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

Tinsley Galyean – Brown University
Trent Hein and Evi Nemeth – University of Colorado

Trouble-MH: A Work-Queue Management Package for a >3 Ring Circus

ABSTRACT

Efficient management of users' problem reports and requests is a fundamental system administration task. This paper presents a work-queue management solution (Trouble-MH) based on the Rand MH Message Handling System. Trouble-MH allows a system administration team to easily track and resolve users' "trouble" reports.

The Problem

Efficient communication between a site's users and its system administration group is the key to a happy campus. At a small site, users know their system administrator (SA) on a first name (or at least login name) basis, and thus reporting a system problem is as simple as chatting next to the coffee machine or sending a mail message to the SA.

At larger sites, however, life gets much tougher. In our case, we needed a way for the system administration staff to manage the queue of incoming system "trouble" reports as well as requests for general information, system improvements, etc. We call the queue which collects all of these messages the "trouble queue."

When we set out to design a package to manage this queue, we had these things in mind:

- It is impossible for the user community to keep track of the names of members of the system administration group. Our Engineering Research Computing Center is staffed by 10-20 SAs, many of whom are students, and thus come and go with the school year.
- In order to simplify status tracking of a task as well as provide users with a personal reply, mail acknowledging the problem should be from a specific individual rather than a program or generic user like "sysadmin" or "trouble."
- Because our system administration group is large and spread out across 750,000 sq ft, it must be easy for any system administrator to electronically determine the current status of a task, including who is working on it already and what progress they have made.
- Multiple SAs need to be able to view and resolve items in the queue simultaneously, without concerns like the need for a lock on the entire queue which using /bin/mail might introduce.
- The solution for trouble queue management needs to be easy-to-learn, easy-to-use, and easily integrated with an SA's personal mail handling methods.
- It is always easier to provide extensions to an existing mail management system rather than to write a new one from scratch. While the finished product is not always perfect in this case, if the underlying system is powerful enough, it can grow as needs change.
- SAs need to be able to resolve items in the trouble queue from their own workstation or terminal at home without being forced to login to some overloaded central queue-management host.
- Each SA must login as himself and be accountable for his actions; therefore no generic "sysadmin" login should be used.

The Base Solution

We decided upon a solution based on the Rand MH Message Handling System. This system, hereafter referred to as MH, is publicly available via anonymous ftp from a number of sites. This paper assumes the reader is familiar with standard MH commands. Readers who are unfamiliar with MH are referred to the MH documentation [3].

MH is well-suited for managing a multi-access "trouble queue" because:

- The trouble queue can be managed as an MH "folder." SAs who normally use MH for their personal mail box can easily switch between their personal mail folder and trouble mail (which appears as a normal mail folder).
- The MH trouble folder can be kept on one host but exported via NFS to other hosts. (Please note, however, that MH performs poorly on memory-starved diskless workstations.)
- Since MH operates on a one-message-per-file basis, there is no need to lock the entire queue

for common operations.

- Trouble mail readers can be added without requiring root or other special privilege access. (access to the trouble folder, if necessary, can be restricted by creating a "trouble" group.)
- SA's can resolve items in the queue from their own account, adding that "personal touch" to the reply the user receives.
- The MH system is well-documented.

The Trouble-MH Gameplan

First of all, let's consider how we'd really like to see things happen:

1. Desperate user mails "trouble" describing a problem using their favorite mail sending technique and editor (no need to learn how to use some special trouble reporting command).
2. User's message appears in SA trouble queue.
3. On-duty SA examines message. If the issue can't be resolved immediately, an acknowledgement is sent back to the user saying that the system administration group has received the message and is looking into it. This acknowledgement is recorded in the message so that other on-duty SAs don't also acknowledge it, thereby drowning the user in content-free mail.
4. As the status of the problem changes, the user is notified and that status report is also recorded in the message. This allows any of the SAs or their managers to see the status of the item and its resolution to date.
5. When the issue finally gets resolved, the user is notified and the item is removed from the queue.

Implementation

After much thought, we decided that it was not sufficient for our purposes to use the MH system strictly "as is." We also did not have the resources to do the project right; so a combination of shell/C hacks were done as a prototype. As with many quick hacks, a useful tool emerged. Trouble-MH now in use by a handful of system administration groups at the University of Colorado to manage a variety of work queues.

For convenience, a user "trouble" was created on one of the machines belonging to our system administration group, and an alias was set up in the aliases file (either /etc/aliases or /usr/lib/aliases) which directed all mail sent to "trouble" to this account. This account has a .forward file which directs incoming mail through the *slocal* program and into the trouble folder.

In each SA's MH directory (usually ˜/Mail) there is a symbolic link which points to this common trouble folder either on the local machine or an NFS mounted partition. To list the messages in the trouble folder, an SA can type the standard MH command:

```
scan +trouble
```

We want the status/acknowledgement information to be stored in the body of the message in the queue. In order to provide this functionality, a new command, *stat* was introduced. *stat* invokes *repl* with these lines appended to the end of the message:

```
----------
Status: by <your-login> <date>
----------
```

After you are finished editing, *stat* appends the text you typed below the Status: line to the end of the message, leaves it in the queue (for others to see later), and also sends it to the user reporting the problem. A typical message in the trouble folder looks like:

```
(Message trouble:10)
To:      trouble@boulder
From:    ajsh@wild.colorado.edu
Subject: Bug in Sun Fortran 1.3
Date:    Wed, 18 Jul 90 18:06:01 MDT

Is this a bug in Sun Fortran 1.3
compiler?

Here's my floating point option, my
fortran program tst.f, how I compile
it, and the results.

  <code example>

What do you make of it? -- Andrew

- ----------
Status: by huntert Thu Jul 19 10:50:13 1990
- ----------
Trouble has received your message, we'll
look into it asap.

- ----------
Status: by hardt Thu Jul 19 16:40:27 1990
- ----------
I have looked at this with him and we
have gotten it down to a simpler example.

- ----------
Status: by kiefer Mon Jul 23 11:44:56 1990
- ----------
I have verified that this is in fact a
bug in the compiler and have reported it
under our software support contract.  I'm
awaiting a reply from Sun ...
```

Another command, *res*, which stands for "resolve message", was also created. *res* uses *repl* to reply to the message and then uses *rmm* to remove it from the queue. One could also refile the message in an old-trouble folder for later reference instead of just removing it.

Thus, *stat* and *res* are front-ends to repl and provide a uniform reporting and tracking mechanism.

After about a year of use, we noticed that SAs spent a lot of time sending the same reply to multiple messages about the same topic (a major print server was down, so ALL the secretaries sent mail about it, for example.) In order to deal with situation, a third command, *glom*, was added. *glom* takes a list of messages and combines them into a single message, with copies going to all the appropriate people, so that a single reply suffices.

Other Advantages

One of the side effects of using Trouble-MH is that each message can serve as a teaching tool for new SAs. It not only shows how to solve a particular problem, but also is an example of the tone of response you expect the SA staff to use in answering users complaints. As such it allows managers to identify incorrect answers or unprofessional responses and bring them to the attention of the errant SA.

Queue-MH

The package was originally called Trouble-MH and was intended for SA trouble mail use only. After a bit of use, it became clear that Trouble-MH was useful in other arenas. In our own group, we soon had Operator-MH (for the operator backup/restore queue) and Wiring-MH (for the wirers). We merged them into one generalized package called (for lack of a better name) Queue-MH.

Acknowledgements

Trouble-MH was originally designed and implemented by Tinsley Galyean and Trent Hein (under the supervision of Bob Coggeshall) in October, 1988. Functionality improvements ("glom", in particular) have been added by John Hardt. Barb Dyker added queue generalization and management documentation. Most recently, Andy Kuo cleaned up the scripts and improved locking.

Source Availability

You can obtain the source code and documentation for the package described in this paper via anonymous ftp from boulder.Colorado.EDU [128.138.240.1] (pub/queuemh.tar.Z). Although we don't officially support this distribution, and are actually somewhat embarrassed about it, we use it daily. Please report any bug fixes or suggestions you have to "systems@boulder.Colorado.EDU."

References

[1] Dyker and Hein, "Using Queue MH," May, 1990.

[2] Dyker, "Managing Queue MH," May 1990.

[3] The Rand MH User's Manual.

Tinsley Galyean received a BS in Computer Science from the University of Colorado, and recently received his Masters degree from Brown University. This fall, Tinsley is continuing his education at MIT. Contact him electronically at tag@media-lab.media.mit.edu .

Trent Hein is both on the staff and a student at the University of Colorado, Boulder. Trent spent his past summer at UC Berkeley working on the 4.4 BSD [illegible]. Contact Trent at University of Colorado, Boulder; Department of Computer Science; P. O. Box 430, Boulder, CO 80309-0430 or electronically at trent@boulder.Colorado.EDU .

Evi Nemeth spent the past year at Dartmouth College on leave from the University of Colorado, Boulder, where she is on the faculty of the Computer Science department. She has recently co-authored a book on system administration: *The UNIX System Administration Handbook*, published by Prentice Hall. Contact Evi at University of Colorado, Boulder; Department of Computer Science; P. O. Box 430; Boulder, CO 80309-0430 or at evi@boulder.Colorado.EDU electronically.