Issues in User Account Management in Academic Environments

Brajesh Pande

Senior Computer Engineer
Computer Centre, Indian Institute of Technology Kanpur, India

<brajesh@iitk.ac.in>

ABSTRACT

Academic environments have unique needs of user account management. Most of the user accounts in this environment are created at the beginning of a term or an academic year though some accounts may be created at any time. Some sort of physical identity verification is also necessary before the creation of a user's account. Once these accounts have been created, passwords may be distributed on machines based on internal policies. An account may also be temporarily suspended due to several reasons. It may be necessary to record and distribute such suspension information. It may also be required to segregate user accounts based on departments and designations. Finally several users leave at the end of an academic session and requests about extension of the time for which the user account remain active may have to be entertained. Though the typical account lifespan extends to the duration of the academic programme for a student, several accounts are created for much smaller durations and several accounts have a larger time span in the password file. Handling these varying time spans for which a user's account is active is also necessary.

Managing the password file in an effective manner so that this can be accomplished is necessary. This paper deals with several such issues in managing user accounts that arose at the computer centre of the Indian Institute of Technology, Kanpur. Methods and strategies for solving these issues and the solutions developed to implement these strategies are also discussed. These set of utilities greatly improved performance efficiency in various scenarios. Developing and implementing these scripts also freed the system administrators as several of these scripts were later handed over to other personnel.

1. Introduction

The computer centre at IIT Kanpur is a central facility for its students, staff and faculty with 4500+ users. The centre has several state-of-the-art systems such as SUN Enterprise 10000, SGI Origin 3400, SUN Enterprise 3500, IBM SP, PARAM 10000, SUN Enterprise 450, Compaq ES40, IBM RS600, HP L3000, several HP-B1000 machines, more than 300 Linux/Windows PCs and work stations. The user files are served via an NFS server over all these machines and UNIX users are authenticated using NIS. The centre supports an institute wide 6000 point, 100 Mbps, fiber-optic backbone and 10/100 Mbps UTP accesses. Network based services such as email and internet connections via proxies are provided through several servers. The servers that provide these services do not use the NIS services directly for reasons of security and high availability. These servers periodically get the master password file from the NIS server.

User accounts were previously created based on a procedure that required responsible staff for entering details of the user who requisitioned for an account at the centre. The password file was also made available for use on servers providing web based authenticated services. Suspension and deletion of users was done manually and sometimes through ad hoc scripts. However, every year the strength of new users entering the institute increased and this method became inadequate. Moreover, increasing demands were placed on the centre to segregate users based on some criteria such as department. This resulted in a pressing need for a better solution to handle the following major problems:

1. Mass intake and exodus of users
2. Demands for segregation of users into smaller groups
3. On demand suspension of accounts, and
4. Distribution of subsets of user accounts on different servers for finer grained access policy implementation or for higher availability of such servers.

The solution had also to be constrained to be a NIS based implementation so as to minimize

any possibility of ripple effects and to keep serving machines that relied on services provided by such a solution but were outside the domain of the Computer Centre. This paper deals with these issues that arise in managing the user accounts of such a facility.

1.1 Choices for Managing User Accounts

Several approaches [1-4] can be envisaged for user account management. Firstly vendor based tools can be used. However, such tools are too simplistic and tedious to cater to bulk needs of user addition and deletion in academic environments. The system proposed by Finke [1] is an elaborate database driven system that could meet many of our needs. However, this system was not chosen because of its extra details and because of reliance on data base systems. It was thought that this type of an implementation would incorporate an additional point of failure in the authentication process, and would create additional difficulties during upgrade and migration processes as well have an added cost. Also, such a system would not seamlessly integrate with the NIS repositories. Augus [3] is another system aimed at providing such services. However, this system is too generalized in the sense that it tries to take into account multi platforms and as a result delegats the selection of "`user id`" and other such task to sub systems that are loosely specified and depends on an external database for the same. Recently LDAP and Kerberos based systems [2,4] have come up but these systems are either too complex in that they also cater to service management, which is not required, or depend at some layer of implementation on databases. Smart card based solutions could also solve our problems, however, at the time of deciding our options, they were still an emerging technology and would also entail revamping our entire infrastructure - a cost that we were not willing to bear.

The only recourse left in designing the required solution was to have several mixed shell and C scripts with appropriate locking mechanisms to handle the password file directly. Though some issues raised herein can also be tackled by using shadow files, however, the shadow file looses its relevance when served through an NIS server. Also, the shadow file does not take care of all the issues that need be tackled by such a management system. Shadow files capture password aging information and there is no information that can be kept in such files for segregating users into groups based on arbitrary classifications. The solution that is suggested is based on categorizing users of the computer centre, judiciously fixing and designing fields of the password file to capture such categorization and other pieces of information, creating scripts and processes for accessing user entries of the password file based on these fields and listing the password file based on these fields.

1.2 User Categorization

It is necessary to discuss the user categorization that is followed at the computer centre before we embark on a discussion of user account management. A user of the centre is categorized based on the following criteria:

Department - There are more than 35 departments (including administrative units) and any classification on this basis helps in making department-wise mailing and other lists

User type - More than 20 user types have been identified. These types include undergraduate students, graduate students, faculty, staff, medical officer etc. Sub typing of user types has been avoided to keep the classifications simple and easy to use.

Account type - a user can have an account of three types - viz. a mailing account, an account that also allows access to the internet and includes mailing facilities and a shell account that includes the previous two facilities and access to various machines for submitting compute and other jobs.

Such a classification serves many purposes. Segregation based on departments and user types facilitates different kinds of file sharing. Lists and password files can be made for distributing to other machines and other departments for use in automation services. User data and file quota management can become streamlined and provide fine grained control on the basis of individual user, user type, departments and account type. The same classification also helps in controlling quotas on proxy servers. As an example assume that faculty and graduate students require more downloads from the proxy server and more space on the file servers and use a different proxy server than the other users. A classification based on the above scheme immediately helps in identifying these set of users within all users.

2. User Information and the Password File

The basic idea of user account management at the computer centre is to use the password file itself for storing several pieces of information

about the user. The password file is distributed to several machines using NIS (network information service) at the computer centre. Some servers use different mechanisms. NIS is used at the centre for historic reasons.

A traditional entry in the password file is of the form

```
username:EncryptedPassword:uid:
gid:GECOS:HomeDirectory:Shell
```

This file consists of colon separated fields in a line. Each line denotes an entry for a user account. Here the "uid" and "gid" are both numbers. The GECOS field is traditionally used to record information about the user in comma separated fields. Some comma separated fields can be used by the "finger" daemon to generate more information about the user. The GECOS field can be changed by the "chfn" command but it is at the discretion of the system administrator to allow the use of this function. Usually one would not want to open up this function as it might result in totally undesired GECOS fields that would not be of further use.

2.1 Fixing Fields of the Password File for User Management

Fields of the password file are fixed to capture user categorization and other pieces of information. We use the "gid" field of the password file to capture the user categorization. Each department is given a two digit number and each user type is given a two digit number. Concatenation of these two numbers captures the classification of the user. Another idea is to use the GECOS field to store the information necessary for managing the user accounts. The combined use of the GECOS field and the "gid" field gives us an effective way of managing the user accounts without additional complications of having databases.

We keep the following structure of the GECOS field

```
UserName utype dept,Phone,
Address,,,DateCreated,ValidTill,
Index
```

Here "UserName" is the full name of the user, "utype" is the user type and "dept" is department as discussed in the previous section. This by default denotes a shell account unless "utype" happens to be either mail or web and "dept" happens to be "nosh". The "DateCreated" and "ValidTill" fields are stored as a number which is a concatenation of the year, month, and day and for all practical purposes 20991231 is considered as infinity. This format is chosen because it lends itself to correct manipulation through the "sort" command. The last field is a unique "Index" for each account and can be the roll number or the personal file number or a social security identification or if all else is inapplicable, a unique number generated by the machine.

3. Managing and Distributing the Password File

Once the GECOS field is fixed all that is required are utilities for managing and distributing the password file. We divide these utilities into five sets. The first set of utilities deal with user account creation, the second set deals with removal and suspension of user account, the third set accesses and manipulates selected entries of the GECOS fields, the fourth set of utilities prune the password file based on some criteria such as "gid" patterns, and the fifth set of utilities are hosted on machines that want to get the password file from the NIS server without using the network information services.

3.1 Utilities for Creation of User Accounts

Some history of user account creation at the centre is necessary to appreciate the currently implemented method of creating a user's account. User account creation used to entail the following steps:

1. The user would fill up a form indicating details of the department, password desired etc.
2. The user would get the form countersigned by some competent authority allowing the use of the centre's facilities. This also served as a physical identity verification of the user.
3. Responsible staff would run a script for entering the details. This script added a new user account to the password file also choosing a "login id" for the user.
4. The user was conveyed the "login id" generated.

This scenario served the centre well enough when user accounts where sporadically created and the number of users was less. Over the years not only did the number of users increase but also the number of users that had to be created within a very short period increased. This was because of an increase in the student intake at IIT Kanpur. The only solution in the changed circumstances seemed to be that of increasing the number of people running the script for entering new students.

After some more deliberations and feed back from the persons involved in the user account creation process it dawned on us that the bottle

neck in the entire scenario was the form filling and the entering of details. If somehow we could decrease the time spent in this we could come up with a better and faster method of creating user accounts in bulk. Such a solution should have also incorporated physical verification of the user's identity. Luckily enough the computation environment of IIT Kanpur was amenable to a solution of the problem.

If we observe the scenario outlined it is evident that details of a user are entered twice, once on the form and once by the person running the script for adding the user. If somehow this entry could be done once and that too in parallel, a lot of effort and time would be saved, especially in the case of bulk entries. We decided to go ahead with this and mandated that the user be able to enter details on any of the PCs or workstations available at the centre. This was the first step of the user account creation process. Once the data of the user had been entered the identity of the user had to be verified before allowing the user access to the facilities of the centre. This was done by a staff who just had to choose between yes and no about the details entered by the user. This the staff did with the help of identity cards provided to the user. If at any point the staff chose a no for an answer the user would re-enter the details.

The full solution is enumerated below:

1. Create a "`login id`" say "`givelogin`" without a password that can run scripts for the user to enter details.
2. Check the "`login id`" requested by the user against existing "`login ids`" and against all enabled "`login ids`" within these scripts. Encrypt and store user passwords during this process.
3. Enable a user on verification of identity against the details entered by the user and the user's identity card. In case of "`login id`" conflicts chose a different "`login id`". If details entered are wrong ask user to re-enter details.
4. Add user accounts to the password file through a "`cronjob`" that also does additional chores like creating user directories and setting file quotas etc. We allow the NIS server to create user's files on the NFS server. This task could have also been delegated to the NFS server.

In the solution outlined it should be noted that the choice of the "`login id`" now rests with the user. Since the user is entering details on NIS enabled machines the choice of the "`login id`" can be checked against the already taken "`login ids`". However, since a user does not have the status enabled unless actually enabled by a staff after physical verification, we do not also check the "`login id`" chosen against all "`login ids`" chosen. This prevents the possibility of denial of service by filling up desired login names. In the rare case that a "`login id`" chosen conflicts with a "`login id`" already enabled the staff can change the "`login id`". Another advantage is that passwords are also known only to the user. In the solution opted we chose to have the actual addition of the enabled users to the password file through a "`cronjob`" run the same night as the day of enabling of the user. It should be noted that the password file as well as the file that contains the data of the enabled users is appropriately locked before accessing these files.

When we implemented this solution we could create all accounts of new students in a few hours as compared to a couple of weeks in the previous case and that too with less manpower. Though we also web-enabled this solution we still prefer to use the script based solution. The difference between the user account creation scenario at the centre and mailing services is that we require a strict checking of the identity of the user. Moreover passwords are only known to the user. Also in the new solution the older method of user creation still exists to take care of sporadically entered users. In our academic environment this method becomes necessary and has proved successful when a large number of students enter the Institute.

3.2 Utilities for Suspension and Removal of User Accounts

Suspension of user accounts at the centre is done due to several reasons. These could be because the user is going on leave or as a result of disciplinary action against the user. Suspension of the account involves changing the encrypted password field of the user's entry in the password file to an arbitrary string. The "`shell`" entry of the account in the password file is similarly mangled to prevent any access to network protocols. When the account is reinstated these entries are reversed. Also an account of the fines at any stage is maintained. A history of disciplinary actions is maintained to check on habitual defaulters. If a user account is suspended several times the level of financial liability may be increased.

Several files are used for helping in this process. Firstly there is a file - **reason** - that pre-assigns a mapping of the reason for suspension to financial liability, the new password string and the new shell and selecting the reason for

suspension from this file. The structure of the file is shown below:

```
reasonnum:reason:mangled
password:mangled shell:level of
suspension
```

Here "reasonnum" is the menu choice entered through the script that is used for disabling the account, "reason" is a string displayed by the script, "mangled password" and "mangled shell" entries are strings that contain the "reason" for suspension and are substituted for the actual encrypted password and the shell.

Secondly there is the file - **level** - that is used for mapping the "level of suspension" to actual liability in terms of financial recoveries and time periods of suspension. Certain levels may have no time periods and financial liability except a visit to the computer centre as in the case of users on leave. Where as some levels can incur varying suspension periods and financial liability. The structure of this file is shown below:

```
level:period:financial liability
```

Finally there is the file - **misuse**. When an account is suspended the "login id" to be suspended and the "reasonnum" is supplied through the script for suspending the account. Knowing this the following entry is recorded:

```
login id:Index:reason:by:date:old
shell:old password:period:fine
```

Where "login id" is the user's login, "Index" is the Index for that account in the GECOS field and "reason" is the reason for suspension as obtained from the reason file. The other fields are also self explanatory. Users can query this file to find out the details of their suspension suspended. A restricted shell account that runs the query script is made available for this. The suspension is accomplished by the script "disablelid" and the reverse is done by the script "enableid".

There are some accounts that have to be only partially suspended. Partially suspended accounts usually do not belong to individuals. Such accounts usually belong to various titular offices that do not require web usage. When partial suspension is requested the shell and password are not changed. Rather an entry of the "login id" is made in a separate file - **noweb**. This entry is used by the pruning utilities to form password files that do not contain those "login ids" that are in the file - **noweb**.

There is a special class of user accounts that are suspended because they should no longer use the services of the computer centre. Such user accounts are marked with special reason for suspension called "TermOver". Any user account marked "TermOver" is not enabled again. Usually such users are identified at the time they obtain a no dues certificate from the centre. A script is duly provided for giving no dues and marking these accounts to term over. However, several such users require a few weeks of extension of usage of computer facilities. Thus no dues cerificate is given by setting the "ValidTill" field of the GECOS entry in the password files appropriately. A "cronjob" marks those user accounts whose "ValidTill" field has expired - "TermOver". This script provides a grace period to all such users. Finally a script "RmTermOver" removes all users whose entry is marked "TermOver" from the password file and deletes their home directories.

3.3 Accessing and Manipulating GECOS fields

There are several scripts that manipulate the GECOS field of the password entry. These are listed below for completeness and some readers may find their functionality useful.

FindIndex: finds the index of a user account given the "login id"

FindName: finds the login id of a person given the index in the GECOS field

FindDate: finds the date till which a user account is valid.

SetDate: sets the date till which a user account is valid.

FindExpired: lists all accounts that have expired.

NoDues: used to find if a user has any financial liability. If the user has financial liability no dues certificate is not issued till that liability is cleared. Though this script accesses the files used in suspending a user account as discussed above, it also sets the "ValidTill" field of the GECOS entry.

3.4 Pruning Utilities

The formatting of the "gid" and the GECOS fields of the password file and the various files generated and used in the suspension process allow us to write several utilities for pruning the password file as needed. We discuss the same with a couple of examples.

Partial suspension requires that "login ids" existing in the file noweb be removed from the password file. This can be accomplished by the one liner

```
sed '1,$s/^/^/' noweb | sed '1,$s/
$/:/' | grep -v - passwordfile >
newpass
```

The file - newpass can be later fetched by the proxy server for its use.

Suppose we wanted to extract all under graduate students from the password file. Assume that the two digits in the "gid" are 18 and the user type is the last two digits of the four digit "gid". Then the following can list all the "login ids" of under graduate students

```
grep "^[^:]*:[^:]*:[^:]*:[1-9][0-9]18:" passwordfile | cut -d: -f1
```

Several such utilities and more complicated ones have been implemented.

3.5 Distributing the Password file

The password file is maintained on a Linux server that runs a NIS (network information service) server. Machines that do not use the NIS get the password file provided via "ftp". An example "ftp script", which runs on these servers, is reproduced below. This script has been stripped of error checking.

```
#!/bin/bash
# Gets the group and password
# files from nis
# File names group.master and
# passwd.master
ftp -i -n << !!
open nis
user dummy DummysPassword
get group.master
get passwd.q passwd.master
!!
```

This script uses the user name dummy and to get the files it requires from the NIS master server.

3.6 Locking Files

Several utilities and scripts outlined above may concurrently access important files like the password file or the misuse file. Locking mechanisms at appropriate places are necessary so that the password file data does not get corrupted in case the scripts are used concurrently. In large computer centres it is highly likely that there will be concurrency involved. The construct "lockfile" which is a part of the procmail package is used for acquiring a lock on a file before accessing the important file. First we try to acquire a lock on some file. The file on which the lock is actually acquired by password management routines is "/etc/.pwd.lock". If we successfully acquire that lock we access or modify the data necessary. When we finish we remove the file on which the lock has been acquired. This construct was found very useful in maintaining the integrity of various files used. This construct is available on command line on Linux machines. A similar C function is "lckpwdf".

4. Delegating Responsibilities

Several features were built into all the scripts before they were delegated to the operators in the computer centre. Firstly every script that modified files in any way kept detailed logs of each transaction carried out on its use. Such detailed logs can help in error recovery if ever required. They also help in maintaining accountability once a script is open to use by several people. Secondly, every script trapped the various signals received and did extensive clean up on receiving these signals. Such cleaning up prevented files used for locking from remaining in the system in case of interrupts from the keyboard. In case of system crashes in the middle of using these scripts - obviously a rare occurrence - the only recourse was to delete the locked file manually and resort to error recovery. Finally it was mandated that only an authenticated user be allowed to use these scripts via "sudo" mechanisms. Each individual who could access the NIS server was assigned to a group and "sudo" permissions were given on an individual / group basis. This added an additional level of security when the scripts were delegated as the root password could now be protected. Provisions were also made to backup the important files on a daily basis to prevent loss of such files.

5. Conclusion

This paper has pointed out several issues in managing the user accounts of a computer centre in an academic environment. Such environments have to deal efficiently with a sudden intake and exodus of users. Such environment may share various resources according to the type of user account created. The time period for which a user is entitled to such a centre also varies with bulk and sporadic exodus observed. User accounts have to be effectively managed in such situations for the central facility to run smoothly. The issues raised in the paper relate to user account creation, user account suspension and removal, pruning and distribution of the password files, and delegation of jobs to other individuals. Parallelizing the job of user account creation and identity verification helps in reducing the time required for user account creation. We see that fixing formats of the "gid" field and the GECOS field of the password file help in pruning the file on a category wise basis and in user account removal. Since the writing of this paper we have used several of the fields of the password GECOS field for designing schemas for an LDAP

implementation of our authentication system. This we hope will give us a seamless migration and efficiency of design of our local LDAP schemas.

References

1. Jon Finke, Automated Userid Management Simon Management System, Community Workshop 92, Resselaer Polytechnic Institute June 13-19 1992, Troy, NY
2. Rossenstien, M., Geer, D., Levine P, The Athena Service Maganement System, Usenix 1988
3. Augus: An Automatic Multi-Platform Account Generation System, Riddle P., Danckaert P., Metaferia M., Usenix 1995
4. Dr Wettistien G., Johannes G., Gaining Middle Ground: A Linux-based Open Source Middleware Initiative, Usenix 2000