Network Working Group
Request for Comments: 2378
Category: Informational

R. Hedberg
Umea University
P. Pomes
QUALCOMM, Inc.

September 1998

The CCSO Nameserver (Ph) Architecture

Status of this Memo

Copyright Notice

Abstract

The Ph Nameserver from the Computing and Communications Services Office (CCSO), University of Illinois at Urbana-Champaign has for some time now been used by several organizations as their choice of publicly available database for information about people as well as other things. This document provides a formal definition of the client-server protocol. The Ph service as specified in this document is built around an information model, a client command language and the server responses.

1. Overview

1.1. Basic Information Model

At its simplest the Ph database can be thought of as a computer-resident "phone book". However, it can be used to collect arbitrary information about people, and in response to a query about an object named in the database, return information about that entity. It is in short a nameserver for people and objects. It was designed to keep a relatively small amount of arbitrary information about a relatively large number of people or things, and provide access to that information over the Internet. In order to structure the information the manager of the database has to decide which views to present of the real-world objects that are to be represented in the database. Each view is then composed of a number of fields and their values. To support this concept Ph has the notion of named information, i.e., categorizing information into what are called fields and assigning descriptive names to those fields.

Even if the database resides and is reachable from the Internet it is local in the meaning that no server is supposed to be able to refer a client to another server which might hold the wanted information. However a server may contain a list of other Nameservers which can be used by clients to query other Nameservers for information.

1.1.1. Fields

A field descriptor is associated with each field and is used to describe the type and behavior of the field. A field descriptor includes the fieldname, the maximum length of information the field can store before truncation, keywords describing the properties of the field as well as free text describing what kind of information the field is supposed to hold.

The keywords can be any of the following:

Always: Forces the field's contents to be always printed in addition to whatever fields specified by the query.

Any: This field is always searched by queries. To be most use ful, a field marked as Any should also have the Indexed and Lookup keywords as well.

Change: Can be changed by the owner of the entry.

Default: Printed if no return clause is given in the query.

Encrypt: Must be encrypted before transmission.

ForcePub: Viewable/searchable regardless of the content of the suppress field

Indexed: Fields that are kept track of in the database's index for efficient lookups. At least one indexed field must be present in each query.

LocalPub: May be viewed by anyone in the "local" domain or address space. Fields with this keyword are completely invisible outside of the "local" domain. They will not be shown with the fields command (section 3.3), and are disallowed in query commands or return clauses (section 3.8).

Lookup: May be used in the selection part of a query. A Field without this keyword may not be used to select entries.

NoMeta: Wildcard searches are disallowed.

NoPeople: No entry of type "person" may include this field.

Private: Field may be viewed by Heros (section 1.4) only.

Public: May be viewed by anyone. Fields not marked with this keyword may only be viewed by the entry's owner or a Hero.

Sacred: Changes to the field are prohibited except via non-network invocations of the server, i.e., from a tty, file, or pipe.

Turn: Users may turn off visibility of a field to everyone except themselves and Heros by prefixing the field text with '*'.

Unique: Any change to the field will be rejected if the change causes the modified field to match the same field in any other entry.

1.1.2. Character Sets

Historically Ph has been restricted to only handle printable characters, that is characters with hexadecimal values between 0x20 and 0x7f. Lately with the spreading of 8-bit clean Operating Systems there is no reason to keep this limitation.

This document therefore proposes that ISO-8859-1 shall be regarded as an alternative character set for Ph, the default still being US-ASCII.

Clients that utilize ISO-8859-1 should request that the server return ISO-8859-1 by using the "set"-command.

In the instance that values are stored using ISO-8859-1 and are to be shown to a client expecting US-ASCII, the characters with character codes outside of the US-ASCII range should be displayed in the "Quoted-Printable" content-transfer-encoding form defined in RFC-2045 [MIME].

1.2. Standardization issues

Each Nameserver manager is in essence free to name new fields to suit the special needs of his/her organization. But in order to make the directory service useful outside of the organization it is recommended that a core set of standard fields always should be present.

Therefore this document defines a couple of standard collections of fields (Appendix A).

Also note that the architecture makes no assumption about the search and retrieval mechanisms used within individual servers. Operators are thereby free to use any kind of dedicated databases, fast indexing software or even gateways to other directory services to store and retrieve the information, if desired.

Ph simply functions as a known front-end, offering a simple data model in addition to a well known port and simple query language.

1.3. Conventions Used in this Document

In examples, "C:" and "S:" indicate lines sent by the client and server respectively.

1.4. Heros

For Ph a Hero is equivalent to a superuser or operator. Being in Hero mode means that some or all artificial limits are removed; full Heros may change any field in any entry in the database, as well as view as many entries as they wish. Heros can also be limited to one field of one other entry. Hero mode is used mostly for administrative purposes, delegation of group authority over selected fields, and is controlled by the acl field.

2. Basic Operation

Initially, the server host starts the Ph service by listening on TCP port 105. When a client host wishes to make use of the service, it establishes a TCP connection to the server host. The client and the Ph server then exchange commands and responses (respectively) until the connection is closed or aborted.

2.1. Command syntax

Commands in Ph consist of a keyword optionally followed by zero or more keywords or values, separated by spaces, tabs or newlines, and followed by a carriage return-linefeed (CRLF) pair. A more thorough description using BNF is given in Appendix C.

Values containing spaces, tabs or newlines must be enclosed in double quotes ('"'). In addition the sequences "\n", "\t","\"" and "\\" may be used to mean newline, tab, double quote and backslash, respectively.

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

2.2. Response syntax

Responses consist of a result code followed by additional information possibly separated by entry index and/or field name and are terminated by a CRLF pair.

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

Responses to some commands might be multi-lined. In these cases each line in the response, except the last, has the appropriate result code negated (prefaced with "-"). The last line then starts with the appropriate result code without negation. Each line must be terminated by a CRLF pair.

If a particular command can apply to more than one entry, then the multilined response must be so organized that all information pertaining to each entry is returned on consecutive lines, and that each of those lines must have one and the same entry index directly following the resultcode. The first entry index should be 1 and incremented each time a new entry is referred to.

```
C: query hedberg return email name title
S: 102:There were 3 matches to your request.
S: -200:1:        email: canheg95@student.umu.se
S: -200:1:         name: Carl Johan Hedberg
S: -200:1:        title: Student
S: -200:2:        email: parheg95@student.umu.se
S: -200:2:         name: Par Hedberg
S: -200:2:        title: Student
S: -200:3:        email: Roland.Hedberg@umdac.umu.se
S: -200:3:         name: Roland Hedberg
S: -200:3:        title: Boss of the Network group
S: 200:Ok
```

Commands that can apply to more than one field must 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 in human readable format, or data from the Nameserver. Whitespace (spaces or tabs) may appear anywhere in the response, but the field name and text columns if present must each begin with a whitespace character.

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, the result codes are in the range 100-699 (or from -699 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 worthwhile to try again.
5: Permanent failure
6: Phquery specific codes

Many commands generate more than one line of response; every client should be prepared to deal with such continued responses. 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.

Clients should assume that any numeric response, within the above mentioned ranges, are valid. Also note that the server is allowed to send one or more lines with result codes between -199 - -100 (the leading "-" indicates a continuation line) and 100 - 199, as status information, before the actual results are transmitted.

2.3. Format of a search string

Matching is not sensitive to upper or lower case letters and is normally done on a word-by-word basis. That is, both the query expression and the entry information is broken up into words, and individual words are compared using exact matching. If the order of the words is important in a query, then the query string can be surrounded by '"' (double quotes), whereby the complete search string is matched against the information in the Nameserver database.

Word delimiters are the following characters: <SPACE>, <TAB>, <NEW-LINE>, ",", ";" and ":" . These characters are not indexed and should not be part of the search string.

However, special symbols, called "wildcard" characters, can be used if the exact spelling is unknown. The '*' (asterisk, 0x2A) is used in place of zero or more characters, '+' (plus, 0x2B) in place of one or more unknown characters, and '?' (question mark, 0x3F) can be used when exactly one character is unknown. If the unknown character can be one of a limited set this can be specified by surrounding the set with brackets, e.g., [ei] means that in that place an 'e' or an 'i' would match.

3. Commands

3.1. status

status

Prints the message of the day and the current status of the nameserver.

```
C: status
S: 100:Qi server $Revision: 1.6 $
S: 100:Ph passwords may be obtained at CCSO Accounting,
S: 100:1420 Digital Computer Lab, between 8:30 and 5 Monday-Friday.
S: 100:Be sure to bring your U of I ID card.
S: 200:Database ready
```

3.2. siteinfo

siteinfo

Returns information about the servers site. Possible fields are

Version	Version information for the server.
Maildomain	The mail domain to use for phquery-type mail.
Mailfield	The field containing the specific email address.
Mailbox	Mandatory entry that names the field to use as maildrop.
Administrator	Guru in charge of service.
Passwords	Person in charge of ordinary password/change requests.
Authenticate	Authentication methods supported by the server, ordered in the site-preferred way. Presently the following options are defined:

1 attempt auto login
2 allowed to be interactive if needed
4 use ANSI X9.9 challenge/response
8 use v4 Kerberos login
16 use v5 Kerberos [KRB5] login
32 use GSS-API [GSS-API] login
64 use email login
128 password encrypted response to challenge
256 use clear-text password
512 use HMAC [HMAC] with SHA-1 of challenge string

Example

```
C: siteinfo
S: -200:1:version:3.1
S: -200:2:maildomain:umu.se
S: -200:3:mailfield:alias
S: -200:4:mailbox:email
S: -200:5:administrator:roland.hedberg@umdac.umu.se
S: -200:6:passwords:roland.hedberg@umdac.umu.se
S: -200:7:authenticate:64:32:128
S: 200: Ok.
```

The mail fields in the siteinfo command direct how address information stored in the Nameserver is to be used for delivering mail.

The specific (username, host) pair to where a user's mail should be sent for final delivery is stored in the field named by {mailbox}. Phquery and like utilities will use this field.

To construct a useable email address from Nameserver information, the algorithm below is followed:

```
if ({maildomain} is not null) then
     address = (contents of {mailfield})@{maildomain}
else
     address = (contents of {mailfield})
```

Some existing client software will not format email addresses correctly if the value of {mailbox} is set to anything other than "email" when {maildomain} is non-empty.

If {mailbox} is set to anything other than {email}, {maildomain} must be reported empty by the siteinfo command. Also reformatting of each record's {mailfield} must be done by the server before reporting it to the client.

3.3. fields

fields [field ...]

Without an argument, a list of all available field descriptors should be delivered. Any space-separated argument(s) restricts the list to the named fields. Fields marked with the "LocalPub" keyword (section 1.1.1) should not be delivered outside of the local domain.

The output of the command consists of two lines describing each field. The first line defines the field in technical terms (max length and field attributes), while the second line is a brief description of what the field is intended to hold. The second number of each response is the field id number.

```
C: fields
S: -200:6:alias:max 32 Indexed Lookup Public Default
S: -200:6:alias:Unique name for user.
S: -200:3:name:max 64 Indexed Lookup Public Default
S: -200:3:name:Fullname
S: -200:2:email:max 128 Lookup Public Default
S: -200:2:email:Account to receive electronic mail.
S: -200:16:other:max 256 Lookup Public Default Change
S: -200:16:other:Other info the user finds important.
S: -200:33:home_phone:max 60 Lookup Public Change Turn
S: -200:33:home_phone:Home telephone number.
S: 200:Ok.
```

3.4. id

id information

Enters the given information in the Nameserver's log. This command is used by the Ph client to enter the user id of the person running it.

3.5. set

set [option[=value] ...]

Sets the named option for this nameserver session to a value. The default string "on" is used if no value is supplied. Used without arguments it return the settable options and their current value. Some common options are

echo	If on, echo the client's commands back to the client.
limit	Changes that affect more than the specified number of entries results in an error.
charset	Return responses to the client in the character set specified.
verbose	If on, report interim progress messages to the client.
addonly	If on, change commands can only create fields in entries, not modify them.
nolog	If on, disable logging.
external	If on, make Fields marked as "LocalPub" invisible.

Example

```
C: set verbose=off
S: 200:Done.

C: set
S: -200:echo:off
S: -200:limit:2
S: -200:charset:iso-8859-1
S: -200:verbose:off
S: -200:addonly:off
S: -200:nolog:off
S: -200:external:on
S: 200:Done.
```

3.6. login, logout, answer, clear, email, and xlogin

3.6.1. login

login [alias]

The "login" command allows client users to identify themselves to the Nameserver. More specifically it identifies a client user with a particular entry in the Nameserver and allows them to change fields in that entry and possibly other entries. It is also necessary to be logged in to the Nameserver to view certain sensitive fields in the user's own entry.

In order to use the "login" command the client must prompt the user for their ph alias and password. The client is then responsible for (optionally) encrypting the password and sending it to the server. This will be covered in sections 3.6.3 (answer) and 3.6.4 (clear).

```
C: login foo
S: 301:,:P"_Y$ONU%"SDUQ6&^`ZZ'?*#Y`A_.Z/A>?@SH>*-
```

3.6.2. logout

logout

The "logout" command allows a user who is logged in to the Nameserver to logout.

C: logout S: 200:Ok.

3.6.3. answer

answer encrypted-response

In response to the login command, the Nameserver responds with a random challenge string. The Nameserver client encrypts the challenge with the password supplied by the user, uuencodes the result into US-ASCII, and returns the printable result in the "answer" command:

```
C: login ppomes
S: 301:.%$&.D^67$*1?<.2S@DR:Z@M*)AV-<:4QM>#R>M*HT
C: answer M5K'F:NI(a?M?O2+-a9`48RA#ZF=L9)G)/XRS7Q^0>0@-R7X$WGb`50B]
S: 200:ppomes:Hi how are you?
```

The encryption algorithm is based on a three rotor Enigma engine. There are known attacks on the security of this approach.

The answer command is also used to return method-specific responses to the xlogin command (section 3.6.6).

3.6.4. clear

clear cleartext-password

The "clear" command can be used instead of the "answer" command to complete a login sequence. It's argument is the user's cleartext password. This command is supplied only to support those clients that have not implemented one of the encryption engines used by the "answer" command. It's use is strongly discouraged.

```
C: login ppomes
S: 301:E=@Y&VW^_9YVI;D5.[EB0:B)9Z#_&X$:2)/eL$VJC87
C: clear MySecret
S: 200:ppomes:Hi how are you?
```

3.6.5. email

email local-userid

The "email" command can also be used instead of the "answer" command to complete a login sequence. The value of local-userid is the user's login name on the local machine. If all of the following conditions are true, then the email command will be accepted by the server:

1) The connection to the server originates on port 1023 or less on the client. Note: This is a system port. Port 1023 is not allocated to this use.

2) The canonical name of the client's host matches the right-hand side of the email address of the requested alias specified in the "login" command.

3) The "local-userid" matches the left-hand side of the email address belonging to the requested alias.

This is a weak but convenient form of authentication. Depending on the information users are allowed to change about themselves and the threat environment the server operates in, this method may be appropriate. Servers should take care to avoid DNS spoofing.

3.6.6. xlogin

xlogin option alias

Extended login command for GSS, Kerberos v4 and v5, ANSI X9.9 token devices (e.g., SNK/4), etc. The option is one of the values returned in the Authenticate field of the "siteinfo" command (section 3.2). Alias is the user's alias.

```
C: xlogin 16 ppomes
S: 301:DoKrbLogin started; send Kerberos mutual authenticator.
C: answer MJa8QO1cJHYz2IdWyg7uhAnixVqgCZQBWr64ciXYku1ktdu....
S: 200:ppomes:Hi how are you?

C: xlogin 4 ppomes
S: 302:SNK Challenge "024142":
C: answer 82344338
S: 200:ppomes:Hi how are you?
```

The answer command returns the requested quantity, Kerberos authenticator, X9.9 device response, etc. Binary quantities are first uuencoded into US-ASCII.

3.7. add

add field=value...

This command is used to add new entries to the database. You must be logged in and have full Hero privileges (section 1.4) to use "add".

```
C: add name="doe john" id="123456789" alias="j-doe"
S: 200:Ok.
```

3.8. query

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

If no field is specified together with a value then the field is assumed to be "name" and/or "nickname". When more than one field-value specification are given in a query, entries matching all specifications are returned (implicit AND).

It is possible to define which fields should be returned by adding a "return" clause. If no return clause is defined the Ph server will return a default list of fields. Typical default fields are "alias", "name", "title", "email", "phone", "address", "department", "www", and "other". A return clause consists of the word "return" followed by a list of fields or the word "all". If the word "all" is used then all viewable fields will be returned.

```
C: query name=doe name=john
S: 102:There was 1 match to your request.
S: -200:1:            alias: j-doe
S: -200:1:             name: doe john
S: 200:Ok.
```

3.9. delete

delete [field=]value...

This command is used to delete entire entries from the database. You must be logged in and have full Hero (section 1.4) privileges to use "delete".

The arguments to the "delete" command are the same as the selection part of a "query" command. "Delete" finds all the entries that match the argument(s) and deletes them.

The "delete" command obeys the Nameserver "limit" option, which can be used to prevent deletion of more entries than intended.

```
C: delete name="doe john" id="123456789" alias="j-doe"
S: 200:1 entries deleted.
```

3.10. change

change [field=]value [make|force] field="value"...

This command is used to change one or more fields in one or more entries to the values specified. The "change" command consists of two clauses, the "change" clause and the "make" or "force" clause.

The "change" clause determines which entries will be affected by the command. It uses the same arguments as the selection clause of a "query" command. The "make" or "force" clause specifies which field(s) will be changed and the new value(s) of the specified field(s). The "force" clause is only used to make non-encrypted changes to fields marked "Encrypt".

You must be logged in to use "change".

The "change" command obeys the Nameserver "limit" option, which can be used to prevent changing the field contents of more entries than intended.

```
C: change alias=j-doe force password=NewSecret
S: 200:1 entry changed.

C: set limit=500
S: 200:Done.
C: change fax="(619) 555-1212" make fax="(760) 555-1212"
S: 200: 113 entries changed.
```

3.11. help

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

Prints help on the Nameserver or on specific clients. 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.

```
C: help native 101
-200:1:101:
-200:1: The Nameserver echo option is set. The text of this response is
-200:1: the command you just gave, which has not (yet) been executed.
200:Ok.
```

3.12. quit/exit/stop

```
quit
```

Terminates the session with the Nameserver and causes the client to exit.

```
C: quit
S: 200:Bye!
```

4. Security

4.1. Transport Layer

In the absence of encryption between client and server, all Nameserver traffic is unsecure. Kerberos v4, v5, and the GSS-API all provide encryption mechanisms, however the Nameserver protocol does not support the means to negotiate encryption between client and server. This implies that all traffic can be seen by other machines having access to the network linking the client and server. Furthermore clear-text traffic is subject to modification in transit between client and server. Possible ways of augmenting this would be to use something like TLS [TLS] or IPSec [IPSEC].

4.2. Server Authentication

Unless one of the mutual authentication mechanisms is used, e.g., Kerberos 4/5 or GSS-API, there is no way to prove the identity of a server. Further, there is no mechanism to prove a given server is authoritative for a set of information.

4.3. Secure User Authentication

The Ph protocol allows the negotiation of several authentication protocols between client and server, some weak and some strong. It does not prohibit the use of cleartext passwords, something which should be depreciated, but is useful when dealing with some clients.

4.4. Privacy and Access Lists

Directory services like the CCSO white pages server that contain information on persons have to consider privacy issues. This paper describes one way of partitioning specific attributes from unwanted access by designating them visible only to the "local" community, visible only to the person connected with the information, or visible only to the database administrator.

4.5. References

[GSS-API] Linn, J., "Generic Security Service Application Program Interface, Version 2", RFC 2078, January 1997.

[HMAC] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997.

[IPSEC] Atkinson, R., "Security Architecture for the Internet Protocol", RFC 1825, August 1995.

[KRB5] Kohl, J., and C. Neuman, "The Kerberos Network Authentication Service (V5)", RFC 1510, September 1993.

[TLS] Dierks, T., and C. Allen, "The TLS Protocol, Version 1.0", Work in Progress.

[MIME] Freed, N., and N. Borenstein, "Multipurpose Internet Mail Extensions, (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996.

5. Miscellaneous

5.1. Authors' Addresses

Roland Hedberg
Umdac
Umea University
901 87 Umea
Sweden

EMail: Roland.Hedberg@umdac.umu.se

Paul Pomes
Qualcomm Inc
6455 Lusk Blvd
San Diego, CA
USA

EMail: ppomes@qualcomm.com

Appendix A

Default fields and suggested lengths connected to different object types.

```
All entries:  Information common to all entries
type              64
name              256
address           128
proxy             32
password          32

type=phone:   Information found in a phonebook
phone             64
fax               64

type=person:  Information about a human being
alias             32
forename          64
surname           64
group             32
email             128
public_key        4096
nickname          128
www               256
acl               128

type=staff:   Information about an employee
empno             16
department        64
supervisor        64
secretary         64
office_location   128
office_address    128
office_phone      64
title             64
pager             64
hours             128

type=unit:   Information about an organizational unit
email             128
www               256
public_key        4096
```

Appendix B

Result codes

100 In progress (general).
101 Echo of current command.
102 Count of number of matches to query.
103 No hostname found for IP address.
200 Success (general).
201 Database ready, but read-only.
300 More information (general).
301 Encrypt this string.
302 Print this prompt.
400 Temporary error (general).
401 Internal database error.
402 Lock not obtained within timeout period.
403 Login would have been OK, but database read-only
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 To many entries selected by change command.
520 CPU usage limit exceeded.
521 Change command would have overridden 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 "/".
525 Email authentication failed
526 Host name address not found in DNS
527 Reverse DNS lookup does not match forward DNS lookup
528 General Kerberos database error.
529 Selected authentication method not available

590 Remote queries not allowed.
598 Command unknown.
599 Syntax error.
600 Ambiguous or multiple match

Appendix C

Description of the client command language using the augmented Backus-Naur Form (RFC822).

```
response = code [index] [field] text CRLF

code     = [-] LDIG 2DIGIT ":"
index    = number ":"
field    = 1*SPACE attribute ":" 1*SPACE
text     = 1*( CHAR / LWSP-char )


command     = ph-command CRLF

ph-command  =  "status" / a-command / oa-command
ph-command  =/ av-command / answer-command / query-command
ph-command  =/ delete-command / change-command / "help" / quit-command

a-command       = ("siteinfo"/"fields"/"id"/"login"/"help"/"email"/
           "clear") [attribute]
oa-command      = ("xlogin") number attribute
av-command      = ("set"/"add"/"make") 1*attribute-value
answer-command  = ("answer") 1*printable
query-command   = ("query"/"ph") 1*selection ["return" 1*attribute]
quit-command    = "quit" / "exit" / "stop"
change-command  = "change" 1*selection make 1*attribute-value
delete-command  = "delete" selection

selection       = value / attribute-value

attribute-value = attribute "=" value

value           = 1*(cstring / quoted-string / set)

cstring         = 1*( ALPHA / DIGIT / S_SPEC / set / quoted-pair )
attribute       = 1*( ALPHA / DIGIT / "_" / "-" )
number          = 1*(DIGIT)

quoted-string   = <"> 1*(qtext/quoted-pair) <">

quoted-pair  =  "\" CHAR
qtext        = 1*( CHAR / CR / SPEC1 / DELIMIT1 / DELIMIT2 / LWS )
set          = '[' 1*(ALPHA/DIGIT) ']'

LWSP-char    = SPACE / HTAB
LWS          = 1*([CRLF] (LWSP-char))
CRLF         = CR LF
```

```
   S_SPEC       = '*'/'+'/'?'
   SPEC1        = "=" / "*" / "?" / "+" / "[" / "]"
   SPEC2        = "\" / """
   DELIMIT1     = SPACE / HTAB / LF
   DELIMIT2     = "," / ";" / ":"
   PRINTABLE    = %d32..%d126
   CTL          = %d0..%d31 / %d127..%d160
   ALPHA        = %d65..%d90 / %d97..%d122
   DIGIT        = %d48..%d57
   LDIG         = %d49..%d54
   SPACE        = %d32
   SEP          = (CR LF) / LF
   CR           = %d13
   LF           = %d10
   HTAB         = %d9
   CHAR         = %d33..%d126 / %d160..%d255
   OTHER        = "(" / ")" / "-" / "." / "/"
          "@" / "$" / "_" / "!" / "~" /
          "'" / "#" / "&" / "<" / ">" /
          "^" / "`" / "{" / "|" / "}"
```

Full Copyright Statement