Open Computing ‘‘Hands-On’’: ‘‘Wizard’s Grabbag’’ Column: April 94

Summarizing UUCP Logs

A Perl program that summarizes information from the HoneyDanBer UUCP logs and a Korn shell random-number generator

The logs that track the activity of the UUCP subsystem can often be a confusing mishmash of timestamps and host, user, and file names. To help make better sense of the data, Mike Brininstool provides a Perl program that summarizes information from the otherwise hard-to-read logs written by the `uucico, uuxqt, uucp,` and `uux` HoneyDanBer UUCP programs. His program allows you to customize the report to extract the data you need more easily, allowing you to track UUCP connection and job transaction status.

Peter Turnbull submits a Korn shell script that provides a handy way to generate a random number whose value is distributed uniformly within a specified range. This utility would be useful in other shell scripts as well as interactively.

History Repeats Itself

Dear Editor:

I run a small site that uses UUCP to transfer electronic mail and Usenet articles and decided to write some utilities for monitoring communications. Originally I wrote them in the Korn shell language, but they were so slow that I rewrote them in Perl, improving their execution speed significantly.

I’m submitting `uuhistory` [Listing 1], which selects certain entries from UUCP logs, sorts them chronologically, and writes a report on the standard output. Current System V Release 4 HoneyDanBer UUCP logs are in the `/var/uucp/.Log` directory tree. There are four subdirectories named after the program that creates the log entries: the `uucico` and `uuxqt` daemons, and the `uucp` and `uux` user programs. These programs then write entries to a file named after the remote host. The `uudemon` cleanup script will consolidate these logs, writing them to a file named `Old-Log-1` in the `/var/uucp/.Old` directory. Typically, several backup copies of the old combined logs are maintained with names `Old-Log-`*`N`*, where *`N`* is a small integer.

Invoke `uuhistory` without arguments to create a report for the current log entries (`/var/uucp/.Log` directory tree). Specify one or more host names to limit the report to those hosts. Alternately, you can name a file as the argument to the `-f` option to examine it instead. For instance, name `/var/uucp/.Log/uucico/uunet` to summarize the current logs written by `uucico` for communicating with the site uunet, or `/var/uucp/.Old/Old-Log-1` to create a report from the combined log entries created the last time the ‘‘cleanup’’ script was run.

Michael P. Brininstool / Plano, Texas

Configuration Note. You can use `uuhistory` with pre-SVR4 HoneyDanBer UUCP systems by changing the top-level directory from `/var/uucp/.Log` to `/usr/spool/uucp/.Log`.

Implementation Strategy. The `uucico` Log entries follow the progress of a transaction. Thus, `uuhistory` was designed to detect the beginning and end of each transaction. If an entry contains the ''SUCCEEDED'' keyword, it indicates that the local host is initiating a call to the remote host. This step should be followed by a ''start-up'' entry, indicating that both hosts can now communicate. If a start-up entry is seen without the preceding SUCCEEDED line, then the remote host called the local one.

The ''complete'' entry indicates the end of the transaction. The ''OK'' keyword indicates success, whereas the ''FAILED'' entry says that the closing handshake wasn't successful. Presumably, the transaction was terminated prematurely. Other keywords indicate receipt of a signal, such as ''CAUGHT'' or ''INTREXIT''. Note that the ''->'' and ''<-'' symbols are used in the report to indicate an outbound or inbound connection relative to the local host, respectively, not the file-transfer direction.

Implementation Notes. Line 7 defines the top-level directory, which may need to be changed for your implementation.

Lines 16-31 process the command-line arguments. The `$fflag` variable is set if the user specifies the `-f` option. Any remaining arguments--which should be host names--are stored in the `@hosts` array. If no arguments were specified, `uuname` is run without options, which names all hosts listed in the HoneyDanBer `Systems` configuration file. Note that this ''systems'' file can be renamed by a `uucico`-services entry in the HoneyDanBer `Sysfiles` configuration file.

Line 34 opens a pipe to write data to the system `sort` command. This way the resulting report will be ordered chronologically because each line begins with a time stamp.

The program is driven by several loops. The outermost loop (lines 37-115) performs one iteration for each host. Within this loop are three primary sections to the program: the first analyzes the transaction logs written by the `uucico` daemon, the second deals with entries written by `uucp` or `uux`, and the last manages `uuxqt` records. Each section opens the requisite log file, selects lines with the appropriate keywords, extracts information from the proper fields, and may reformat it for presentation. In particular, the date and time is reformatted so the file can be sorted chronologically.

If the user specifies a file to analyze with the `-f` option, that file is only opened once, but subsequent program sections seek to the beginning of the file before selecting the necessary entries.

Space limitations prevent more extensive analysis and summary of the logs in this month's column. However, you can apply the principles illustrated by this program to customize the report to your particular needs.

Random Thoughts

Dear Editor:

I often find it helpful to generate random numbers without resorting to C language code. The `krand` Korn shell script [Listing 2] generates a random number in the specified range. By default, the `RANDOM` environment variable is seeded by the process -ID number of the shell running the script. A subsequent

modular calculation produces a random number in the range specified.

Peter Turnbull / Australian Tech Resources / Canberra, Australia

Explanation of `krand` *script*. Line 8 provides an initial value for the `Seed` variable, which is the process-ID (PID) number of the shell running this script. The script user can override this value by specifying a seed value argument on the `krand` invocation command line.

Lines 10-14 process command-line arguments. If two arguments were specified, line 11 stores their values in the `Lower` and `Upper` variables; line 12 also stores a third argument in the `Seed` value. Any other argument count is an error, so line 13 displays the correct usage and terminates the program.

Next, several ‘‘sanity’’ checks are made: Lines 16-24 make sure that the lower, upper, and seed values are integers. The `expr` command--on lines 17, 20, 23--returns an exit status of two if the expression is invalid, which is the case if the value added to zero is not a decimal integer. Lines 27-34 check that the lower, upper, and seed values are within the correct range, which is equal to or greater than zero, but less than 32,768. Finally, lines 36-37 make sure the upper value is greater than the lower one.

If all these ‘‘sanity’’ checks pass, line 40 initializes the Korn shell’s random-number facility by assigning an integer to the built-in `RANDOM` environment variable. This variable will contain an integer value that is uniformly distributed from 0 to 32,767 whenever it is referenced. Line 43 divides that value by the specified range using the modulo operator so the result will be less or equal to the range. The lower value is then added so the result will fall within the specified range. Finally, line 46 places the derived value on the standard output.

Acknowledgments

I wish to thank the following readers for their help with testing this month’s contributions (operating system): Kees Hendrikse, Echelon Consultancy, Enschede, The Netherlands (current SCO Unix and Xenix versions) and Endre Balint Nagy, Walton Networking Ltd., Hungary (UnixWare Application Server, 1.0).

Copyright © 1995 The McGraw-Hill Companies, Inc. All Rights Reserved.
Edited by Becca Thomas / Online Editor / UnixWorld Online / beccat@wcmh.com

Go To Content Search

Last Modified: Tuesday, 22-Aug-95 15:47:22 PDT