Best Practices in User ID Formation

The variety of naming conventions and the resulting volume of user IDs needed to access IT resources is a user and security administration nightmare. A global naming convention is key to solving this problem.

Many enterprises are considering the use of a global user ID naming convention to help with user-to-ID correlation, to promote end-user "goodwill" and to achieve cost savings through the reduction of help desk calls caused by forgotten ID/passwords.

There is little value in a stand-alone project that converts user IDs to a single format because the effort doesn't generate a return on investment by itself. It is not needed for single sign-on or password management because single-sign-on products keep track of all ID/password combinations for each user, thereby alleviating the need for the user to remember them (except for his or her initial logon credentials). Password management projects that include password synchronization reduce the problem of forgotten passwords to levels easily managed by the enterprise. However, if the enterprise is considering a user-provisioning project, it is common for a global user naming convention to be included in this project so that in the long term, its benefits can be realized for new users and applications.

However, an enterprise is unlikely to achieve a single global user ID format if it has a heterogeneous technology environment because operating system (OS), database and application vendors have not yet formed consensus on what the ID should be — they likely will never do so. Although a user-provisioning solution can automatically generate an ID for each user and then use this ID for new users across multiple target systems, it is limited by what formats makes sense on each target system: Each has its own rules for valid formats.

Enterprises should not consider user IDs to be confidential information. IDs are used in too many "public" ways — for example, sending messages, printed on company reports and, if tied to the employee ID, for paging an employee over the intercom. Both a user ID and an independent authentication

Gartner

© 2003 Gartner, Inc. and/or its Affiliates. All Rights Reserved. Reproduction of this publication in any form without prior written permission is forbidden. The information contained herein has been obtained from sources believed to be reliable. Gartner disclaims all warranties as to the accuracy, completeness or adequacy of such information. Gartner shall have no liability for errors, omissions or inadequacies in the information contained herein or for interpretations thereof. The reader assumes sole responsibility for the selection of these materials to achieve its intended results. The opinions expressed herein are subject to change without notice.
mechanism are needed for robust access control. The user asserts an identity by entering an ID and verifies that identity by entering a password while using a token or supplying a biometric sample.

**User ID Background**

A user identifier (ID) is a character string that is used in a system to uniquely identify a user. Some systems call it a user name or username, which is somewhat confusing because other systems use that label for the data item that holds the user's given name. (Formally, this is the user's *common name*.)

Some systems use a single unique ID for each user. Others use different "external" and "internal" identifiers. External identifiers are used to logon. Internal identifiers are used by the system. Either or both of these identifiers will be unique, and one or both can be used to assign resource access, ownership and other activities. The distinctions between these and the user's common name are sometimes blurred. For example:

- **IBM z/OS Resource Access Control Facility (RACF)** — Only one user ID is used for logon and internally (for example, in access control lists), such as G63812E or T6AA. RACF holds the user's common name in a data field called *username* or *pgmrname* (programmer name), but this is only an attribute.

- **Unix OSs** — There are two user IDs, the *username* (a short character string, traditionally based on the user's common name) and a *user identification number*; for example, halffreeman and 501. The username is used for logon, but either ID can be used for file permissions (for example, chown -RP halffreeman * and chown -RP 501 * are equivalent). Both are assigned by an administrator. User identification numbers can be nonunique, but the OS regards two usernames with the same ID as being aliases for the same user. Many Unix OSs support the user's "full name" or "real name" — for example *Hal Freeman* — and some (for example, Apple Computer's Mac OS X) allow the user to login with this name.

- **Microsoft Windows 2000** — There are many user IDs for each user, both external and internal. The *user principal name* is the preferred logon name for Windows 2000 users and is unique with an Active Directory forest. It is an Internet-style login name for the user based on Internet Engineering Task Force RFC 822. By convention, it maps to the user's e-mail name; thus, the user only needs to remember a single name. The user principal name comprises the *user account name* (or *user logon name*) and a Domain Name System
domain name joined by an "@" — for example, susan.conran@example.com. Windows 2000 also can use Security Accounts Manager (SAM) account names for backward compatibility with Windows NT. Active Directory generates a 128-bit globally unique identifier when a user is created. This is unique across the enterprise — and anywhere else. A user's globally unique identifier is never changed. Each user also has a security identifier that is issued by, and unique within, a Windows 2000 domain. Windows 2000 inserts the user's security identifier into an access token to identify the user in all subsequent security interactions — for example, S-1-5-21-917267712-1342860078-1792151419-513. Windows 2000 also holds the user's common name in separate first name and last name data fields.

Creating a Good User ID

An ideal user ID will be:

- **Unique** — At least within an administrative domain (that is, any single "image" of a given system), but preferably across the enterprise. User IDs also should be unique over time — that is, a user ID should not be reused by another person within a period shorter than the retention period for audit records. Reuse can expose "ghost" authorizations and erode the forensic value of audit trails.

- **Consistent** — User ID formats should not indicate the employment status or role of the users.

- **Persistent** — A user should use the same ID throughout his or her lifetime in the enterprise, regardless of change in employment status.

- **Memorable** — A user, especially one with multiple IDs across multiple target systems, should not be overburdened with hard-to-remember IDs.

There are four basic formats for user IDs:

- The user's given name or something derived from it
- An arbitrary or sequential number of alphanumeric strings
- A structured ID that concatenates a role (for example, "sysadm") or organizational unit codes with either of the previous two formats
- A hybrid of the first two formats

A growing number of enterprises enable end users to create their own IDs. Typically, this procedure is allowed for lower-risk customer environments.
Name-Based User IDs

Examples for users with the common names Edgar Rice Burroughs, Catherine Lucille Moore and Clark Ashton Smith:

- E-mail address style — edgar.burroughs, cl.moore, clarkashton.smith
- Last name, one or more initials — burroughser, moorecl, smithca
- First initial, first seven letters of last name — eburroug, cmoore, csmith
- First name, initial letter of last name — edagrb, cathm, clarks
- Initials — erb, clm, cas
- Nicknames or informal names — erb, cathy, smitty

Any of these formats may be combined with a DNS domain name to create an RFC-822-style name — erb@example.com, cl.moore@example.com, csmith@example.com. However, if a domain name includes an organizational unit, it will turn the name into a structured ID hybrid.

Strengths

- The ID is easy for a user to remember.
- It is easy to recognize the user on reports, audit logs and other documentation.
- The user’s privileges are not implied.

Limitations

- The enterprise must deal with duplicates effectively. Duplicates will arise in schemes for common names. They will be more common the more abbreviated the user ID is. (Thus, only small companies can use the first initial and the first seven letters of the last name, or the first name and initial letter of the last name, as the user ID.) This usually means adding an increasing suffix to the ID — for example, jsmith2, jsmith3 and so on.
- The enterprise must address changes of name. Depending on the system, this likely requires creating a clone of the old user account, with all of its attributes and privileges, and perhaps copying files (for example, in Unix OSs, from one user/username/directory to another), then deleting the original.
- It may be difficult to avoid ID reuse.
Arbitrary User IDs

Examples include a sequentially assigned number, such as 63812, which may be followed by a check-digit for validation (for example, 63812E). This may be a payroll or other HR-assigned employee number.

Strengths

- It satisfies a principle of data analysis: Any ID should carry no information about the object it identifies.
- The user's privileges are not implied.
- It can easily provide a unique ID for authorization and for audit trails.

Limitations

- It may be difficult for users to remember.
- If it is an employee number, the enterprise must decide what contractors and other temporary staff will use, or what will happen if a temporary worker assumes a permanent position.

Structured User IDs

Examples include:

- Functional — "sysadm3" for a systems administrator
- Hierarchical — department code | section code | initials | distinguishing number (for example, ITADJS2)

Strength

- The user's position or function with the organization is clear.

Limitations

- Creates a huge administrative burden for frequently changing employee roles within the organization, as well as re-organizations.
- User privileges may be implied, giving attackers a target for an attack.
- It may be difficult to avoid ID reuse.

Hybrid User IDs

Examples include initials | employee number (for example, AA17792).
Strengths/Limitations: Varied, but generally hybridization emphasizes the limitations, rather than the strengths, of the base formats.

Types of User IDs to Avoid

Enterprises should not use government-assigned numbers, such as driver's license numbers, as unique personal IDs for their operations. These numbers are easily obtained by attackers, increasing the risk of identity theft, "economic crimes" such as money laundering and identity fraud, terrorism and other crimes.

Specifically in the United States, enterprise use of Social Security Numbers (SSNs) is being discouraged:

- The Federal Trade Commission, state-level consumer advocates and many representatives in the U.S. Congress are dedicated to stopping the indiscriminate use of SSNs as the United States' standard individual identification numbers.

- The Social Security Administration, through its Inspector General, has repeatedly called for restrictions on the private and governmental use of SSNs. This includes restrictions on the sale or public display of SSNs by governmental agencies, and prohibitions of private-sector sales, purchases or display of SSNs, except for the confidential use of credit headers. The Social Security Administration maintains that the refusal of a business to conduct trade without the receipt of an SSN is an unfair, deceptive practice and should be forbidden.

Which Format Is Best?

There is no single best format. There is a worst, however: The structured user ID format has gotten enterprises into trouble when a user moves within the organization or changes employment status, especially when access control rules are explicitly tied to the ID. For a sign-on ID, usability considerations generally will take precedence. Thus, the user's common name or a derivative format is better.

Key Facts: Enterprises looking to uniquely identify their users should:

- Avoid user IDs based on organizational unit codes, roles or externally assigned identifiers (for example, SSNs).

- Evaluate the cost-effectiveness of converting user IDs to a new format, unless there are compelling privacy concerns that need to be addressed (for example, migrating from SSNs).
Where an enterprise has an opportunity to introduce a new user ID format, it should use:

- User IDs that are based on users' common names when they will be used for sign-on or messaging (for example, e-mail).
- Arbitrary (random or sequential) user IDs when these IDs will be used only internally.

Where usage can't be separated, enterprises must determine which format will provide the lowest total overhead in administration and user support. With increasing automation (for example, user provisioning tools), the usability benefits of the common-name-based format become more easily measured.

**Bottom Line:** Enterprise policy must require a unique ID for every user — internal and external — so that personal accountability can be assured in business processes. Users must be informed that their logon credentials are used to authenticate them personally, and that all activity transacted using those credentials will be attributed to them. Enterprises must ease the complexity of the authentication process by establishing a common naming convention for the unique IDs.