Path Processing Issues

William E. Burr

31 March 1999

willian.burr@nist.gov

301-975-2914

Path Processing Standards

- X.509 (6/97)
  - section 12.4.3
  - not yet published
- RFC 2459
  - section 6.
  - ftp://ftp.isi.edu/in-notes/rfc2459.txt

Informal Meeting 23 March

- Bill Burr
- Tim Polk
- Santosh Chokhani
- Tim Moses
- Hoyt Kesterson (phone)

Path Processing Issues

- Where does path processing start?
  - PKIX RFC 2459 is different than X.509
- When does policy mapping occur?
  - X.509 does it wrong
  - RFC 2459 is ambiguous
- Where do parameters come from?
- Parameter Inheritance

Where Does Path Proc. Start?

- Trust anchor key only (X.509), or;
- Trust anchor certificate (RFC 2459)
  - Could initialize various state variables
    - » permitted and excluded subtrees
    - » mapping flag
    - » path length constraints
    - » authority constrained policy set
  - could allow different restrictions only on relying parties of “this CA”

Where Does Path Proc. Start?

- Starting with CA self-signed Cert. is questionable
  - will probably break some existing implementations
  - setting flags takes flexibility from applications, and
  - doesn't seem to be necessary
    - » can almost always do the same thing some other way

Parameters & Path Processsing

- Where do parameters come from?
  - three Algorithm ID fields that might hold them
- Right Answer:
  - Sub Public Key field of issuer cert.
    - » Chokhani 1996 NISSC paper
  - implicit not explicit in X.509
  - might be well to be explicit in path processing description

Parameter Inheritance

- Not in X.509 at all
- Discussed under DSS in RFC 2459
  - » might be well to show in path processing machine
- References
  - annex B of ISO CD-1578-2
  - MISSI has an explicit state machine
  - http://csrc.nist.gov/pki/twg/parameters/index.htm

Policy Mapping Issue

- Chokhani found problem (twg-99-15)
- Should include subject domain policy, not issuer domain policy in CA certs
  - example to follow
  - must map before policy checking
    - but X.509 checks before mapping
    - RFC 2459 is ambiguous

An Example of the Problem

User init. policy set = USHigh

Issuer:	**USA CA**
Subject:	**Friendly CA**
Cert Policy:	**USHigh**
Policy Map:	**FrnHigh = USHigh**
IPM skipcerts:	**0**
REP skipcerts:	**0**

Issuer:	**Friendly CA**
Subject:	**Lybia CA**
Cert Policy:	**FrnHigh**
Policy Map:	**LybHigh = FrnHigh**
IPM skipcerts:	**0**
REP skipcerts:	**0**

Issuer:	**Lybia CA**
Subject:	**Bad Guy**
Cert Policy:	***USHigh***

Note that policy mapping is inhibited and explicit policy required

The LybHigh = FrnHigh mapping has no effect on USA CA Relying Parties

But Lybia CA cheats and asserts USHigh Policy OID

The Problem

- Because policy mapping occurs after checking issuer must put a policy in his domain in CA cert.
  - therefore Friendly CA puts a policy in his domain (FrnHigh) in Lybia CA Cert
  - Lybia CA can scam US relying parties by asserting USHigh in certs

The Solution

User init. policy set = USHigh

Issuer:	**USA CA**
Subject:	**Friendly CA**
Cert Policy:	**FrnHigh**
Policy Map:	**FrnHigh = USHigh**
IPM skipcerts:	**0**
REP skipcerts:	**0**

Now we have subject policy (FrnHigh) in cert. mapped to USHigh (need change to map First)

Issuer:	**Friendly CA**
Subject:	**Lybia CA**
Cert Policy:	**LybHigh**
Policy Map:	**LybHigh = FrnHigh**
IPM skipcerts:	**0**
REP skipcerts:	**0**

And LybHigh in this cert. Since Mapping is now disabled LybHigh is not in acceptable Policy set & fail to validate

Issuer:	**Lybia CA**
Subject:	**Bad Guy**
Cert Policy:	**USHigh**

USA Relying Party never gets this far because cert. above fails

The Solution

- Do policy mapping before checking so issuer can put a policy in the subject's domain in CA cert.
  - Friendly CA puts a policy in Lybia CA's domain (LybHigh) in Lybia CA Cert
  - US RP doesn't recognize Friendly CA's mapping and rejects the cert. Friendly CA issues to Lybia CA, because it doesn't contain an acceptable policy

Recommendations

- Clarify PKIX to ensure that self-signed certs are not required, and that, if used, only the keys are used
- Revise PKIX and X.509 to do policy mapping before path processing
- Add parameter inheritance to X.509, and (possibly) describe in PKIX path processing