The CCSO Nameserver Server−Client Protocol

by
Steven Dorner s−dorner@uiuc.edu
Computer and Communications Services Office
University of Illinois at Urbana

April 2, 1990

updated by
Paul Pomes paul−pomes@uiuc.edu
Computer and Communications Services Office
University of Illinois at Urbana

August 2, 1992

Introduction

This document describes the protocol used by the CCSO Nameserver. It provides all the information necessary to write a program that interfaces with the Nameserver, or for a human to speak directly to the Nameserver.

While CCSO provides a client program for our Nameserver, we also expect the Nameserver to be used from programs other than this client. First, our client does not run on every system. Secondly, the Nameserver is potentially of use for more than just human lookup of information; other programs (such as mail delivery agents) may wish to use the Nameserver.

This was kept in mind when designing the protocol used by the Nameserver. It is fairly easy to generate and parse (if not totally regular), and should prove easy to incorporate in any program.

General Format

The general format of the protocol is request/response, like that of FTP;[1] the client makes requests, and the server responds to them. The conversation is in "netascii", with a carriage return-linefeed pair[2] separating the lines, as in telnet.[3] This allows a user to use the Nameserver with any telnet client, if they wish.

A request begins with a keyword, and may have zero or more keywords or values, separated by spaces, tabs, or newlines, and followed by a carriage return-linefeed pair. Values containing spaces, tabs or newlines should be enclosed in **double** quotes ('"'). Any printable characters may be used in a quoted string (except '"'). In addition, the sequences "\n", "\t", "\"", and "\\" may be used to mean newline, tab, double quote, and backslash, respectively.

Converted to portable n/troff format using the -me macros from funky Next WriteNow format (icch).

[1] See RFC-959, *File Transfer Protocol (FTP)*, J. Postel and J. Reynolds.

[2] The carriage return is optional.

[3] See RFC-854, *Telnet Protocol Specification*, J. Postel.

Like FTP, numerical values are used to indicate the Nameserver's response to requests. Unlike FTP, data is passed on the same connection as commands. The format for responses is as follows:

result code:[*entry index:*][*field name:*]*text*

Multiline responses should preface each line of the response with the appropriate result code, negated (prefaced with "−"), on all lines of the response but the last. If a particular command can apply to more than one entry, responses involving individual entries will have an entry index directly following the result code. This index will begin with 1, and be incremented each time a new entry is being referred to. Commands that can apply to more than one field will have the name of the field to which the response applies directly following the entry index. The text of the response will be either an error message intended for human consumption, or data from the Nameserver. Whitespace (spaces or tabs) may appear anywhere in the response.

Since more than one specific piece of information may be manipulated by a particular command, it is possible for parts of a command to succeed, while other parts of the same command fail. This situation is handled as a single multi-line response, with the result code changing as appropriate.

As for FTP, numerical responses are in the range 100−599 (or from −599 to −100 for multiline responses), where the leading digit has the following significance:

1: In progress
2: Success
3: More information needed
4: Temporary failure; it may be worthwile to try again.
5: Permanent failure

Specific numbers have meanings to some commands; all commands obey the general scheme.

Many commands generate more than one line of response; every client should be prepared to deal with such continued responses. It is worthwhile to note that a command is finished when and only when the result code on a response line (treated as a signed integer) is greater than or equal to 200.

Keywords must be given in lower case; case in the values of fields is preserved, although queries are not case-sensitive.

The Commands

`query [field=]value. . .  [return field1 [field2]]`
`ph [field=]value. . .  [return field1 [field2]]`

This is the basic client request. It may be used in any of the Nameserver modes.[4] Entries whose fields match the given values will be found, and the requested fields printed. If no field name is specified in the query part of the command, the "name" field is assumed. If no fields are specified with a `return` clause, a default set of fields will be returned. Fields from each entry will be prefaced with an entry index, a colon, the field name, and another colon. If the special field name "all" is given in the `return` clause, all fields from the entry will be printed (subject to normal constraints with regard to Nameserver mode and field properties).

Note that to view some sensitive fields, it is necessary to use Nameserver login mode. Note also that fields whose descriptions include the property Encrypt cannot be printed by the server. Values containing newlines will be broken into lines and printed one line per response.

The second number on each response is the entry index; it is incremented each time the response refers to a new entry.

Some implementations of *qi* return a 102 response before the actual entries, giving the number of entries found; be prepared to see or not see this response.

[4] See *The CCSO Nameserver − A Description*, S. Dorner and P. Pomes, for a description of Nameserver modes.

"Query" and "ph" are synonyms.

Examples

```
query name=dorner phone=244−1765
−200:1:         alias: s−dorner
−200:1:          name: dorner steven c.
−200:1:         email: dorner@garcon.cso.uiuc.edu
−200:1:         phone: (w) 244−1765
−200:1:       address: 181 DCL, MC 256
−200:1:              : 1201 W. Washington, C, 61821
−200:1:    department: computing services office
−200:1:         title: res programmer
−200:1:      nickname: Steve
−200:1:         hours: 8−4 weekdays
200:Ok.
query alias=s−dorner
−200:1:         alias: s−dorner
−200:1:          name: dorner steven c.
−200:1:         email: dorner@garcon.cso.uiuc.edu
−200:1:         phone: (w) 244−1765
−200:1:       address: 181 DCL, MC 256
−200:1:              : 1201 W. Washington, C, 61821
−200:1:    department: computing services office
−200:1:         title: res programmer
−200:1:      nickname: Steve
−200:1:         hours: 8−4 weekdays
200:Ok.
query dorner return alias hours
−200:1:         alias: m−dorner
−508:1:         hours: Not present in entry.
−200:2:         alias: j−dorner
−508:2:         hours: Not present in entry.
−200:3:         alias: s−dorner
−200:3:         hours: 8−4 weekdays
−200:4:         alias: j−dorner1
−508:4:         hours: Not present in entry.
200:Ok.
query alias=s−dorner return id
−503:1:            id: You may not view this field.
200:Ok.
query name=dorner address=moon
501:No matches to your query.
```

change [field=]value. . . make field=value

`Change` looks much like `query`. The entries to be changed are specified as in `query`. They keyword `make` separates the search criteria from the fields to be changed. The change command works in hero mode, or in login mode if applied to fields whose description contain the Change property[5] in the entry of

[5] See *The CCSO Nameserver − A Description*, S. Dorner and P. Pomes, for a description of Nameserver field description properties.

the logged-in user. If it is desired remove a field, Adjacent double quotes ("") should be given as the “new value” of the field. Fields whose descriptions include the property Encrypted must be encrypted before transmission to the Nameserver, unless the *qi* program is being run directly from a terminal. This encryption should be done with the password of the logged in user.

Examples

```
change alias=s−dorner make hours="when the sun shines"
506:You must be logged in to use this command.
change steven dorner make hours=""
200:1 entry changed.
change steven dorner make name="Dr. Strangelove"
−505:name:you may not change this field.
500:1 entry found, none changed.
change ikenberry make email=zzz@xxx
518:Too many entries (3) selected; limit is 2.
change stanley ikenberry make email=zzz@xxx
−510:s−ikenberry:You may not change this entry.
500:1 entry found, none changed.
```

`login` `alias`
`answer` `code`
`clear` `password`

This is used to enter login or hero mode. The Nameserver will respond with a random challenge, which may be returned in encrypted form via the `answer` command. The encryption key will be a password known to both the Nameserver and the user. Alternately, the client may respond with the `clear` command, and give the proper password in clear text. This is not the recommended method, and is only provided for the lazy protocol implementor.

Examples

```
login s_dorner
301:dkeiigjasdvvnmnmeigh
answer ewituegndvbngkgdfkgl
200:s−dorner:Hi how are you?
login s−dorner
301:?;_?DB,F9X;8O=H8Y<H[H=FY?1*;>?#(ˆ=’<!HH
answer ellwekkewdfasoiioiogdfkldfg
500:Login failed.
login s−dorner
301:aksjdsflkajajeruopqwda,mcdfkklqopakdjl
clear mysecret
200:s−dorner:Hi how are you?
```

`logout`

Exits login or hero mode, entering anonymous mode. The connection is not closed, however.

Example

```
logout
200:Ok
```

fields [field...]

With no arguments, lists all field descriptions. With field names as arguments, descriptions of the named fields are given. The second number of each response is the field id number.

Example

```
fields
-200:6:alias:max 32 Indexed Lookup Public Default Change Turn
-200:6:alias:Unique name for user, chosen by user.
-200:3:name:max 64 Indexed Lookup Public Default
-200:3:name:Full name.
...
-200:24:def_account:Default account for printing
200:Ok.
```

add field=value...

Creates a nameserver entry with the given fields. Note that this command adds an **entry**, not a field; to add a field to an existing entry, the `change` command should be used. Hero mode is required.

Examples

```
add name="churchill winston" address="england"
511:You may not add Nameserver entries.
add name="churchill winston" address="england"
200:Ok.
```

delete [field=]value...

Deletes one or more nameserver entries. Note that this command deletes an **entry**, not a field; to remove a field from an existing entry, the `change` command should be used. Hero mode is required.

Example

```
delete winston churchill
200:1 entries deleted.
```

siteinfo

Returns information about the server's site. Some servers will provide more information than others; clients should be ready for specific items to be missing, or for the whole command not to be implemented.

Defined data items are:

Maildomain – domain to use for phquery-type mail.
Mailfield – field to use for phquery-type mail.
Administrator – guru in charge of service.
Passwords – person in charge of ordinary password/change requests.

Example

```
siteinfo
-200:1:maildomain:uiuc.edu
-200:2:mailfield:alias
-200:3:administrator:s-dorner@uiuc.edu
-200:4:passwords:nameserv@uiuc.edu
200:Ok.
```

set option[=value]...

Sets an option for this nameserver session.

Examples

```
set verbose=off
200:Done.
set language=french
-513:language:unknown option
513:No option recognized.
```

id information

Enters the given information in the Nameserver logs. This command is used by the CCSO Nameserver client to enter the user id of the person running it.

Example

```
id 103
200:Thanks.
```

status

Prints the current status of the Nameserver.

Examples

```
status
200:Database ready.
status
201:Database ready, read-only.
```

help [{native|client} [topic...]]

Prints help files for the Nameserver. If *client* is specified, it should be a valid Nameserver client identifier, such as “ph”. The client-specific help will first be searched for *topic*, and then the native help will be searched. If *topic* is omitted, a list of all available help texts will be returned. If "native" or *client* are also omitted, a list of clients will be returned.

The second number of each response is the help text index; this number is incremented each time a new topic text is being printed.

Examples

```
help
-200:1:The following clients have help:
-200:1:native   ph
```

```
200:Ok.
help native
-200:1:These "native" help topics are available:
-200:1:100              401              505
...
200:Ok.
help ph
-200:1:These "ph" help topics are available:
-200:1:edit             login            password         uiuc.general
...
-200:2:These "native" help topics are also available:
-200:2:site        policy
...
200:Ok.
help ph add
-200:1:add:
-200:1:  SYNTAX: add name-of-field=value-of-field...
...
200:Ok.
```

quit
stop
exit

Ends a nameserver session. *Qi* will break the connection. These commands are synonymous with one another.

Example

```
quit
200:Bye!
```

APPENDIX A

Command Summary

```
query [field=]value...  [return field...]
ph [field=]value...  [return field...]
change [field=]value...  make field=value...
login alias
answer code
clear password
logout
fields [field...]
add field=value...
delete [field=]value...
set option[=value]...
id information
status
siteinfo
help [{native|client} [topic...]]
quit
exit
stop
```

APPENDIX B

Result Codes

100	In progress (general).
101	Echo of current command.
102	Count of number of matches to query.
200	Success (general).
201	Database ready, but read only.
300	More information (general).
301	Encrypt this string.
400	Temporary error (general).
401	Internal database error.
402	Lock not obtained within timeout period.
475	Database unavailable; try later.
500	Permanent error (general).
501	No matches to query.
502	Too many matches to query.
503	Not authorized for requested information.
504	Not authorized for requested search criteria.
505	Not authorized to change requested field.
506	Request refused; must be logged in to execute.
507	Field does not exist.
508	Field is not present in requested entry.
509	Alias already in use.
510	Not authorized to change this entry.
511	Not authorized to add entries.
512	Illegal value.
513	Unknown option.
514	Unknown command.
515	No indexed field in query.
516	No authorization for request.
517	Operation failed because database is read only.
518	Too many entries selected by change command.
520	CPU usage limit exceeded.
521	Change command would have overriden existing field, and the "addonly" option is on.
522	Attempt to view "Encrypted" field.
523	Expecting "answer" or "clear"
524	Names of help topics may not contain "/".
598	Command unknown.
599	Syntax error.