LISA IV
October 17-19, 1990
Colo. Springs, Colorado

Life Without Root

Steve Simmons – Industrial Technology Institute

ABSTRACT

''Power corrupts.''
– I. Amin

Often the people most qualified to perform certain system administration tasks are not necessarily qualified to have root access in general. This paper will discuss the rationale and methods for having non-root accounts do some types of systems administration. We will discuss two subsystems which we are currently administering without root and apply that experience to suggest some general rules.

Rationale

''With great power comes great responsibility.''
– R. Nixon

A significant number of system break-ins are caused by incautious use of the root account. Other times an operator or new administrator inadvertently destroys sensitive system setups while using the root account for some completely unrelated purpose. Both of these have their root cause (pun intended) in UNIX's inability to grant privileges greater than ordinary user accounts without granting total access.

While there is no general solution to the problem, it can be mitigated by reducing the number of subsystems which require root access. This requires either removing the subsystem (not usually acceptable) or modifying it such that a normal user account can administer it. We have had good luck with the latter, successfully administering news and uucp without using root except for creating accounts.

Along the way we've fallen into a few holes and established a few rules for keeping things running properly. We also have some suggestions for present systems designers.

Using non-root accounts to administer such subsystems as news may have some additional benefits. In many situations management may be unwilling to have any fraction of administrative time devoted to such ''frivolous'' pursuits. Or they may allow it, but put it at such a low priority that it effectively is never done. Using a non-root account enables one to hand off responsibility to some highly motivated third party who will do the administration on his or her own time without giving that third party special privileges.

Practical Experience

''What have you done for me lately?''
– S. Goodman (deceased)

To date we have fully implemented two subsystems on which we can do rootless administration. The first of these, the network news, is described in some detail. The second, uucp, was significantly easier and is only briefly described.

News

''The News Is What You Need.''
– Detroit News Adv.

Overview

We have successfully administered both Bnews 2.11.14 [1] and Cnews [2] without using root. This turned out to be surprisingly easy; in retrospect we were fortunate in choosing news as the first subsystem to do this with.

Working as root, a standard installation of Bnews was done. (A later installation of Cnews was done, requiring work under 3 different accounts.) A new user account, news, was installed on the system. The home directory for this system was installed as **/usr/lib/news**.

It was not judged sufficient that the news account simply be able to maintain the system; we wanted it to be able to apply and install patches, debug systems, etc. Thus the first step was to find all the files needed by news and make sure they were writable by news.

We did not desire that news be able to install files in such sensitive directories as **/bin**, etc. This would have opened up the entire system to any errors or malice by the netnews administrator. Instead we decided that initial installation must be done by root, but all files installed would be owned by news.

News was installed (actually re-installed, since we were already running news) and we began. Immediately we ran into problems.

Shell scripts run by *cron*[3] tend to create files owned by root. The news user was not able to modify these files, making administration difficult. We considered a number of approaches to deal with this, and actually implemented two of them. Both worked, but the second was definitely the better method.

We considered and rejected modifying all the news scripts to explicitly set file ownerships, groups, and permissions. This would permit *cron* to run as root while not causing problems with news administering news. The modified scripts would be owned by news, letting the admin modify the scripts without having to modify **crontab**. While this probably would have worked, we rejected it as making us diverge too far from standard news setups. The fixes and scripts made available by the general community of news managers are invaluable in running news. Wholesale modification would have made tracking those fixes tedious and error-prone.

Our next attempt focused on using *cron*. We modified all the **crontab** entries so that they did a *su* to news before actually invoking the maintenance script. Note that Berkeley *cron* has a feature to do this automatically, but we were not running BSD everywhere and desired to keep **crontab**s as similar as possible. While this worked, it resulted in a rather baroque **crontab** (assuming that's not redundant) and still left the problem that any modification to **crontab** itself must be done by root.

At this point we got lucky in the person of Paul Vixie. Paul was looking for someone to beta test a new PD implementation of *cron*, *cron2*[4]. *Cron2* works rather like System V *cron*[5] but has additional small but powerful features which have eased its use. Like System V *cron*, it is significantly better about error handling, security and reporting than other *crons*. Most important, it allows users to have individual **crontab**s which are always executed as the user, not as root. Vixie *cron* worked fine even in beta release (it has been submitted to comp.sources.unix as of this writing) and quickly went into general release on our systems, displacing the vendor-supplied crons.

Two particular features of Vixie *cron* must be mentioned. First, it allows definition of a user to be contacted whenever a script has unexpected (untrapped) output. Thus the *usenet* alias can continue to receive all incoming messages (presumably both the prime system admin and the news admin would be interested in these), while ordinary *cron* messages would go strictly to the news admin.

Second, it allows an alternative PATH to be specified for all invocations from a **crontab**. This permits one to place all administrative scripts in a separate directory not in the standard user path but does not require 'hard paths' in the shell scripts. This reduces system clutter and makes the scripts more readable and maintainable. It also increases system security in several ways. First, an explicit path defeats several known security holes; second, one can make the script directory inaccessible to ordinary users.

This done, we proceeded to use the system. To our pleasant surprise it worked almost immediately, with most of the tuning done on the shell scripts.

When Bnews patches 15-17 came out, we were able to install them over the existing software without requiring root. Similar results were obtained with *rn*[6] and *nn*[7]. When we converted to NNTP-based news, root was required only to create new programs in areas such as **/bin**.

When the time came to convert to Cnews, we were able to administer it as news immediately. No script or program modifications were required. The installation method for Cnews is rather baroque, and will require root intervention for the foreseeable future.

Our news administration has been done on BSD 4.3 and System V.2 without root for approximately a year; on Ultrix for 5 months. It is now being ported to SunOS 4.1.

UUCP

''Uucp is a dead protocol.''
– P. Honeyman

With news in place, the next task was uucp. This too proved easier than expected. Building on our lessons from Vixie *cron*, it took only odd moments across a week to convert to non-root administration.

One might ask why bother with uucp? We have between 45 and 60 uucp connections at any given time, and are regularly adding or deleting them. Often queues or locks need to be reset, and we run an extensive set of trouble-watching scripts.

Uucp administration also maintains and updates our *pathalias* [8] database and maps. This required some careful division of responsibility between news and uucp, but has been running as a stable system for some time.

Creation and deletion of uucp login accounts still requires root intervention, but all other work is now done by the uucp account. This system has been in production use on BSD 4.3 and System V.2 for approximately a year. As of this writing we are converting it to HDB uucp on SunOS 4.1 and Ultrix 4.0.

Other Possibilities

''Anything's possible.''
– U. Geller

We are convinced by our initial success than many subsystems could be broken away from root. Other more pressing tasks have prevented us from addressing them, but we hope to return to them by the end of this year. Some of these are listed below, with preliminary thoughts on difficulty.

Line Printers

Like news and uucp, it should be possible to administer the line printer spooling system without requiring root. Informal contacts suggest that *plp*, the public domain replacement for BSD-style spooling, could easily be used for this purpose.

There are additional difficulties which come from the overuse of the daemon account, see below for more on this.

Mail

We are convinced it is possible to administer mail without requiring root. What we are not yet convinced of is whether it is worth the effort. Such things as alias files, sendmail or smail configurations, etc, could be managed from an unprivileged account with relatively little effort. Going beyond that may require massive effort, and the early returns say it would probably not be worth it for a single site.

Mail also suffers from the overuse of daemon.

Prescriptions For Rootless Subsystems

''Just follow these 327 simple steps...''
– Too common

To be a candidate for non-root administration, a subsystem must be relatively self-contained and require few or no suid root programs. Once those guidelines are met, a relatively small number of rules will help get the job done.

Distribute Responsibilities Carefully

Dividing work between the news and uucp accounts proved interesting. Consider the case of comp.mail.maps and *pathalias*.

Once uucp maps have been unpacked for *pathalias* and queued to remote systems there's no point in keeping them in the news system. But who should be responsible for what? Currently the news system has responsibility for unpacking the maps and creating the pathalias database. This is because any other solution would have required giving uucp permission to delete or modify files owned by news. But the pathalias database in properly the province of uucp, not news. Since we only maintain one map set for our entire network, we build several databases on one central system and distribute them as needed.

Our initial selection is functional, but probably not optimal. Doing this as news requires a news account on all systems, regardless of their having news. A better breakdown would be to have news unpack and install the maps, then perform expires on comp.mail.maps after enough time has passed to feed our neighbors. Uucp should be creating and updating the databases.

One Account, One Purpose

As one possible solution to above, we considered having a single account to manage both systems. We rejected this on two grounds. First, it violated the rule of keeping separate systems separate. Second, it could lead to errors when suddenly the entire news system seems to be owned by uucp or vice-versa.

There's No Place Like Home

Location of homes for uucp and news accounts turned out to be important. Putting them in **/usr/lib/news** or **/usr/lib/uucp** quickly turned out to be a bad idea. Any account has a lot of files associated with it, and having those files appear in the system management areas quickly leads to confusion. The best solution was a subdirectory under each of these that was the home for the account.

We were concerned that having uucp someplace other than **/usr/spool/uucp** or **/usr/spool/uucppublic** might lead to problems with files being delivered to unexpected places; in practice this has not been a problem.

Get A Real Cron

You can run without root using even the oldest, creakiest *cron*. It's just a lot easier with a good one. Standard system V *cron* is the minimum acceptable for our systems, Vixie *cron* is the preferred solution.

Never Use Root!

Once you've converted to non-root usage, never get lazy and do something as root. More than once this has broken things in strange and subtle ways. The most common problem comes from running some news or uucp script as root. The script leaves behind files owned by root rather than the administrative id. These files are usually no longer usable by the subsystem, causing lots of interesting problems. In general, once you've converted a subsystem to non-root administration, don't go back.

Daemon Considered Evil

One of the difficulties we see in converting some systems to rootless administration is the gross overuse of the daemon account. Some mail logs use it; the line printer spooler uses it; everybody and their worm's mother uses it. This might have made sense once, but no longer does.

Giving a daemon user account would award the owner power over a large number of different subsystems. This is not acceptable for reasons of security, safety, and compartmentalization. Any subsystems we administer from non-root accounts will not use daemon.

Future Directions

''This is another fine mess you've gotten me into.''
– R. Reagan

It is not difficult to design subsystems such that rootless administration is possible. Such systems as backups, batch processing, network management, etc, could very easily be administered without root. Given such tools as suid root *perl* scripts, even tasks like user account management could be done without requiring root.

If this is true, why doesn't it happen more often? The primary reasons seem to be:

- lack of forethought on the part of the system designers;
- lack of documentation on how to do so;
- lack of time or willingness on the administrators part.

All of these are solvable problems; we hope this paper has at least partially lit an unnecessarily dark corner.

References

''Read the f-f-f-f-fine manual.''
– All of us

[1] M. Glickman, M. Horton, R. Adams, *''USENET Version B Installation''* from *''UNIX System Managers Manual, 4.3 Berkeley Software Distribution'',* Computer Systems Research Group, Computer Science Division, Department of Electrical Engineering and Computer Science, University of California, Berkeley, April, 1986. SMM:10.

[2] G. Collyer, *''Installing ''C News'' Network News Software,''* CNews documentation. Posted to comp.sources.unix volume 19, 1989.

[3] Cron(8) manual page, from *''UNIX System Managers Manual, 4.3 Berkeley Software Distribution'',* Computer Systems Research Group, Computer Science Division, Department of Electrical Engineering and Computer Science, University of California, Berkeley, April, 1986.

[4] P. Vixie, unpublished cron2 documentation. With cron2 beta release software, December 1988.

[5] Cron(8) manual page, *''SunOS Reference Manual, Volume III'',* Sun Microsystems, Revision A, March 27, 1990.

[6] L. Wall, rn installation documentation and scripts. Posted to comp.sources.unix, volume 1, 1985.

[7] K. F. Storm, NN usage and installation documentation. Posted to comp.sources.unix volume 22, 1990.

[8] P. Honeyman and S.M. Bellovin, *''PATHALIAS or The Care and Feeding of Relative Addresses,''* ''USENIX Conference Proceedings,'' Atlanta, Summer 1986.

Steve Simmons is a graduate of the University of Michigan, and has done UNIX-based development at Bell Northern Research, Schlumberger Technologies, and ADP Network Services. He is currently the UNIX systems manager at the Industrial Technology Institute and a consultant. His publications include music, humor, essays, and software. He has published no intentional fiction. Reach him at Industrial Technology Institute; P. O. Box 1485; Ann Arbor, MI. 48106 or electronically at scs@iti.org .