Implementing CertPath Validation: Lessons Learned

Steve Hanna
Sun Microsystems, Inc.
steve.hanna@sun.com

Cert Path Validation & Building

- Widely needed
  - S/MIME, TLS, IPsec, ...
- Very complex
  - son−of−2459 includes 18 certificate extensions
  - Validation requires 24 steps per certificate
  - Not including revocation and building!
  - Complexity causes bugs, security holes, and cost
  - But every feature’s required in some environment
- What can we do?

Possible Solutions

- Simplify the standards?
- Delegate Validation/Building to a Trusted Server
  - DPD/DPV, XKMS
- Grit your teeth and write the code
- Use a Library that supports Validation/Building
  - Getronics CML
- Use a Platform that supports Validation/Building
  - CertPath API in J2SE™ 1.4

CertPath API

- Standard API for CertPath building and validation
- Standardized through Java Community Process™
  - Expert group: Bluestone, DSTC, Entrust, IBM, Sun, Verisign
  - Currently Proposed Final Draft (JSR 55)
- Incorporated into Merlin (J2SE 1.4)
  - Will be included in future J2SE implementations
  - Beta 3 available from http://java.sun.com, FCS soon
  - Free, available for most OS's

CertPath API Features

- Read/write encoded CertPaths (e.g. PKCS#7)
- Validate CertPaths
- Build validated CertPaths
- Retrieve certs & CRLs from a directory
- Abstract API supports X.509 or non−X.509 certs. PKIX−specific API also included.
- Extensible. Can load multiple implementations, custom validation checks, and such at run time.

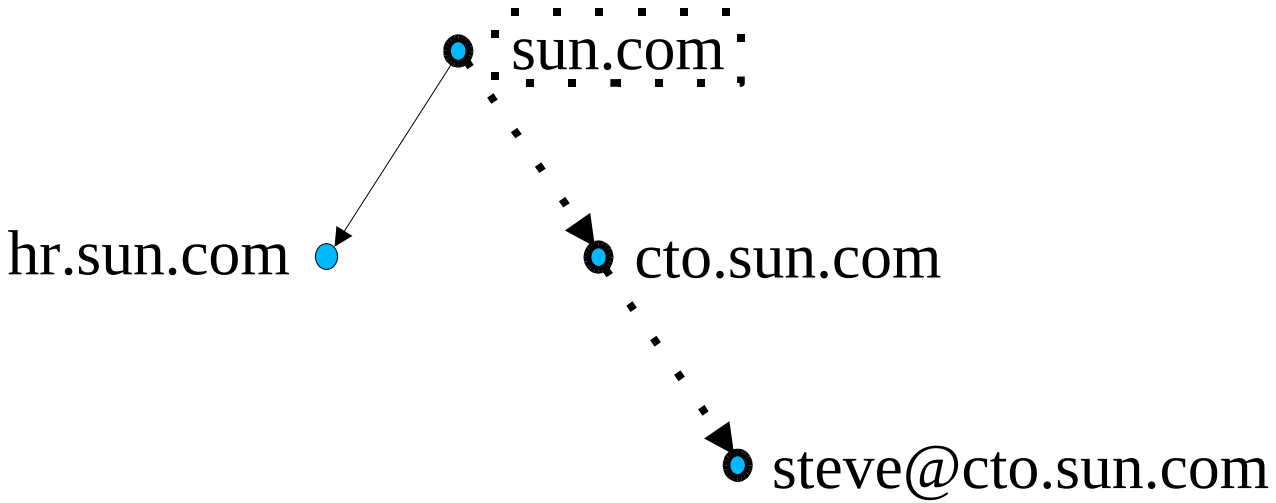
Features of Sun's Implementation

- Read/write encoded CertPaths
  - PkiPath or PKCS#7 formats
- Validate CertPaths
  - Compliant with draft−ietf−pkix−new−part1−08.txt
  - No support for CRL DP, SIA, or AIA extensions, DeltaCRL and IDP CRL extensions (all optional).
- Build validated CertPaths (more info later)
- Retrieve certs & CRLs from a directory
  - from LDAP directory or collection

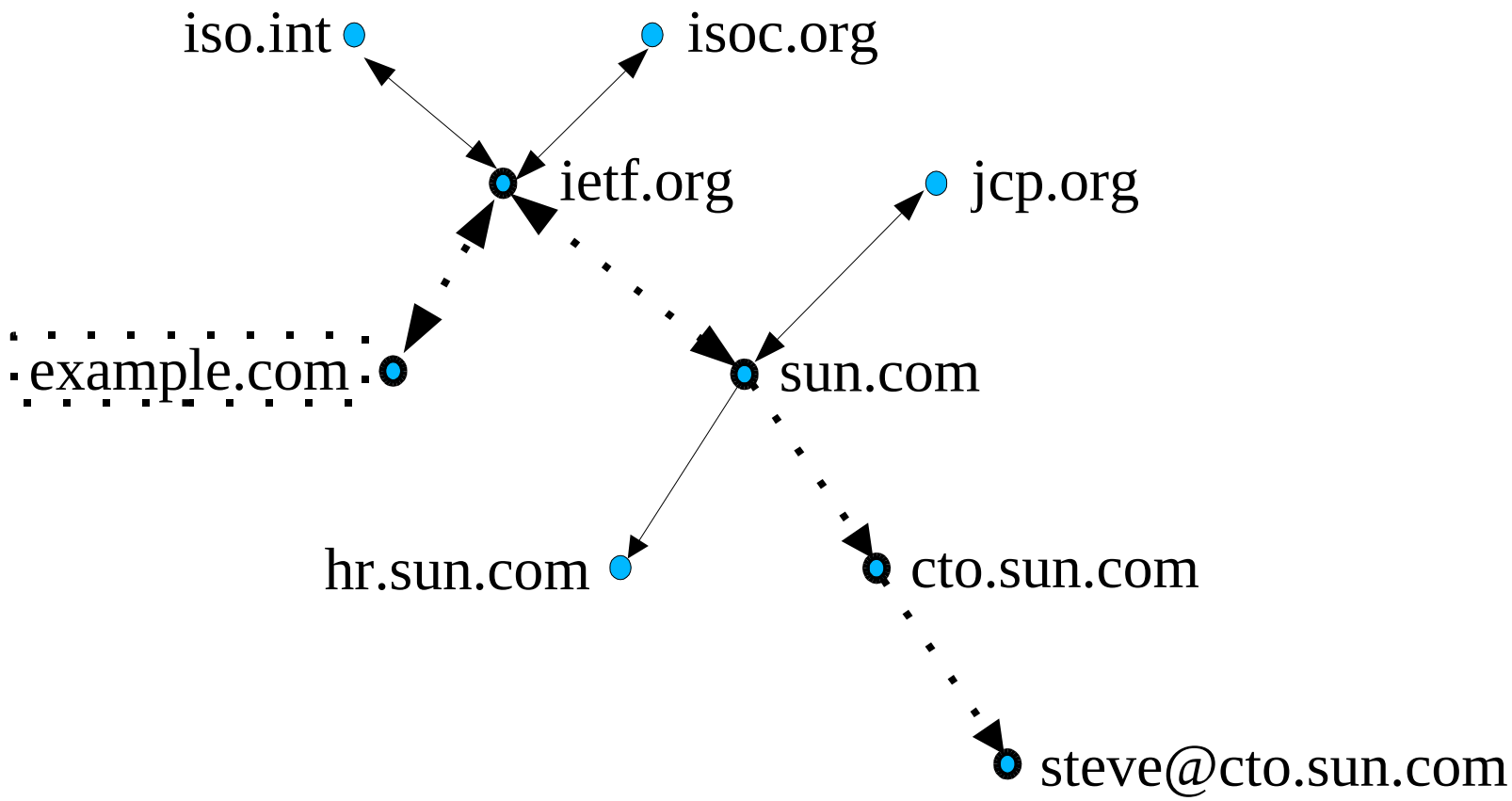
CertPath Building

- Little analysis to date
- Should you start with EE or TAs (trust anchors)?
  - Starting with EE (forward) better in hierarchy
  - Starting with TAs (reverse) better in other topologies
- Name constraints crucial in non−hierarchies
- Loops should be prohibited
- Self−signed certs should be ignored
- See NDSS ’01 paper for details

Hierarchical Topology



Non−Hierarchical Topology



Lessons Learned

- Implementation and interoperability testing provide surprising insights.
- PKIX validation and building works!
- You don’t have to implement it yourself. Use a library (like Getronics CML) or platform (like Java™) that implements it.

URLs and Q&A

- J2SE 1.4

  http://java.sun.com/j2se/1.4

- NDSS ’01 paper

  http://www.isoc.org/isoc/conferences/ndss/01/2001/papers/elley.pdf

- Getronics CML

  http://www.getronicsgov.com/hot/cml_home.htm