The following paper was originally presented at the
Seventh System Administration Conference (LISA '93)
Monterey, California, November, 1993

Simplifying System Administration Tasks: The UAMS Approach

Roland J. Stolfa
Oklahoma State University

For more information about USENIX Association contact:

1. Phone: 510 528-8649
2. FAX: 510 548-5738
3. Email: office@usenix.org
4. WWW URL: http://www.usenix.org

Simplifying System Administration Tasks: The UAMS Approach

Roland J. Stolfa – Oklahoma State University

ABSTRACT

The User Account Management System (UAMS) is an extension of the original User DataBase (UDB) system presented at the USENIX Large Installation System Administration Conference in 1990. This paper describes the extensions of the UDB system from a single administrative entity tool for a distributed set of computers to a multidepartmental system over the period of three years since the first paper. It also covers the added features that have been developed, such as support for Novell networks and POP clients.

Introduction

In the university environment, a computer system administrator's job can be quite diverse. For instance, here at Oklahoma State University in the Computer Science Department, there have been times when two system administrators have been called upon to manage over 40 hosts, each with a separate password file, backup requirement, and operating system type. In addition, these same two individuals were required to field questions from over eight hundred users on these 40 hosts as well as the other hosts run all over campus.

As was made clear very early on, this situation had to be improved upon. The first and foremost area of concern was in improving the generation of accounts. At first, user accounts were stream lined to be a simple prefix, followed by a set of digits. An example would be "fs" for "file structures" followed by the numbers one through the number of students in the class. One problem with this was that we found several students having several accounts on the same system for purposes of one semester's class work. Further, pinning down exactly which student was associated with which account was not a very easy task: a task made more important with the attachment of the OSU campus to the Internet.

In short, there was no real solution to the user account creation, deletion, and management problems for a university. This situation lead to the original development of UDB, The User DataBase System, in 1987. Further refinements lead to the presentation on UDB at the Large Installation Systems Administration Conference in 1990 (available as OSU-CS-TR-90-04)[1].

After the first paper on UDB, several other colleges and departments within OSU decided to participate in the types of services UDB was providing. However, to extend UDB to that domain would have required all departments that wanted to participate share one database on one host. For various practical reasons, a distributed solution had to be found. The OSU Computer Science Department found itself in need of a system of providing database services to a majority of the campus. A system was needed that could maintain a campus wide flat name space, for both logins, or Universal Computing Identifiers (UCI) as we call them, and numeric user ids (NUID), allowing separate administrative entities access to only those pieces of information that directly relate to their organization. Due to the development of UDB and the extensive number of hours spent developing the code, a conscious effort was put forth to extend UDB to meet the new challenges. The remainder of this paper discusses some of the extensions to support this effort. Several other account maintenance systems were presented at the 1990 LISA conference. Some of their high points are summarized below.

In ACMAINT[2], a central database was presented to allow a single system administrator to manage computer account creation across a heterogeneous set of computers. However, it utilizes daemon programs running on both the server and the clients that rely on TCP/IP networking facilities. Although these facilities are gaining popularity here at Oklahoma State University, not all hosts have TCP/IP, hence this approach was not usable here.

In GAUD[3], a central database is accessed by many hosts over Remote Procedure Call (RPC) protocol to allow access by the various offices that might allow or deny access to a particular user to a particular machine. However, GAUD suffers from the reliance on the source code to the operating system, an item not all universities have. Furthermore, here at OSU, RPC is not available on all hosts, hence its unsuitability.

In newu[4], a functionality that is already in UDB was described (i.e., the ability to create and delete accounts on a foreign host). It suffers from the same problem UDB had, in that it only worked within one administrative entity.

In Uniqname[5], a system of merging existing accounts with a 'global view' is presented. As UDB

started with a unified 'global view', Uniqname offered a solution to something that was not a problem in our case. In addition, it presents a solution to a problem that UDB did not even attempt to address, that of preferred mail box address.

Data Analysis

An analysis of the data involved in account creation was undertaken to determine the easiest way to modify UDB to address the issues raised while trying to support the majority of the campus. This left an understanding of which fields of the database needed to be 'global' and which ones could be 'local' to the administrative UAMS site.

Global Data Fields

Most of the global data fields are overwritten in the process of receiving a new data feed from the registrar. The UCI and NUID are the exceptions in that they are generated only the first time the given user's record is entered into the system.

Student/Faculty identification number (ID)
Student/Faculty ID card issue number (ISSUE)
Student/Faculty full name (FULLNAME)
Universal Computing Identifier (UCI)
Preferred Numeric User ID for NFS (NUID)
Student department affiliation (MAJOR)
Automatic rights (AUTO)

Figure 1: Global Fields of the Database

The data analysis concluded that all slave UAMS sites across campus would have to share some part of the the global database maintained on the master site. The slave UAMS sites would treat this data as read only, allowing the master UAMS site to overwrite these fields at will. The global fields are treated read only on the master UAMS site after the initial creation of the users record. This is because the entire list of UCIs and NUIDs are kept unique on the master UAMS site. Once generated uniquely on the master site, these global data fields can be transmitted to any of the slave UAMS sites while still guaranteeing the data integrity (i.e., no duplicate UCIs or NUIDs).

Local Data Fields

The local data fields are unique to each administrative UAMS site. This allows each UAMS site to have control over the special case users without infringing on any other UAMS site.

Clear text initial password of user (PASSWD)
Encrypted initial password of user (EPASSWD)
Manually granted rights (GRANTED)
Comment field (COMMENT)
Last update time of this record (LUPDATE)

Figure 2: Departmental Fields of the Database

One consequence of this splitting of each record is that each administration would be able to set the default initial password, while maintaining the same UCI. This would help maintain some level of security between UDB hosts. This arrangement disables one user, knowing their UCI on one UDB administered host, from logging in ad-hoc to other UDB administered hosts based purely on the knowledge of the original password. It also prevents an individual user's account information from being of much use to someone else, as the initial password would be different between UDB administrations.

Each administrative UAMS site would also have a separate and unique GRANTED right field for each user in their system. This allows each site to specify unique (and possibly conflicting) URIs, or Universal Rights Identifiers as we call them, to give access to different clients (hosts, etc.). On the master UAMS site, the GRANTED right field is also used to select those special case users, like 'root', that need to go to a slave UAMS site in addition to those users destined to go to the slave site because of enrollment information. However, as the GRANTED right field on the master UAMS site does not go with the record to the slave site, the slave UAMS site never sees that URI.

Another result of this data analysis is that each department is allowed their own comment field (COMMENT) for each user. That way, any comments on a user are held in the confidence of the commenting department.

Transport Methods Analysis

Some method had to be found to get the parts of the database distributed amongst the various UDB sites. As previously mentioned, RPC could not be used because some hosts did not have it. Some hosts, although fewer than in the past, lacked TCP/IP, so things like a socket based transfer protocol were out. This left the original UDB's solution of lowest common denominator, email.

Although email has served UDB quite well for a number of years, it suffers the same security risks as any other information exchange media. With a good understanding of this, we were forced into using it to communicate with these lowest common denominator systems. We have attempted to make it as secure as possible, but more from a data integrity stand point. This was viewed as very important in assuring that the data destined for a particular site is the correct data for that site.

So for our installation, the original choice was once again validated. The transport mechanism for sharing the data between the various UAMS sites would be email.

Sharing the Data

In order to facilitate the sharing of data, a simple master/slave model was chosen. This represents the administrative association of the 'global fields' of the UAMS databases amongst each other. Hence there is a master UAMS site. This is the site that

receives the enrollment data from the registrar. It is also the only UAMS site that can definitively assign UCIs and NUIDs. Whenever any slave UAMS site needs a UCI and NUID for a user new to it, it must defer to the master UAMS site for the definitive information. The master UAMS site also maintains the master copy of the enrollment data for all students enrolled in classes that all of participating UAMS departments have authorized the master UAMS site access to (i.e., all the classes that the participating departments teach).

The master/slave model allows the master UAMS site to select all of the records for a slave UAMS easily. This is typically based on enrollment data held in the AUTO and MAJOR records (for students). Additionally, to select all non-standard records (for such things as 'root', 'uucp', faculty, etc.), a specialized GRANTED right, unique to that slave UAMS departmental administration, is used.

To provide each participating department autonomy over their users, only the 'global fields' are shared between the different UAMS sites. This implies that any URI given to a user on the master system as a GRANTED right is not propagated to any other UAMS slave site. This includes the specialized GRANTED right, as the GRANTED field is not in the 'global fields' list of data being shared. This also allows each departmental UAMS to have overlapping (and possibly duplicated) GRANTED right URIs. Further, it allows each departmental UAMS to use the COMMENT field as they see fit, without having to conform to some standardization scheme.

The Server/Client Model

A server/client relationship exists between any UAMS site and a system that is administered by the owner of the UAMS server. In this system, any server, either a master UAMS or a slave UAMS, may provide a data feed to a client system. The format of the data delivered to the client is client specific, and is typically used to generate some end product specific to that client. As way of example, the rest of this section will discuss the generic UNIX client system.

```
...
uci:epasswd:fullname:granted:major:auto:nuid
...
```

Figure 4: Sample datafeed for UNIX client system

In this model, all URIs are propagated to all client systems (along with the information needed to create /etc/passwd and /etc/group) from their administering UAMS server to allow the generation of the URI database for each host. This allows each departmental machine to make use of all of the enrollment data, the major code, and all of that departmental UAMS server's unique URIs for its own ends. One of the uses of the URIs that has been implemented here at OSU is the ability to run various programs (similar to access control lists). Another is the automated maintenance of mailing lists.

Final Design Criteria

There have been several ideas presented so far that guided the design process of UAMS. Among the most important were:

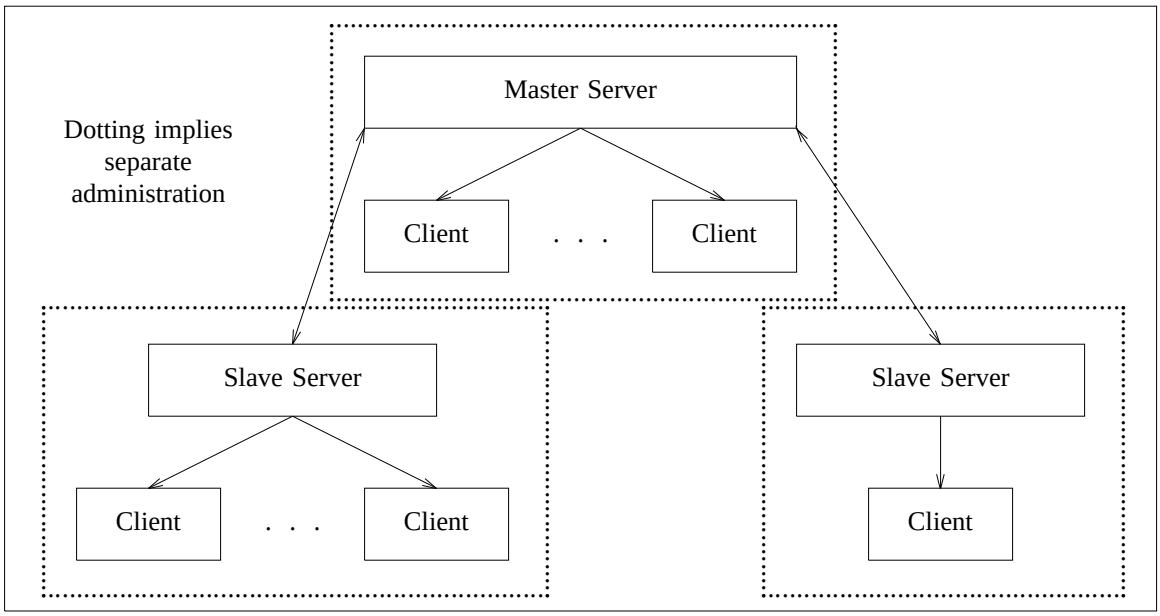

Figure 3: Master/Slave and Server/Client Relationships

- Centralized master copy of data base for the main unchanging parts of the database.
- The entire package should work with mail. No special use of or dependence on TCP/IP or RPC was allowed because not all hosts on campus have such capabilities.
- The system should, as did UDB before it, work with all the UNIX utilities without having to change said same utilities. This is because we do not have source code for all of the hosts on campus.
- There should be no long-running daemons in the system.
- The system should not use any special operating system specific code, nor any commercial product (such as a commercial database package). This is because for the vast number of platforms that it would have to support, the inherent cost would be prohibitive.
- As we want to distribute the resulting system, no AT&T derived source code was to be included in the system.

The UAMS/UDB System

The User Account Management System (UAMS) is an extension of the original UDB system to encompass all of the changes discussed in the previous sections while adding several additional new capabilities. Some of the modifications are listed below:

The Master to Slave Data Feed

The original UDB used a selection list, called the Rights Access File (RAF), to select which URIs (a combination of AUTO, MAJOR, and GRANTED field's contents) granted you access onto a particular system. A simple extension of this was used in UAMS to select out those rights that granted a record passage to a slave UAMS site. As it turned out, the only additional functionality was a simple wildcarding facility so that such choices as 'MATH*' (to select all classes taught under the 'MATH' heading) could be made. These selected records were then sent through a filter and mailed to the destination UDB.

```
...
id:issue:fullname:auto:major:uci:nuid
...
```

Figure 5: Sample master to slave data feed

Special Case Users

To handle the special case users, such as 'root', 'uucp', et.al., several things had to be overcome. First, these users did not have OSUIDs, any enrollment data, and quite seldom a FULLNAME. To cover these cases, as well as the case of the occasional guest account (or odd software package) that did not have an OSUID, a simple hueristic was formed of taking the proposed UCI (say 'root') and using a '+' prepended to the UCI as the OSUID. Thus 'root' would have the OSUID of '+root'.

This simplifies some areas of system administration. For instance, all of the users with plus-records ('+root' would be a plus-record) have no OSU student or staff id. Therefore they need not be selected for loading into our card key lock software[1]. Also, they are typically what we call 'mechanical accounts', i.e., they come with an operating system and do not have a physical person behind them. So when we scan the password file for accounts to set no-login, we can use UAMS as an aid in this process (this is not an enforced function of UAMS, merely an example of using the data UAMS maintains).

Anti-Rights

As in most any other university setting, students will be students. As UAMS was applied to an ever larger body of students, it was inevitable that a student would need to be kicked off of a machine due to an infraction of the rules. Shortly before this became necessary here at OSU, I had developed the anti-right. With this granted right, a user could be excluded from a machine, regardless of their other rights.

Let's say the user 'foo' is to be kicked off of the machine 'A'. However, currently 'foo' is granted access to that machine due to either a class enrollment right (in an AUTO field in this user's record), or their MAJOR (if they are a guest on that machine, indicated by a URI in this user's GRANTED field of, say, 'A', the granted right 'A' is simply removed). To delete all those rights would be unreasonable. First off, the next time UAMS received this user's enrollment data from the central university data base, the AUTO field enrollment right would return. At this time, the MAJOR code would be restored also. As these are not solutions, the anti-right was developed. In the case of 'foo', a GRANTED anti-right would be given as '!A'. In fact, if we just wanted to lock 'foo' out for a specified period of time, an expiration date could be added, giving an expiring anti-right of '!A-YYYYMMDD'.

These anti-rights do nothing abnormal to the record of the user. Instead, they alter the list of users selected to go to a site, 'A' in this case, to exclude this user. If this same user has some other granted rights, for instance this user has an account on the same department's POP server, their POP server rights are unaffected.

Comment Fields

As with any major software project, oversights are pointed out as soon as the code is delivered.

[1] As described in [1], the Computer Science Department runs a card key lock system on several labs within the department.

One of the other departmental UAMS administrators found a good use for a comment field, but discovered my oversight in not having one. So I enhanced UAMS to have a per user comment field.

This field is not automatically filled in, or in fact created, for every user. Therefore it was implemented as a sparse record within the database, similar to the AUTO and GRANTED fields records within the database.

New Novell client

The Novell version 3.11 client came about because of one of the other departmental UAMS administrators. Within the other department, Novell was used to link together several personal computers within a student lab. However, all of these computers had to be configured for users, with much the same information, just as the UNIX hosts that UAMS already served. As this is just another type of host, using unique login names and passwords, a new client was written to provide the information.

After researching the Novell manuals and considering the options, the client was written to generate a data file for the Novell user administration program 'MAKEUSER'. The UNIX side would keep a list of the users currently authorized for a Novell site, compare that with what UAMS was giving it, and generate add and delete commands as necessary. This preserved the feel of the UNIX client, without having to write a program for a personal computer.

New POP Client

The Post Office Protocol (POP) client came about because there was an interest within several departments to provide mail to personal computers. The POP system, as distributed with the RAND Corp. MH mailer, was chosen by some of them to fill this need. Again, this system had to be configured for users, just as the UNIX hosts did. As POP uses a password file that is in many ways similar to the UNIX password file, this client required several minor changes to one of the existing clients, and it was up and running.

New Administration Interface

After some time of using UDB, I found myself facing a problem. UDB had no real command line interface mechanism. This proved to be a hassle when I wanted to change the name of a particular URI to all those users who were granted it. In addition, I was trying to lure some DEC VAX/VMS system administrators at the time to use UDB. As a result I wrote a simple command line interface for UAMS, similar to DEC VAX/VMS AUTHORIZE. It operates on a single field of a single user's record at a time strictly from the command line.

Benefits

The UAMS package here at OSU has been operational for about three years now, with UDB operating for about three years before that. The system currently supports over 15,000 user account records across three colleges. They are split approximately as follows (the remainder are holdovers between semesters):

- 5500 on a collection of Silicon Graphics, Sun Sparc, Sun 3/60, and AT&T PC in the College of Architecture, Engineering, and Technology.
- 1200 on 26 IBM RS/6000 in the College of Architecture, Engineering, and Technology.
- 140 users on seven Sun Microsystems computers in the Department of Agricultural Engineering.
- 150 users on 12 Sun Microsystems computers and 3000 users of a PC/Novell network in the Mathematics Department.
- 1200 users on a Sequent Symmetry S-81 computer in the Computer Science Department and 700 users of a card key lock system.

Listed below are some of the benefits we have seen as a result of UAMS.

Uniqueness

When a user first enters the UAMS system, they are given a unique login. This allows such things as backups and mail to be uniquely identified across campus with a user. Since this information is keyed to their Oklahoma State University id card, even years hence, this student will be able to be uniquely identified and their files retrieved with a good level of certainty that they are being retrieved for the correct person.

Start of Semester Crush

At the start of each semester, there is a large enrollment influx of new users, mostly students. To create all of these accounts by hand would be impossible. With UAMS, to add an entire class of students to a machine is as simple as adding three lines to a configuration file, running a new enrollment database through the system, and installing the resulting password file. This entire process can take as little as 30 minutes.

Mail Lists

In the past, different instructors have tried to keep track of which students are in their class for purposes of a class mail list. UAMS automates this procedure and makes sure that the list is correct, up to the last enrollment data feed.

Interdepartmental Data Sharing

In the past if a user wanted to have the same data in two accounts, their only real options were along the lines of mailing the data between the two accounts. With UAMS in place on all of the hosts involved, it is possible now to automount the data

across campus. This is a direct result of having the UCI and NUID the same for each user on any UAMS administered host.

Conclusion

UAMS offers system administrators in a distributed departmental environment a unified environment for the administration of users. It does so without infringing on the local department's internal organization while offering support for quite a number of different clients (UNIX, Novell, POP, etc.). UAMS is quite extensible to any environment where a simple UCI, password, and or numeric user id is needed.

Credits

I would like to thank Mark Vasoll (Computer Science Department) for helping proof read this paper and for giving me inspiration on various sticky bits. I would also like to thank Rod McAbee (College of Engineering, Architecture and Technology), Russ Smith (Mathematics Department), and Clay Couger (Agricultural Engineering Department) for their interested support.

Author Information

Roland Stolfa is a Software Specialist on the staff of the Computer Science Department of Oklahoma State University. His interests include network simulation, robotics, aeronautics, and system administration automation. Mr. Stolfa has worked for the university since 1986. Reach him via electronic mail at rjs@a.cs.okstate.edu. His U.S. mail address is Computer Science Department; 219 Mathematical Sciences Building; Oklahoma State University; Stillwater, OK 74078.

References

[1] R. Stolfa and M. Vasoll, "UDB – User Data Base System," in *USENIX Conference Proceedings, Large Installation Systems Administration IV*, Colorado Springs, October, 1990, pp. 11-15.

[2] D. Curry, S. Kimery, K. De La Croix, and J. Schwab, "ACMAINT: An Account Creation and Maintenance System for Distributed UNIX Systems," in *USENIX Conference Proceedings, Large Installation Systems Administration IV*, Colorado Springs, October, 1990, pp. 1-9.

[3] M. Urban, "GAUD: RAND's Group and User Database" in *USENIX Conference Proceedings, Large Installation Systems Administration IV*, Colorado Springs, October, 1990, pp. 17-22.

[4] S. Schaefer and S. Vemulakonda, "newu: Multi-host User Setup," in *USENIX Conference Proceedings, Large Installation Systems Administration IV*, Colorado Springs, October, 1990, pp. 23-26.

[5] W. Doster, Y. Leong, and S. Mattson, "Uniqname Overview," in *USENIX Conference Proceedings, Large Installation Systems Administration IV*, Colorado Springs, October, 1990, pp. 27-35.