User Services Strategy for User Accounts Maintenance

Beata Sarnowski, Arctic Region Supercomputing Center, University of Alaska, Fairbanks, Alaska

ABSTRACT: *Several software tools and UNICOS commands currently exist for the maintenance of user accounts. These tools/commands have proven very useful, but the separate executions for necessary procedures and the incomplete log file to trace the history of accounting are the disadvantages.*

This paper discusses a compendium of tools, scripts and techniques that ARSC consultants have informally developed or acquired to enhance account management. With the scripts and online information (internal web server), uid maintenance procedures are easier or more consistent and consultants are better able to do their jobs.

1 Introduction

The Arctic Region Supercomputing Center (ARSC) was funded on August 21, 1992 as a part of the Strategic Environmental Research and Development Program of the U.S. Department of Defense, and became operational in January, 1993. Our mission is to support environmental research and science with an emphasis on high latitudes and the Arctic. Thirty percent of the supercomputers' time is dedicated to DoD research with the remaining time available to support academic and commercial users. ARSC supports mission-related use by federal agencies, projects with other universities, and commercial organizations.

The use of supercomputers to solve highly complex problems is becoming more prevalent today. This is especially evident with the dramatic growth in the number of supercomputing facilities. This paper will explore compendium of tools, scripts, and techniques that ARSC consultants have informally developed or acquired to enhance account management.

2 Maintaining Users

User Services performs maintenance tasks such as adding, inactivating, reactivating, and deleting users, communicating with users, modifying information that controls how users interface with UNICOS, and training new consultants.

At ARSC, utilities are written in Korn shell script. Some of the latest are now being written in PERL. PERL offers the pattern matching capabilities of *awk* with many more features, faster execution, and greater capabilities.

3 Account Maintenance

To add, remove, and change user accounts on UNICOS, the *nu* utility is used. The *nu* utility is a full-screen, prompt-driven utility for interactively adding, deleting, and modifying user accounts. However, the utility has been modified to accommodate our needs. In addition, we wanted to record account maintenance activities in a log file.

Consultant tools are usually maintained by the tool's authors. Every effort is made to make certain that no value such as pathnames, field numbers or timing values are buried deep into the code. All such values are stored in variables in an initialization section at the beginning of the code for easy modification. All code contains in-line comments. Code pieces which perform specific, self-contained actions are broken into Korn shell functions. Each time any account maintenance activity is involved the UDB is backed-up.

3.1 Creating User Logins

To add a user to the UDB the */etc/nu -a* command is used. The Parallel Applications Specialist sends out by e-mail approximately 300 issues of *ARSC's CRAY T3D Users' Group Newsletter*. Each time a new user login for a T3D user is created an e-mail is sent to him in order to update a T3D mailing list. Also, the new T3D userid is recorded in a log file.

The */etc/udbgen res.script* command is used to create a new project on our system. A *res.script* is a template file (see Appendix A for a template of the file).

ARSC allocations formerly were given in units of CPU time. The *udbsee* command monitored its usage (see Appendix D for an example). Current allocations are given in Service Units (SU). The SU provides dynamic accounting for memory and disk usage as well as CPU usage on both the Y-MP and the T3D.

The *reslist2* tool (see Appendix B for an example) tracks and manages user allocations during a requested period of time. It gives a detailed usage listing that includes a user's Y-MP and T3D CPU usage, as well as memory, disk, and silo usage.

The *resadmin* command (see Appendix C for an example) initiates an interactive program to establish and modify allocation limits for users and projects (see Appendix E for a new charging algorithm - Oct. 1, 1995). It also allows granting of some management privileges to selected users of the account.

3.2 Changing User Accounts

A number of useful scripts have been developed to modify users' UDB. To be able to make changes for more than one user at the same time, the *udbgen* command is used instead of the *nu* tool.

Other kinds of interaction with user accounting maintenance include transferring files between file systems, moving a user to another project, modifying a project name, acid, and resgrp, modify the acid and group files, adding users to new project(s), adjusting user allocations, and more.

3.3 Deleting User Logins

Deleting a user from the system is more sensitive than adding a user to the system because there is the possibility that you will be removing valuable data from the system.

At ARSC we want to keep accounting records in order and we want to ensure that the userid is not reused. The *nu -d* command (in contrary to *nu -k*) is run whenever a user needs to be deleted. Deleting accounts involves changing the password so that the user cannot log in again, deleting all the user's files and directories, and deleting the user's mailbox. The script has been expanded to delete user directories on the */tmp* file system since a purge script does not do it. A user entry is still in the UDB, and even if the user cannot log in again, he/she still can receive e-mail. A special cron job runs every week and removes e-mail of deleted users, if any.

For tracking, a comment field of deleted users is modified to :

fi*rst and last name*_deleted_*date of deletion.*

4 Communicating with Users in a Variety of Ways

There is a need to frequently communicate with our users. Several methods of communication are available. The method to use in any specific instance is generally determined by the urgency of a message.

4.1 Issuing Special Messages to Users Logging in

It is important to keep users abreast of any recent developments and system changes which could affect their batch or interactive work. The ARSC staff employs tools (*chmotd* and *ednews*) to keep the user community informed of the current system status, scheduled system outages, new online documents and the like.

The *chmotd* tool adds, removes, or edits messages in the *motd*, the Message of the Day, a special message printed each time a user logs in to the system. This message is contained within the */etc/motd* file. In addition, *chmotd* maintains a log of all messages ever residing in the *motd*. This log is located in */usr/news/LOGS/motd.log*.

An *ednews* tool is a tool for use by User Services staff to create or edit news articles in */usr/news* and edit log files in */usr/news/LOGS*. When a new news article is created, *ednews* creates a template, including the title of the news article, the author's name, and the day. Options within *ednews* include renaming the article and checking the article's spelling.

4.2 Policies and information

Our future plan is to keep one copy of user policies and all information for users on the ARSC Web Server. We do not want to alienate those Cray users with limited local resources who do not have a local WWW browser installed. A non-graphical WWW browser, *lynx*, should allow that. We are in the process on porting *lynx* to denali as a alternative method for users to access ARSC user policies, news files, and other information that would be located on our Web server.

5 Web pages are never really complete; like a coral reef they just keep growing

The User Services goal is to provide access to procedures for new employees and newly employed students. To reduce new consultant training time, we have implemented *User Services pages* on the internal Web server. Access to the internal Web server is controlled by userid and password. The User Services pages are our repository of all defined procedures, developed tools, including man pages defined for those new tools. It reduces training time significantly.

6 Conclusions

The lessons learned from consultant utilities development over an extended period of time can be summed up as follows:

- Make sure that you have good online documentation and provide plenty of examples. Make sure that you have clear, concise man pages for each tool/utility which include at least one good example. Examples are the best way to demonstrate how a tool can and should be used.
- Publicize your tools. Consultants cannot use them if they do not know they exist. Give demos when and where possible.
- Recognize when a utility can improve a situation and write one.
- Tools should be easy to use. Go for simplicity in display and function.
- Stay consistent with your options and naming conventions.
- Utilities need to be useful. A simple script of only a few lines can be just as valuable as a complex piece of code if it makes someone's job easier.
- Make your code accessible to your co-workers so they can learn from your code or use all or a part of it if they wish.

7 Future Plans

Future utilities are likely to be developed in PERL and C code except where a short, simple Korn shell script will suffice. Every effort will be made to keep future utilities portable, maintainable, and functionally consistent with existing utilities.

Our plans include:

- Develop tools for account maintenance for SGI users
- Continue to develop tools for CRL (Cray REELlibrarian) maintenance

- Eliminate hardcopy mailings of new user documentation
- Develop a mechanism (different from e-mail) for announcing modifications to utilities or newly developed utilities

8 For More Information

An online version of this paper is available at ARSC's WWW server using the URL:

http://www.arsc.edu/ftp/publications/papers

You can reach the author surface via electronic mail to:

sarnowsk@arsc.edu

or through mail to:

Arctic Region Supercomputing Center
University of Alaska
PO Box 756020
Fairbanks AK 99775-6020

9 Acknowledgements

I would like to thank Mike Dority of Cray Research, Inc. who provided support and guidance.

ARSC is supported by the Strategic Environmental Research and Development Program (SERDP) under the sponsorship of the Army Corps of Engineers Waterways Experiment Station.

10 References

[1] *UNICOS Administrator Commands Reference Manual (SR-2022),* Cray Research, Incorporated, Minnesota, 1994.

[2] *UNICOS File Formats and Special Files Reference Manual (SR-2014),* Cray Research, Incorporated, Minnesota, 1994.

[3] *UNICOS User Commands Reference Manual (SR-2011)* Cray Research, Incorporated, Minnesota, 1994**.**

Appendix

[A] A template of the *res.script* file:

```
create  ::     uid    ::
comment :project resgrp:
passwd        :*:
      gids    :32:
       acids   ::
       resgrp         :: # User name::
      shflags :040000: # octal
      shares ::
# add to /etc/groups and /etc/acid
```

[B] The *reslist2* tool (an example)

SUs Usage from June 1, 1995 to September 1, 1995

[1] denali<5> reslist2 -s 010695 -e 010995 -a TEST

```
name  uid   acid   acc     pctg
doe   736   206    U       100.00

allocated                total used
4800:00:00.00            1490:44:24.48Joe Doe
```

[2] denali<6> reslist2 -s 010695 -e 010995 -a TEST -L

```
name    acid    cpu/mpp SU       disk/silo SU
doe     206     1476:22:43.39    12:38:10.69

mem SU     total used
1:43:30.38   1490:44:24.47
```

[3] denali<7> reslist2 -s 010695 -e 190995 -a TEST -l

```
name          uid      acid     cpu SU
doe           736      206      390:25:19.89

mpp SU              disk SU              silo SU
1085:57:23.49       6:40:34.51           5:57:36.18

mem SU          total used         allocated
1:43:30.38      1490:44:24.47 4800:00:00.00      Joe Doe
```

[C] The *resadmin* tool (an example)

[1] To create a new project:

```
resadmin> create account=TEST allocation=300
weight=0
```

[2] To create a new user:

```
resadmin> modify user=doe account=TEST allocation=300 access=MU
```

[D] Monitor total current CPU usage through the udbsee command

```
denali<8>   udbsee doe
create      :doe:   uid     :736:
       comment :Joe Doe:
       passwd    :*:
...
       logtime   :790377998: # Tu Sep 5 12:26:43 1995
       cpuquotaused  :2434145.7:
       shextime :790378273: # Tue Sep 5 12:31:13 1995
```

[E] New charging algorithm - October 1, 1995

```
   1.000  *  Y-MP              (CPU-hours)
+  0.010  *  T3D               (CPU-hours)
+  0.005  *  Memory            (MWord-hours)
+  0.003  *  Disk              (GByte-hours)
+  0.003  *  DMF Silo          (GByte-hours)
+ 0.000   *  CRL Tape
 ---------------------------------------------
= total SUs
```