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

Untying the Sendmail Gordian Knot

Here’s a Perl script that reveals how your Sendmail configuration rewrites addresses

By Becca Thomas

*Request:*We’re looking for someone to rewrite `chkaddrs` for Version 8 Sendmail. We’ll pay $100 to the reader who contributes the version we decide to publish. Please start with the version of `chkaddrs` discussed here and try to maintain its style.

The Sendmail program has a dubious reputation: it’s difficult to configure and even more difficult to understand. Yet it’s still the most universal mail delivery agent used on the Internet. In response to configuration angst, Rob Kolstad and colleagues provide a Perl program that processes pre-Version 8 Sendmail rule-test mode output to help the user configure address rewriting, probably the most difficult part to get right. I’ve taken a look at their creation and made some modifications of my own to simplify their Perl script.

Rule Sets Rule

In my ongoing effort to understand internetworking, I attended the ‘‘Internet Security With Firewalls’’ tutorial presented by Rob Kolstad and Tina Darmohray at the Winter 1994 Usenix conference. One utility discussed caught my attention, `checksendmail`, which can help the administrator to set up Sendmail’s configuration file. I rewrote `checksendmail` for presentation in this column. I renamed it `chkaddrs` to avoid confusion with the original. Functionally similar to `checksendmail`, `chkaddrs` has been commented extensively and reformatted for clarity and now uses Perl’s ‘‘format’’ facility to beautify the report. Part A of the Listing shows the resulting program.

Usage Notes. To use `chkaddrs`, provide a list of addresses to test in a file. If the file uses the default name `address.resolve`, just type ‘‘chkaddrs’’ to get the report. You can also specify an alternate Sendmail configuration file instead of using the default, `/etc/sendmail.cf`. If you’re running `chkaddrs` without permission to write in the queue directory, you can specify an alternate for which you do have access. Part B shows the general command-line format.

Implementation Notes. Lines 8-16 comprise the configuration section. Line 8 defines the `$=` variable, which determines the page length for the output report. We set `$|` to a nonzero value to force a flush after every write (or print) operation. Line 10 defines the default address file name, line 11 the default Sendmail configuration file, line 12 the temporary file name, and lines 13-16 define the correct command-line usage.

Operation Notes. Lines 18-40 collect command-line options and check their arguments. Status information is displayed for the user by lines 43-47. Lines 50-57 define a handler to trap keyboard-generated interrupt and quit signals so the temporary file can be erased before the script terminates.

Lines 61-87 define the formats of the output reports. Line 61 overrides the default page length with the value assigned to the `LINES` environment variable, if defined. Line 65 sets `$-` to indicate to the write command that the print commands on lines 45-47 had already consumed five lines of the first page. See Chapter 11 in *Learning Perl* by Randal L. Schwartz, (O'Reilly and Associates, ISBN 1-56592-042-2), for an extensive discussion of the format declarations that are used by the Perl write command.

The main program consists of two primary sections: the first part (lines 151-198) reports the delivery agent, destination host, and destination user assigned for each input address interpreted as the recipient address; the remaining section reports how the Sendmail configuration will rewrite each input address interpreted first as a recipient then as the sender address. Both sections prepend the address with appropriate rule-set directives and feed the result to the Sendmail program invoked in rule-test mode (specified by the `-bt` command-line option). The Sendmail output is postprocessed to generate the desired reports.

Part C of the Listing shows some sample input addresses, and Part D shows the resulting report produced by `chkaddrs` on a SPARCstation 1 clone running Sendmail Version 4.1 under SunOS 4.1.1. In this example, Sendmail is run on a ‘‘mail hub’’ machine, which collects mail from other workstations on the local network and sends it to remote sites over the Internet. Local workstations forward all mail to the hub, where local users read their mail over the local network, say by remotely mounting the ‘‘spool’’ directory from the hub machine.

In Part C, `beccat` is a local user with an account on a workstation named `yang`. Thus, the first address is on the local hub machine, the second is for the same user on their workstation, and last is for Rob Kolstad, who is reached over the Internet.

Part D displays the status messages, then a section that reports the delivery agents and associated destination hosts and users. The Sendmail program didn't name a destination host for the local delivery agents so the value used to initialize the `$dest_host` variable (`XXX`) is displayed.

The Sendmail program selects the `ddn` (for Defense Data Network, now called the Internet) delivery agent for the remote address, `kolstad@bsdi.com`. The destination host is `bsdi.com` as expected, but the destination user, `kolstad<@bsdi.com>` looks somewhat strange. Actually, the Sendmail configuration file on the test system uses the *`string<@host.domain>string`* internal format for names within its rewriting rules. All addresses are placed in this format except for local names, which have no host part. The angle brackets allow the relevant host name to be on the front (for source routing) or on the back (normal form) to make it easy to find.

Next, `chkaddrs` displays how the input would be rewritten if it were a recipient or a sender address. The domain names are added to local addresses for use by the `ddn` mailer but passed through unchanged for the ‘‘local’’ delivery agent.

You'll need to enhance `chkaddrs` to support IDA and V8 Sendmail because they allow a different sequence of rule sets for header and envelope addresses. Refer to *Sendmail*, by Bryan Costales with Eric Allman and Neil Rickert (O'Reilly and Associates, ISBN 1-56592-056-2), for more details.

About the `checksendmail` Program Authors

The `checksendmail` Perl program was written by Gene Kim (also author of the Purdue Tripwire security system) under the supervision of Rob Kolstad. Both Jeff Polk and Rob Kolstad improved on the program later; in particular, Rob added V8 Sendmail support. Rob Kolstad is president of Berkeley Software Design Inc., purveyors of the BSD/386 Berkeley Unix-workalike for PCs. Gene Kim graduated from Purdue University and is now attending University of Arizona graduate school. Jeff Polk is a member of Berkeley Software Design's technical staff.

Edited by Becca Thomas / Online Editor / UnixWorld Online / beccat@wcmh.com

Go To Content   Search

Last Modified: Tuesday, 22-Aug-95 15:45:56 PDT