Registry & Directory Infrastructure

Overall Scenario

• Stanford is a highly decentralized enterprise with multiple, virtually autonomous, systems of record.

• People are a common, shared business object..
  – Personnel’s system: faculty & staff
  – Registrar’s system: students
  – Affiliated enterprises’ systems: SLAC, Hospital
  – Non-trivial number of variously-affiliated people who aren’t in any system

• Other objects: Groups, (network) Services
**Goals**

- Support general-purpose whitepages directory service via web-based and teletype user interfaces (UIs).
- Replace (please) our utterly byzantine, amalgamated, and high-latency data feed (next slide).
- Support off-the-shelf products, e.g. browsers with embedded LDAP-based address books, Mail clients, etc.
- Support access by both the Stanford community and the Internet-at-large.
- Push *responsibility* for information maintenance and visibility out to subjects.
- Support & tie-in & leverage-off-of authentication infrastructure -- “SUNet ID”.

Our Data-Feed Challenge

Personnel

?  SU Printed Directory  ?

SLAC Personnel DB

?  Communication Services OnLine Directory
(used by phone operators only, mainframe-based Spires database)

?  Some unknown, mysterious post-processing that apparently takes days.

SMTP (!)

WhoisDB

Whois service

Whois

Hospital Apps that sorta know about people.

Desktop Whois Client
Overall Problem Statement

- LDAP, and to a lesser degree X.500, are simply *protocols*.
- Though possessing rich informational and functional models, they don’t provide any capabilities on their own for, for example, expressing business rules.
  - ensuring any particular identifier is unique
    - [I’m using “identifier” here in terms of an arbitrary attribute value, *not* as a RDN value]
  - ensuring syntax of string-based attribute values
    - This is more an issue with LDAP, X.500 has provisions for server-based attribute syntax validation.
  - etc.
• Directories, as a class, are subtly different than general-purpose relational databases (RDBMSs)
  – RDBMSs have historically been the platform of choice for implementing repositories of data+business rules
Registry&Directory Infrastructure

Overall Problem Statement, cont’d

• RDBMS properties:
  – Strongly typed data
  – Can represent complex relationships
  – Transaction support
  – Support on-the-fly data view generation, aka “Join”
    • “find all the people whose managers are located in New York” and place result set in a new table for later use
  – Has notion of referential integrity
  – No open “on the wire” protocol standard
Registry & Directory Infrastructure

Overall Problem Statement, cont’d

• Directory properties
  – Strongly typed and structured information, like RDBMS
  – Object-oriented, hierarchical
  – Multi-vendor interoperability due to:
    • open standard access protocols
    • core standard schema
  – Extensible schema
  – Highly distributable
  – But no notion of “Join” or “report generation”, etc.
  – Notion of “referential integrity” not in protocol - implementation-dependent
Registry & Directory Infrastructure
Definitions and Scope

- Our mission definition for registries:
  - “A registry is a service that serves the needs of applications for coordinated maintenance of identity information about a class of business objects.”
  - E.g. Some classes are: People, services, groups.
- A registry is a transaction-oriented service...
  - Client applications will use one mostly to enter and update information, i.e. a registry is write&update-oriented.
  - Read-oriented access will typically be handled by other components of the overall system, e.g. the Directory.
The scope of our Registry is *enterprise-wide*. All people affiliated with the university *should* be in the Registry.

- I.e. if you need others within the enterprise to recognize your affiliation, you need to be in the Registry.

A primary materialization of this requirement:

- Needing an authentication principal - a SUNet ID
- Many network services are *authenticated*
  - E.g. AFS distributed file system, various web pages, distributed computing resources (e.g. POP-based email service)
  - Authentication infrastructure is Kerberos-based
• Registry information is disseminated to other network entities via the Directory
• Various applications utilize the Directory when they need information about people, e.g...
  – “@Stanford.edu” email routing
  – Web authentication
  – Authenticated Printing service
  – Dial-In Network Service
  – Whitepages (i.e. general purpose Directory)
  – HelpSU (“helps-you”) action-request system
  – Rudimentary Authorization Service
Overall Directory & Registry Infrastructure Dissemination

"Systems of Record"

Registrar
Personnel

Univid
University ID number system
SPIRES-based

SUNetID System

Stanford LDAP-OpenServer Gateway
(Maps Registry Schema into Directory Schema, plus is effective "Directory Business Rules" Implementation)

Desktop Clients

Kerberos DB

LDAP-based Directory Service

LDAP (R/W)

TDS

Middleware Event Broker
RDBMS-based

TDS

TDS

TDS

Leland Account DB
RDBMS-based

Registry
RDBMS-based

Network-based Applications

LDAP (Reads)

LDAP (Reads)

university ID number system SPIRES-based
Overall Directory & Registry Infrastructure Update

"Systems of Record"

Registrar Personnel

SUNetID System

Directory Service

SLOG

LDAP (Reads)

LDAP

Desksops
(Browsers)

Network-based Applications

Subject’s Desktop
(browser)

HTTP
(effectively authenticated writes)

Web-based User Interface for Data Maintenance

Registry

Middlewar e Event Broker

Univid

TDS

TDS

TDS

TDS

TDS
The Source/Registry/Directory Data Chain

Events:
univ : reconcile

1. Formerly "identity".
2. Lifestyle event for relationship as a whole, as distinct from changes to the data within.
3. Changes all affiliation data. Corresponds to control codes "affiliation/status", "affiliation/privilege", etc.
4.成果 only "Stanford academic".
5. Separate visibility event only made when there is no corresponding table change event.
6. Groups on id is helpful and work flows from stable and source service levels, so replaces source and group event.

(source = faculty | student | staff | hospital)

Source : matching (1)
Source : name (sourcetype) (2)
Source : relationship (sourcetype) (3)
Source : affiliation (sourcetype) (4)
Source : email (sourcetype) (5)
Source : group (4)
Source : visibility (where, what) (6)
univ : identifier
univ : group (6)

sunet : relationship
sunet : identifier
sunet : role
sunet : registry_service_extension (7)

[ sunetid : new
sunetid : ses
sunetid : service ]

leland : registry_service_extension

registry : new
registry : identity
registry : name (sourcetype)
registry : identifier (type)
registry : relationship
registry : group
registry : address (sourcetype)
registry : phone (sourcetype)
registry : email (sourcetype)
registry : url
registry : extension (9)
registry : visibility (where, what)
registry : service (10)
registry : data_exception (11)

Illus. & much Design Credit: Lynn McRae

04/21/99
Names & identifiers are subtly different beasts

**Identifiers** are unique

– A given identifier maps to one subject
– Subjects have multiple forms of identifiers
– E.g. “account form” -- 8 chars, alphanumeric
– “Long form” -- First.Last
– Some system-specific forms, e.g. Unix UID, DCE UUID, MS GUID, etc.

**Natural names** are both..

– Non-unique
– and *mutable* -- they can & do change

• Use the directory to map all the above to a subject
To: **user@Stanford.edu**

(Incoming Email Message)

**Email Routing**

1. **SMTP**
   - “Hmm.. A message for **user@stanford.edu**, lessee..” (#1)

2. **LDAP**
   - “Yo, Directory.. Please lookup “seasSunetId=**user**”, and return “mailDrop” (#2)

3. **LDAP**
   - “Ok, so the directory said: mailDrop=**user@host.stanford.edu** (#3),
   - So I’ll route that message to that address (#4)"

**Directory**

**Mail Server**

(MTA)

“user” on Host.stanford.edu
1. User entered “Doe” as a call-back name.

2. HelpSU calls directory for list of “Doe”’s and their contact info.

3. HelpSU builds a “pick list” in it’s UI using the directory info for “Doe”.

4. User picks person they really meant from the list, and it is entered into the action request.
Issues

- Implementing policy and to some extent policy machinery is a do-it-yourself proposition
  - E.g. No searches on first name
  - Administrative limits per requester role
- Implementing policy may reveal issues down to the standards level
  - E.g. specification of filter interpretation -- whitespaces in the face of substring matches
Registry & Directory Infrastructure
Summary

• The natures of Registries and Directories are subtly different
• X.500/LDAP-based directory services are not RDBMSs
• Makes sense to combine them into an overall system -- play on their strengths
• Project at Stanford is far from, if ever, “finished” -- will continually evolve
  – Present deployment effort is “Phase II”
  – Phase III will involve policy implementation in order to support off-the-shelf LDAP clients
Registry & Directory Infrastructure
Themes / Philosophies

• DNs are immutable and persistent
  – a DN is a primary key, yet another identifier.
  – DNs are not necessarily human-palatable.
  – “(Natural) Names” are another class of a subject’s attributes and aren’t primary keys in and of themselves.

• A Directory can’t do it all by itself
  – Has to have site-specific procedures & conventions wrapped around it.
  – E.g. how are subjects vetted and assigned their initial identifier?
  – E.g. selecting a unique identifier of some given form.
• How effectively user-oriented applications can leverage off of a directory infrastructure is *directly proportional* to how well-formed and well-specified the system’s notions of *identifiers & names* are.

• The currently prevalent directory access protocols, *in and of themselves*, are *not* “strong” authentication protocols.

• Directory technology is a key underlying enabler for Authorization Services (among lots of other possibilities).

• Like the “single-sign-on” notion morphing into “fewer-sign-ons”, the “single directory repository per administrative domain” notion should more realistically be “fewer repository/directories, with cleanly-crafted roles and data feeds”.
“Trust management” is an often used term, but perhaps a more relevant way to consider the notion is as its inverse:

– “Risk Management”.

Privacy is a huge, emerging issue.

– We all need to pay a lot more attention to it.

“Open is good, closed is bad”

– where protocols, and to a large extent even implementations, are concerned.

All the above is IMHO, of course.
Acknowledgements

• The Registry & Directory (recently subsumed into “The Horton Project”) team is comprised of (at least):
  – Booker Bense, Carol Farnsworth, Jill Fukuhara, Michael Hart, Jeff Hodges, Craig Jurney, John Klemm, Bill Lucker, Jeff Mapes, Danno McKinnon, Lynn McRae, Dennis Michael, RL “Bob” Morgan, Catherine Mulhall, Pat Nolan, Michael Puff, Dennis Rayer, Sandy Senti, Tim Torgenrud, Dwayne Virnau.

• Lynn McRae (leader/manager), Craig Jurney, Michael Puff, Michael Hart, with RL “Bob” Morgan (enterprise Architect) largely conceived of and implemented the Registry, with input from the team at-large.
Acknowledgements, cont’d

• Jeff Hodges (Tech leader), Danno McKinnon, and Tim Torgenrud (manager) configured and deployed the directory, designed the schema (with input from the Registry team folk), and designed and implemented the Access Control approach, again with overall input from the team at-large.

• RL “Bob” Morgan authored the initial drafts of the SUNetID concept and was generally responsible for evangelizing our (still emerging & evolving) concept of a single identifier namespace.
References

• This talk will be available at..
  – http://www.stanford.edu/people/hodges/talks/

• Selected References..
  – Stanford Registries & Directories pages..
    • http://www.stanford.edu/group/itss-ccs/project/registry/
    • http://www.stanford.edu/group/itss-ccs/project/registry/registries.html
    • http://www.stanford.edu/group/itss-ccs/project/sunetid/
    • http://www.stanford.edu/group/networking/directory/
  – Project Horton
    • http://www.stanford.edu/group/itss-ccs/project/horton/
  – SUNet ID
    • http://www.stanford.edu/group/itss-ccs/project/sunetid/
References, cont’d

– *Why do I need a Directory when I could use a Relational Database?* -- by Steve Kille  

– *Directory Services: DIT Design* -- James Rommel  

– *Risk Management is Where the Money Is* Dan Geer, CertCo  
  • http://www.stanford.edu/~hodges/doc/Geer-RiskManagement.txt