NAME

ph − CCSO Nameserver client (on-line phone book)

SYNOPSIS

ph [**−s** *server*] [**−p** *port*] [**−t** *type*] [**−f** *field1,field2,...*] [**−mMrRbBTlLF**] query
ph [**−s** *server*] [**−p** *port*] [**−t** *type*] [**−f** *field1,field2,...*] [**−h** *topic*] [**−mMnNrRbBTlLFcC**

DESCRIPTION

Ph queries the CCSO Nameserver, a database of University faculty, staff, and students. The database contains nearly all the information in the *Student/Staff Directory* (the University phone book), as well as other information, including electronic mail addresses.

Ph may be used in two ways; interactively or with command-line arguments.

If given arguments, *ph* will treat the arguments as a query and return the results of the query. For example,

```
ph paul pomes
```

would return the entry for the maintainer of *ph* , Paul Pomes. For more information on what types of queries you may make, see the **QUERIES** section below.

If given no arguments, *ph* will enter interactive mode, print a prompt, and wait for commands. Interactive mode will be discussed in detail below.

Ph is not intended for the generation of mailing lists. Therefore, it will refuse any requests resulting in more than a small number of matches. This is not negotiable.

OPTIONS

Ph recognizes the following options. They may be specified in the *PH* environment variable, given on the command line, or set with the **switch** command from inside *ph*. Options given with the **switch** command override all others; options given on the command line override those in the *PH* environment variable.

−n Do not read the *.netrc* file. This option has meaning only when using *ph* in interactive mode (see below for descriptions of the *.netrc* file and interactive mode).

−N Do read the *.netrc* file. This is the default.

−r Do not reformat *alias* and *email* fields to show alias-based e-mail addresses.

−R Reformat *alias* and *email* fields to show alias-based e-mail addresses. This is the default.

−b Do not beautify query responses; print them in gory detail, complete with response codes.

−B Beautify query responses. This is the default.

−m Do not use a paging program (like *more*(1)) when printing responses.

−M Use a paging program. This is the default.

−l Do not label field values with field names when printing queries.

−L Label field values with field names when printing queries. This is the default.

−t *type*
Use *type* as a default type on queries. This is just like adding **type=***type* to all queries. The **−t** option can be overriden by specifying an explicit type in the query, as in, "`ph pomes type=phone`".

−T Do not specify any type by default; this is the default.

−s *server*
Use *server* as a Nameserver host, instead of the default host. A list of suitable servers may be found with the query "`ph alias=ns-servers`".

−p *port*
Connect to the tcp port *port*, instead of the default port.

−c Do not wait for confirmation of edit commands. This is the default.

−C Wait for confirmation of edit commands (see **edit** below).

−f *field1,field2,...*
: Return fields *field1,field2,...* instead of the default list of fields, if no return clause is specified on queries. This is just like adding "`return field1 field2 ...`" to all queries.

−F Return default list of fields; this is the default.

−h *topic*
: Display a list of on-line help topics. If the **−h** option is followed by the name of one of the on-line help topics, the help screen for *topic* will be displayed.

QUERIES

The Nameserver's database contains over 70,000 entries. Each entry is comprised of multiple *fields*, each containing information about the entry. Each field has a name that is descriptive of what the field contains; for example, the field named *address* contains the office mail address of the person in question (for more information on fields, see the description of the **fields** command in the **INTERACTIVE** section below).

By default, queries are assumed to refer to the *name* or *nickname* field of the entry. Therefore, saying "`ph john doe`" looks for entries whose *name* or *nickname* field contains "john" and "doe". Fields other than the *name* and *nickname* fields must be specified; for example,

```
ph pomes address=DCL
```

would return entries with *name* or *nickname* "pomes" whose *address* contained "DCL."

Matching in *ph* is done on a word-by-word basis. That is, both the query and the entry are broken up into words, and the individual words are compared. Although *ph* is insensitive to case, it otherwise requires words to match exactly, with no characters left over; "john" does **not** match "johnson", for example. This behavior may be overriden by the use of normal shell metacharacters ("?" to match any single character, "*" to match zero or more characters, and "[]" to match a single character from a set of characters).

Ph will display only entries that match **all** of the specifications in the query. For example,

```
ph paul pomes
```

will return all entries with **both** "paul" and "pomes" in the *name* or *nickname* fields.

Ph returns a certain set of fields by default. It is possible to ask for different fields in a query. This is done by specifying the *return* keyword and listing the fields of interest. For example,

```
ph paul pomes return email
```

would print only the electronic mail address of the maintainer of *ph*. You may also ask for all fields in the entry by using "all" as a field name. This will show you every field you are allowed to see in the entry.

All output from *ph* is sent through *more*(1) (or whatever program is specified in the *PAGER* environment variable).

INTERACTIVE MODE

If *ph* is given no arguments, it enters interactive mode, where it prompts for, executes, and displays the results of Nameserver commands. Interactive mode provides access to more Nameserver features than mere queries. Some of these features require the user to identify him/her self to *ph* by use of the **login** command; others do not. Commands may be abbreviated, provided enough characters are given to distinguish them from other commands.

.netrc file
: *Ph* reads the same *.netrc* file as does ftp (see *ftp*(1)). If it finds a *machine* named "ph" that has a login and a password specified for it, *ph* will automatically do a *login* command, using the values from the *.netrc* file. *Ph* will silently refuse to use a *.netrc* file that has any permissions for group or other (see *chmod*(1)).

Public Commands

The following commands do not require the user to be logged in to the Nameserver:

help *[topic]*
Provides explanations of Nameserver commands. Given no arguments, **help** lists the available help topics. Given a *topic* as an argument, **help** will print help for that topic. A list of commands and a one-line description of each command may be obtained by requesting the topic *commands*.

query Performs Nameserver queries exactly like non-interactive *ph* queries except that metacharacters do not have to be quoted.

fields Lists the fields currently in use in the Nameserver. For each field, a display like the following (admittedly ugly) is produced:

```
-200:2:email:max 64 Lookup Public Default Change
-200:2:email:Preferred electronic mail address.
...
```

The leading number is a reply code from the Nameserver. The next number is the field number. Following the field number is the name of the field, the maximum length of the field, and the attributes for the field. The second line has, in addition to repeated reply code, number, and name, a one-line description of the field.

The attributes determine how a field may be used. *Lookup* means the field may be searched in a query. *Indexed* means the field is indexed (at least one *Indexed* field must be included in every query). *Default* means the field is displayed by default. *Change* means that users may change the field.

set *option[=value]*
Allows Nameserver options to be set. These options are for future use.

switch −*option [value]*
Allows *ph* options to be set. See the **OPTIONS** section above.

quit Exits *ph*.

login *alias* Identifies the user to the Nameserver. *Alias* is your Nameserver alias, a unique name for you in the Nameserver; it is printed in *ph* queries, as the first thing after "email to:":

```
email to: p-pomes@uiuc.edu (paul@uxc.cso.uiuc.edu)
```

In this case, the alias is "p−pomes". You will be prompted for your Nameserver password when you give the **login** command, unless you are using *ph* from the login in your email field (the one in parentheses on the "email to:" line), and your system administrator has made *ph* "setuid root", in which case no password will be required.

Your Nameserver password is **not** the same as your system password; the only way to discover your Nameserver password is to bring yourself and a University ID to the CCSO Accounting Office in 1420 DCL. Because of abuses in the past, passwords cannot be given out via email, phone, or to third parties.

You are allowed to change your Nameserver alias; there are, however, restrictions on Nameserver aliases; they must be unique within the Nameserver, they cannot be common names ("david" is right out), and they can only contain letters, digits, dashes (−) and periods (.).

Commands Requiring Login

The following commands require that the user executing them be logged in to the Nameserver.

passwd *[alias]*
Changes your Nameserver password. You will be asked to type your new password twice. *Ph* will complain if your password is too short or contains only numbers (although it does allow such passwords). Privileged users may change the passwords of certain other users by specifying the alias of the other user when giving the **passwd** command.

me Lists the Nameserver entry of the currently logged-in user.

edit *field [alias]*

Allows *ph* users to change those fields in their entry that have the *Change* attribute set. *Edit* will retrieve the value of the named field (if a value exists), and will allow the user to edit the value with *vi*(1) (the *EDITOR* environment variable may be used to override the use of *vi*). The changed value will then be reinserted in the Nameserver. If the **−C** option is in effect, the message, "Change the value [y]?" will be printed after the editing is finished. Pressing return alone, or anything beginning with "y", will make *ph* change the value; anything beginning with "n" will make *ph* discard the changes.

make *field=value [field2=value2...]*

Allows *ph* users to change those fields in their entry that have the *Change* attribute set. **Make** will set the specified field(s) to the specified value(s) in the entry of the currently logged in user.

add Adds entries to the Nameserver. This is a privileged command.

delete Deletes entries from the Nameserver. This is a privileged command.

logout Undoes the effects of a **login** command.

QUERY EXAMPLES

Here are some examples to clarify *ph* queries. Each example is preceded by a description of the desired effect. It is assumed that the queries are being done with *ph* from the command line, rather than by using the interactive mode of *ph*. The only difference for interactive mode is that metacharacters would not have to be quoted or escaped.

Find the *ph* entry for Paul Pomes:

```
ph paul pomes
```

Find the *ph* entry for P. Pomes, where the rest of the first name is not known:

```
ph p\* pomes
```

Find Alonzo Johnson (or is that JohnsEn?):

```
    ph alonzo johns\?n
or
    ph alonzo johns\[eo\]n
```

Find Paul P., where the rest of the last name is unknown:

```
ph paul p\*
```

The last query fails because it matches too many entries. It is therefore necessary to narrow the search. Suppose it is known that Paul P. has an office in DCL:

```
ph paul p\* address=DCL
```

Alternately, suppose Paul P. works for CCSO. You might try:

```
ph paul p\* department=CCSO
```

When that failed, a good next guess would be:

```
ph paul p\* department=computing
```

The moral of the story is that fields in *ph* generally contain whatever the user wishes them to contain, and, hence, there may be many different spellings and abbreviations for any particular field (some fields are exceptions, including the *name* field, which is always the full name, as known to the University, of the person involved). It pays to make liberal use of metacharacters and creativity when searching fields other than *name*.

Suppose all that is wanted is the full name and electronic mail address of P. Pomes:

```
ph p\* pomes return name email
```

RENAMING PH

If *ph* is invoked with a name other than *ph*, slightly different option processing is done. For the sake of an example, let us assume *ph* was invoked with the name, "unit". The following consequences obtain:

Ph will assume an option of "−t unit". *Ph* will read the *UNIT* environment variable, **after** reading the *PH* environment variable, and **before** reading command-line options.

This feature allows the easy installation of entry-type specific lookup commands, as well as custom configuration of those commands.

BUGS

Separate words in a query are allowed to match the same word in the entry; "`ph s\* smith`" is functionally equivalent to "`ph smith`", because the "s*" is allowed to match "smith".

Ph does some looking about in the commands you give it, but does not understand the full syntax of Nameserver commands. This can occasionally lead to mistakes, especially when dealing with quoted strings.

DISTRIBUTION

Source code for *ph* is available by anonymous ftp to `uxc.cso.uiuc.edu`, in the file `pub/ph.tar.Z`. The complete system, including source for the *qi*(8) server side is in the file `pub/qi.tar.Z`. This source works on 4.[23]BSD UNIX systems. Any troubles encountered porting *ph* to a particular system are of interest to the maintainer of *ph*, as are ports done to other operating systems.

SEE ALSO

The CCSO Nameserver − An Introduction, by Steven Dorner; updated by Paul Pomes.
The CCSO Nameserver − Server−Client Protocol, by Steven Dorner; updated by Paul Pomes.
qi(8)

AUTHOR

Steve Dorner (sdorner@qualcomm.com), Qualcomm, Inc. (formerly at the University of Illinois Computing and Communications Services Office)

The code is now maintained by Paul Pomes (p−pomes@uiuc.edu), University of Illinois Computing and Communications Services Office.