Requirements and Approaches for a Publicly Visible Persistent Identifier for Person Entries in the Stanford University Enterprise Directory Service

Jeff Hodges (NeuStar)
RL "Bob" Morgan (University of Washington)

Both formerly of Computer and Communication Services,
Stanford University

v1.1
original: 29-Jul-1998
revised: 23-Aug-2006

Abstract

Directory entries representing people in the Stanford University enterprise directory service have both non-unique names and unique identifiers, although not all entries have "palatable" unique identifiers. This presents issues when one is designing a user interface where queries may return result sets with more than one entry -- a user may need to view and utilize unique identifiers in order to refine such result sets. We analyze this need and identify various plausible solution options, given our naming and identification infrastructure. Two solution approaches are presented and their differentiating factors discussed. The approach we decided on and its detailed description closes the paper.

Executive Summary

Character-based user interfaces to directories need to supply the user with some identifier to use to refine searches that return multiple entries. This concept is explicitly embodied in the Whois protocol [Whois] and termed a *handle.* An entry's handle is a guaranteed-unique identifier for that entry. It is typically displayed in all query result sets, whether a set of one or many.

We are migrating our enterprise whitepages directory from Whois-based technology onto LDAP-based technology [LDAP]. We anticipate that many of the user interfaces (UIs) to the new LDAP-based enterprise directory service will be graphical (i.e. GUIs) in nature. However, character-based UIs will remain in use for the foreseeable future. For example, we intend to provide Whois-based access to the LDAP-based directory via a Whois-to-LDAP gateway.

Note that although one can argue that a common web-based GUI approach to displaying directory query result sets is as a simple textual list (perhaps tabularly arranged) in a subwindow, disambiguation between entries with identical names can be accomplished via rendering the entries using hyperlinks to further information, obviating the need for a disambiguating ancillary entry identifier. Since we plan to provide a

Whois-to-LDAP gateway, we need to provide some identifier for entries which serves the purposes that the Whois handle does.

Though, the Whois handle, as instantiated in our Whois-based directory, has been problematic for users in our experience. Our Whois handles are based on peoples' names and are used strictly within the context of the directory. Since other people with similar names may arrive at the university or people may change their names, one's Whois handle is not guaranteed persist over time, i.e. it may change. But users continually mistake the Whois handles for email addresses and are perturbed when they change without their knowledge. We believe the amount of user confusion and our on-going effort to dispel it justifies looking into the issues and seeing what we can do to resolve them while we're migrating to a new technology base.

This document analyzes the requirements for the "handle" concept, which herein is termed a Publicly Visible Persistent Identifier (PVPI), and presents two solution approaches. They are summarized as..

A1. Construct and utilize a guaranteed-unique form of Registered Name, i.e. a *"Uniquified" * Registered Name* (URN).

A2. Construct and utilize a unique, fixed-length, non-Registered-Name-based alphanumeric identifier.

The tradeoffs involved in utilizing either approach are subtle and varied. They are summarized in the section entitled *An Alternative Solution Scenario and Two Subsequent Approaches.* The *Bottom Line* section summarizes the quandary, and Our Chosen Approach specifies the solution. We recommend that readers read at least those three sections. Readers interested in the gory details are invited to begin with the following section.

* "Uniquified" was first coined by Tim Howes in a message to Jeff Hodges about how they rendered people's names unique in the UMich X.500 directory service. It means, essentially "to have been rendered unique".

Document Conventions

The key words "MUST", "SHOULD", and "MAY" used in this document are to be interpreted as described in [ReqsKeywords].

Definitions, Background, and Motivation

Currently, person entries in the Stanford University enterprise directory service have, as attributes, various forms of "natural names" and "general identifiers" [SUNetIDReqs, SUNetIDDesign]. First, we will outline the definitions of these terms (see the referenced documents for a more detailed discussion).

A *subject* is a named, often real-world, entity. A person, for example. A person entry in the directory is an entry with an objectclass of "person" [LDAPattributes] which we map to a real-world subject, i.e. a person. We accomplish such mapping through the use of subjects' names and identifiers. A *name* is a character string that may map to zero, one, or more subjects. A *natural name* is a name that is based on a subject's real-world name. An *identifier,* in contrast, is a name that maps to exactly one subject. An identifier may be based on a

subject's natural name, or it may be artificial -- a number or a bit string, for example.

We use different forms of names and identifiers in various situations depending upon contextual requirements. For example, a user interface (UI) to some application might prompt a user for their "name", and expect that most people will enter some form of their natural name. Another UI might prompt a user for an identifier and expect that people will enter one rather than their natural name. Though, there may be environments where one's natural name is treated as an identifier, but we explicitly do not encourage that approach in our environment. This is because although natural names typically map to individual subjects, occasionally some map to an additional number of subjects.

In our interwoven Stanford University Network Identifier [SUNetID], Person Registry [Registry], and Directory [Directory] environment we have defined the following types of names and identifiers...

- Registered Name -- the full, official, legal natural name a subject supplied in the act of becoming known to the university. Registered Names are often unique, but not always. Additionally, one's Registered Name may be unique within the directory at one point in time, yet not be unique at another point in time -- or vice versa.

- General Identifiers (GeneralIDs) -- a General ID is the base class of identifiers utilized in the SUNetID system. General IDs may take several forms...

    - Registered Name-based GeneralIDs

      There are both short- and long-form Registered Name-based GeneralIDs. Short forms are suitable as "login names" for most of the various currently-utilized types of operating systems (OSs) extant on the Stanford University Network (SUNet). Each form of a subject's Registered Name-based generalIDs must be unique across the entire General ID namespace, by the definition of an identifier.

    - Other General ID forms, e.g. a DCE Universal Unique Identifier (UUID).

      These forms aren't germane to this document's discussion because they aren't based on a subject's natural name.

    - All GeneralIDs are managed by the SUNet ID system, and thus are SUNet IDs.

Note that person entries in the directory will have these properties...

- *All* person entries in the directory will have an attribute containing the subject's (not-guaranteed-unique) Registered Name.

- *Not all* person entries in the directory will have a (guaranteed-unique) General ID attribute.

This document's topic is directly due to these properties. Here's a prime motivational example...

> Some UI scenarios for directory queries require selection by the user of particular entries from a set of returned, and often summarized, entries. Examples of this arise with "whois" and "finger" command line-based UI "frontends" to the directory. With these UIs, it is typically up to the user to enter an identifier from the first result set in order to obtain a complete view of a single entry . Other examples arise in the context of graphical UIs (GUIs) to the directory. Here, though, the GUI can hide from the user the unique identifier used to make the singular selection from the original result set.

Currently Recognized High-level Requirements

In the former example, where the user must view and potentially re-enter the identifier, there are the typical requirements for the identifier to be based upon one of the subject's names, to be of "reasonalble" length, or otherwise be "human palatable". E.g., a *long* alphanumeric string bearing no relation to the subject's natural name(s) is usually considered sub-optimal, i.e. "not terribly palatable/memorable/wieldable". However, for example, a short numeric string, similar to a drivers license number or a University ID number is considered by some to be acceptably palatable. Also, there are simple techniques for rendering a subject's Registered Name unique. The resultant length depends up the length of the subject's Registered Name, of course. This approach is also considered palatable by some. And there are some that consider the former or the latter or both to be unpalatable.

Another clear requirement we've derived over the years of running the Whois-based directory service is that subjects would prefer it if all things in their entries smacking of "names" or "identifiers" would only change if they perform some explicit action that reasonably causes them to change. E.g. by changing their Registered Name. This of course occurs "natuarally" through various life changes, e.g. matrimony.

The third requirement is one of consistency -- the style or form of the human palatable identifier should be the same for all person entries. Our experience suggests that user and subject confusion will result if there is not a consistent form of human palatable identifier across all person entries.

The fourth/final requirement is that this identifier must be visible to all directory clients whether they are authenticated and authorized or not -- i.e. it must be "publicly visible".

The Problem Statement

Thus, the problem statement containing these essential requirements is...

- What do we use as a publicly-visible, consistent, not arbitrarily changing, human palatable identifier for ***all*** person entries in the directory?

We term such an identifier simply a *publicly-visible persistent identifier* (PVPI), since it would be expressly intended to be used in any query from any requestor and would not *itself* be protected by an access control list. Note that the PVPI is essentially the same in concept as the "handle" utilized in the Whois protocol [Whois] and is typically used directly in user interfaces to Whois-based directory services [SUNetWhois].

Note that we're explicitly not using the term *handle* in this document in order to avoid confusion with the particular-to-Whois instantiation of this concept.

Detailed Technical Requirements

These are the detailed, nominal requirements that we feel define the problem space upon which the above expose' is based...

- A PVPI MUST be unique within the set of all PVPIs in the directory at PVPI instatiation time, and on into the future.

- An entry MUST have at most one PVPI.

- A PVPI MUST be reasonably persistent.

    E.g. no more than 2 changes per year per entry.

- A PVPI MUST be reasonably typable.

    E.g. less than 17 not-totally random characters containing no whitespace or control characters.

- PVPIs MUST be capable of being applied to any applicable class of entries in the directory, where applicable classes MAY be drawn from: classes and departments, if they are present in the directory and generally visible to ad-hoc users.

- Each applicable entry MUST have one.

- PVPIs MUST be generated without requiring input from the subject.

- A requestor MUST be allowed to retrieve the PVPI attribute if the requestor may legitimately retrieve any portion of the entry, given the requestors authorization level.

The below anciliary requirements help to further refine the problem space. Some of the requirements below are not finalized, they are the ones with options indicated in []'s with " | " separating the options. Finializing these requirements is part of refining the solution options, since these requirements essentially present tradeoffs in system behavior. The identified anciliary requirements are...

- PVPIs SHOULD be strongly persistent. I.e. the same value persists while the user is in the overall system though it MAY be changed at user request, if supported.

- PVPIs [MUST | SHOULD | MAY] [not] be reassigned. I.e. the PVPI for one person is [never | might be] used for another.

- PVPIs [MUST | SHOULD | MAY] [not] look similar to a person's name, even across name changes (so the PVPI is familiar). This requirement conflicts with the persistence requirement, and, for long names, it may conflict with length requirement.

- PVPIs SHOULD be easy to remember. I.e. they are name-like, or similar to some other attribute, e.g. uid (nee userid, neeSUNet ID), or simply short, e.g. 7 or 8 digits.

- PVPIs SHOULD work with command-line-oriented Whois- and Finger-based gateways to the directory. I.e. there's no embedded whitespace or UNIX/DOS shell metacharacters in the PVPIs.

- PVPIs [MUST | SHOULD | MAY] not have any conflicts with SUNet IDs. I.e. a PVPI can be the person's SUNet ID, but should never be someone else's SUNet ID.

- PVPIs SHOULD reuse existing identifiers if possible. There's no reason to invent new ID space for this modest purpose if another one will do, e.g. SUNet ID for those who have them.

- PVPIs SHOULD be consistent across all entries. This is so PVPI users will have a more consistent user experience.

- PVPIs SHOULD be unique across all commonly-searched attributes. E.g. "hodges" or "morgan" are a

poor choices since they match many people's last names -- and last names, as embodied in the surName (sn) attribute, will be commonly searched upon, as will commonName (cn) which contains fullnames and thus surnames.

- PVPIs SHOULD not be easily confused with other ID-oriented attributes. E.g. "whois handles" look like login IDs, but aren't. This can be quite confusing and a Bad Thing when someone's handle is someone else's ID. Additionally, an all numeric PVPI might be confused with a phone number, or a University ID number, or even a zip code.

Readily Apparent Solution Scenarios and their Issues

One potential approach is to copy what the Whois-based directory does. It has the concept of a unique "handle" which is entirely based on the equivalent of one's Registered Name. The handle is an algorithmically derived abbreviation of Registered Name and is guaranteed unique across all entries in the Whois-based directory at a given point in time. However, one's handle can change over time. Each time a new directory entry is created, collisions between the new entry's handle and other similarly-named entries' handles are resolved by adjusting the new entry's handle and possibly those of the similarly-named entries. Thus a subject's handle may change without the subject's knowledge. This clearly violates the requirement of having no arbitrary changes made to one's names or identifiers...

I1. Whois-style "handles" may arbitrarily change over time without the subject's input or approval.

The Registered Name-based identifiers supplied by the SUNet ID system would seem to be prime candidates to utilize as human palatable identifiers. However, not all entries in the directory have SUNet IDs, as discussed above. Thus only a subset of person entries will have a Registered Name-based identifier which might be presented in result sets. Currently, the only other identifiers in, or planned for inclusion in, all person entries are numerically-based and bear no relationship to a subject's natural name. Additionally, our default authorization posture is that we reveal current SUNet IDs only to previously authorized clients, e.g. authenticated users. This further limits our being able to utilize current SUNet ID-based identifiers as humanly palatable identifiers. Thus this scenario's issues are...

I2. Not all subjects have SUNet ID-based identifiers, and some of the rest will not have publicly revealed any of theirs.

A third potential solution would be to utilize whichever form of SUNet ID a subject has authorized to be publicly-visible, e.g. by authorizing it as an email alias [SEAS], or, if no publicly-visible SUNet ID form exists, utilize some other identifier that's based on Registered Name. This isn't consistent. Additionally, if one authorizes a SUNet ID as an email alias, where one wasn't authorized before, then it follows that one's human palatable identifier returned to clients should probably change. But this violates the no arbitrary changes requirement.

I3. Using SUNet ID-supplied, Registered Name-based identifiers for some entries and something else for others is inconsistent, plus one's human palatable identifier would be subject to arbirary change.

An Alternative Solution Scenario and Two Subsequent Approaches

An alternative solution is to create yet another form of General ID for all person entries and use it as the PVPI. The form of this identifier arguably hinges on this single question...

Should PVPIs be "name-like"?

If they *are* name-like, then...

- They must be alterable as a person's name changes.
- They should be alterable to meet the person's desire for what their name should look like.
- Their use and management overlaps so much with SUNet IDs that it is obvious just to make them *be* SUNet IDs, with the uniquified Registered Name as a default long-form SUNet ID.

Or, if they *are not* name-like, then..

- they should be short numbers, with..
- some distinguishing look so that they aren't likely to be confused with University ID numbers, phone numbers, Zip codes, or some other relatively pervasive, short number in our lives.

Both approaches objectively satisfy the primary PVPI requirements as long as we haven't specified a maximum-length requirement for PVPIs (that's less than the longest possible name which might enter the system).

In summary, the two approaches are..

A1. The *Guaranteed-unique Form of Registered Name* Approach

- Construct and utilize a guaranteed-unique form of Registered Name, i.e. a *Uniquified Registered Name* (URN).
- Form would be "Registered Name with a number at the end".
- The URN is managed as a SUNet ID to ensure no conflicts.
- The URN can be used just like any SUNet ID, e.g. as a SEAS alias.
- URNs will change automatically as a user's Registered Name changes.
- The user can choose to make the PVPI be some other SUNet ID.
- The PVPI can change at the user's request.
- A PVPI is non-reassignable.

A2. The *Fixed-length Non-Registered-Name-based Alphanumeric Identifier* Approach

- Construct and utilize a unique, fixed-length, non-Registered-Name-based alphanumeric identifier.
- Assigned by the Registry.
- Short, for ease of remembering and using.
- Alphanumeric, in order to avoid confusion with Univ IDs. Perhaps the alpha portion should be fixed in order to aid the point above. E.g. DS1234567
- The ID is permanent, non-re-assignable, and non-changeable.

The implications of each approach are discussed below.

Implications of Approach A1

The advantages of this approach are...

- It has an obvious relationship to the subject since it is based upon the subject's Registered Name.
- Arguably more familiar to current Whois users, since it would be similar to current Whois handles.
- More mnemonic in general.
- Those who wish could potentially reuse an existing SUNet ID form.

The disadvantages are...

- The length varies and is relatively unconstrained.
- The user is exposed to the management of the PVPI.
- There will be many more entries in the SUNet ID system.
- Possible confusion with auto-generated SUNet ID PVPIs mixed with user-chosen ones.
- Not consistent across all entries.

Implications of Approach A2

The advantages of this approach are...

- The length can be constrained.
- Consistent across all entries.
- Essentially no additional UI required in the overall Registry/SUNet ID/Directory system.
- No changes ever, period.

The disadvantages are...

- It will not necessarily have any obvious relationship to the subject's Registered Name. I.e. it is not mnemonic.
- Confusion due to people having yet-another-number to deal with.

The Bottom Line Decision...

To some degree the differences between the approaches center on aesthetics, however, the factor of constrained or relatively unconstrained identifier length could be a quantifiable tradeoff some situations, and the Registered Name-based approach has a quantifiable impact on the SUNet ID system.

A2 is much easier to implement, but may not be as palatable.

Though, can we afford to do A1; is it worth the effort?

In the next section, we discuss and specify the approach we decided to implement.

Our Chosen Approach

We decided to go with A2: the *Fixed-length Non-Registered-Name-based Alphanumeric Identifier* approach. We feel its advantages outweigh its disadvantages and that approach A1's cost to system developer/maintainers was too high given what we feel are not terribly great advantages over A2 from the user's perspective.

Here's the format we chose for the PVPI...

> DSnnnAnnn

Where...

> The PVPI is an alphanumeric string constructed according to these rules...

- "DS" represents a 2-chacter alphabetic, fixed upper case, substring, i.e. [A-Z], with the *constant value* of "DS" for now, but we *may* add other values in the future.

- "nnn" represents 3-digit numeric, i.e. [0-9],substrings, which are *randomly* assigned, rather than serially.

- "A" represents a single-character, fixed upper case, alphabetic substring, also with *randomly* assigned values, from the set [A-H, J-N, P-Z]. Note that "I" and "O" are excluded in order to eliminate potential confusion with "1" (numeral one) and "0" (numeral zero).

  (Please note that an [ABNF] specification for the PVPI is presented in Appendix A, below.)

For example...

> DS468J135

Our motivations for choosing this particular format are that it is...

- Entirely unlike a spoken or written "word",
- Largely unlike a phone number (at least in recent times, in this country), University ID number, Zip code, car license plate number (at least in California), etc.
- Relatively easy to remember,
- Fixed length,
- Scalable to millions, but not billions or more.

Conclusion

We did not yet have any operational experience with this approach at the time of the writing of the 1.0 version of this paper, and we both left Stanford shortly thereafter -- and we have not yet caught up with our colleagues there to see how it has worked in the intervening eight years. PVPIs were implemented in the SUNet Person Registry [Registry] and Enterprise Directory Service in Fall 1998 as a part of the rollout of the then new, LDAP-based enterprise identity, registry and directory services called "StanfordWho". PVPIs can be seen in action by, for example, using the following command (at a command prompt) on most any internet-connected *nix system: "whois -h whois.stanford.edu <some common family name>". Note that in the gratuitous whois help information returned, the PVPI is refered to as a "DS number".

Appendix A

This is the PVPI syntax expressed in [ABNF] form...

```
      pvIdent  = alphaTag numericTag alphaChar
                 numericTag

    alphaTag  = %x44 %x53          ; "DS"

    alphaTag  =/ alphaChar alphaChar ; may alloc
                 other values in

                                   ; the future from the
                 alphaChar set

   alphaChar  = %x41-48 / %x4A-4E / %x50-5A

                                   ; A-H, J-N, P-Z, I & O are
                 excluded

  numericTag = DIGIT DIGIT DIGIT    ; 0-9 0-9 0-9
```

References

[ABNF] *Augmented BNF for Syntax Specifications: ABNF*. D. Crocker, Ed., P. Overell. Internet Engineering Task Force, RFC 2234. November 1997. Available as: http://info.internet.isi.edu:80/in-notes/rfc/files/rfc2234.txt

[Directory] *Directory Services Project.* Stanford University. Jeff Hodges, Directory Services Lead Geek. Project established July 1994. In production as of May 1996 with on-going evolution. Info available at: http://www.stanford.edu/group/networking/directory/

[LDAP] *An LDAP Roadmap and FAQ*. Annotated bibliography of resources about the Lightweight Directory Access Protocol. Jeff Hodges. Periodically updated. Available at: http://www.stanford.edu/group/networking/directory/x50

0ldapfaq.html

[LDAPattributes] *A Summary of the X.500(96) User Schema for use with LDAPv3*. Mark Wahl. Internet Engineering Task Force, RFC 2256. Available as: http://info.internet.isi.edu:80/in-notes/rfc/files/rfc2256.txt

[Registry] *Registries Project*. Stanford University. The Registry Team. Project established ca. 1996. In production as of June 1998, with evolution on-going. Info available at: http://www.stanford.edu/group/itss-ccs/project/registry/

[ReqsKeywords] *Key Words for use in RFCs to Indicate Requirement Levels*. Scott Bradner. Internet Engineering Task Force, RFC 2119/Best Current Practice 14. Available as: http://info.internet.isi.edu:80/in-notes/rfc/files/rfc2119.txt

[SEAS] *Stanford Email Alias Service*. Stanford University. The SUNet ID Team. Project established ca. 1995. In production as of May 1996 with on-going evolution. Info available at: http://www.stanford.edu/group/itss/services/sunetid /alias-info.html

[SUNetID] *Stanford University Network Identity System*. Stanford University. The SUNet ID Team. Project established ca. 1995. In production as of May 1996 with on-going evolution.

[SUNetIDDesign] *Stanford University Network Identity System: Design*. RL "Bob" Morgan, Stanford University. 1-March-1996. Available as: http://www.stanford.edu/group/itss-ccs/project/sunetid /sunetid.design/sunetid.design

[SUNetIDReqs] *Stanford University Network Identity System: Scope and Requirements*.
RL "Bob" Morgan, Stanford University. 1-March-1996. Available as: http://www.stanford.edu/group/itss-ccs/project/sunetid/sunetid.design/sunetid.requirements

[SUNetWhois] *SUNet Whois-based Directory Service*. Stanford University. Originally by the Systems and Technology team of Networking Systems. In production since near the beginning of time. Available at: http://www.stanford.edu/cgi-bin/whois2html/

[Whois] *Nicname/Whois*. K. Harrenstien, M.K. Stahl, E.J. Feinler. Internet Engineering Task Force, RFC 954. October 1985. Available as: http://info.internet.isi.edu:80/in-notes/rfc/files/rfc954.txt