtrees** is present, subject names within these subtrees are acceptable. If **excludedSubtrees** is present, any certificate issued by the subject CA or subsequent CAs in the certification path that has a subject name within these subtrees is unacceptable. If both **permittedSubtrees** and **excludedSubtrees** are present and the name spaces overlap, the exclusion statement takes precedence for names within that overlap. If neither permitted nor excluded subtrees are specified for a name form, then any name within that name form is acceptable. If **requiredNameForms** is present, all subsequent certificates in the certification path must include a name of at least one of the required name forms.

If **permittedSubtrees** is present, the following applies to all subsequent certificates in the path. If any certificate contains a subject name (in the **subject** field or **subjectAltNames** extension) of a name form for which permitted subtrees are specified, the name must fall within at least one of the specified subtrees. For example, assume that two permitted subtrees are specified, one for the DN name form and one for the rfc822 name form, no excluded subtrees are specified, and **requiredNameForms** is not present. A certificate that only contained a DN and where the DN is within the specified permitted subtree, would be acceptable. A certificate that contained both a DN and an rfc822 name and where only one of them is within its specified permitted subtree, would be unacceptable. A certificate that contained only names other than a DN or rfc822 name would also be acceptable.

If **excludedSubtrees** is present, any certificate issued by the subject CA or subsequent CAs in the certification path that has a subject name (in the **subject** field or **subjectAltNames** extension) within these subtrees is unacceptable. For example, assume that two excluded subtrees are specified, one for the DN name form and one for the rfc822 name form. A certificate that only contained a DN and where the DN is within the specified excluded subtree, would be unacceptable. A certificate that contained both a DN and

an rfc822 name and where at least one of them is within its specified excluded subtree, would be unacceptable.

When a certificate subject has multiple names of the same name form (including, in the case of the **directoryName** name form, the name in the subject field of the certificate if non-null) then all such names shall be tested for consistency with a name constraint of that name form.

If **requiredNameForms** is present, all subsequent certificates in the certification path must include a subject name of at least one of the required name forms.

Of the name forms available through the **GeneralName** type, only those name forms that have a well-defined hierarchical structure may be used in the **permittedSubtrees** and **excludedSubtrees** fields. The **directoryName** name form satisfies this requirement; when using this name form a naming subtree corresponds to a DIT subtree.

The **minimum** field specifies the upper bound of the area within the subtree. All names whose final name component is above the level specified are not contained within the area. A value of **minimum** equal to zero (the default) corresponds to the base, i.e. the top node of the subtree. For example, if **minimum** is set to one, then the naming subtree excludes the base node but includes subordinate nodes.

The **maximum** field specifies the lower bound of the area within the subtree. All names whose last component is below the level specified are not contained within the area. A value of **maximum** of zero corresponds to the base, i.e. the top of the subtree. An absent **maximum** component indicates that no lower limit should be imposed on the area within the subtree. For example, if **maximum** is set to one, then the naming subtree excludes all nodes except the subtree base and its immediate subordinates.

This extension may, at the option of the certificate issuer, be either critical or non-critical. It is recommended that it be flagged critical, otherwise a certificate user may not check that subsequent certificates in a certification path are located in the name space intended by the issuing CA.

Conformant implementations are not required to recognize all possible name forms.

If the extension is present and is flagged critical, a certificate-using implementation must recognize and process all name forms for which there is both a subtree specification (permitted or excluded) in the extension and a corresponding value in the **subject** field or **subjectAltNames** extension of any subsequent certificate in the certification path. If an unrecognized name form appears in both a subtree specification and a subsequent certificate, that certificate shall be handled as if an unrecognized critical extension was encountered. If any subject name in the certificate falls within an excluded subtree, the certificate is unacceptable. If a subtree is specified for a name form that is not contained in any subsequent certificate, that subtree can be ignored. If the **requiredNameForms** component specifies only unrecognized name forms, that certificate shall be handled as if an unrecognized critical extension was encountered. Otherwise, at least one of the recognized name forms must appear in all subsequent certificates in the path.

If the extension is present and is flagged non-critical and a certificate-using implementation does not recognize a name form used in any **base** component, then that subtree specification may be ignored. If the extension is flagged non-critical and any of the name forms specified in the **requiredNameForms** component are not recognized by the certificate-using implementation, then the certificate shall be treated as if the **requiredNameForms** component was absent.

In clause 12.4.3 (1997) and 10.3 (2000) add a new path processing variable as follows and renumber subsequent bullets accordingly:

d) *required-name-forms:* A (possibly empty) set of sets of name forms. For each set of name forms, every subsequent certificate must contain a name of one of the name forms in the set.

In clause 12.4.3 (1997) and 10.4 (2000) add a new initialization step as follows and renumber subsequent bullets accordingly:
d) Initialize the *required-name-forms* to an empty set;

In clause 12.4.3 (1997) and 10.5 (2000) add a step to the checks applied to all certificates as follows:

h) If the certificate is not an intermediate self-issued certificate, and if *required-name-forms* is not an empty set, for each set of name forms in *required-name-forms* check that there is a subject name in the certificate of one of the name forms in the set.

In clause 12.4.3 (1997) and 10.5 (2000) add a step to the constraint recording actions applied to intermediate certificates as follows:

c) If the **nameConstraints** extension with a **requiredNameForms** component is present in the certificate, set the *required-name-forms* variable to the union of its previous value and the set consisting of the set of name forms specified in the certificate extension. If the **requiredNameForms** component contains more than one name form, the *required-name-forms* variable shall signal that a name of at least one of the indicated name forms in this extension shall be present in all subsequent certificates. The union of a previous value of *the required-name-forms* variable with the value from the current certificate extension is a set of sets signalling requirements for all subsequent certificates. For example if the current *required-name-forms* is set to requiring that either a DN or an rfc822 name must be present in certificates and the current extension in the certificate being processed indicates that either rfc822 names or DNS names are required, the resulting union that is the new *required-name-forms* indicates that each of the subsequent certificates must have either an rfc822 name or both a DN and a DNS name.

In Annex A, ***certificateExtensions*** *module update the asn.1 for* ***nameConstraints*** *extension as above*

In Annex A, ***certificateExtensions*** *module add the following:*

id-ce-nameConstraint OBJECT IDENTIFIER ::= {id-ce 30 1}

In Annex A, ***certificateExtensions*** *module, delete the following*:

id-ce-nameConstraints OBJECT IDENTIFIER ::= {id-ce 30}

In Annex A, ***certificateExtensions*** *module, add the following to the set of OIDs not used in this specification:*

id-ce 30

12. Editor's Response:

Accepted solution proposed by source