CS 580 Client-Server Programming
Fall Semester, 2002
Doc 14 Client-Server Protocol & POP

Contents

Protocol...........................................................................................................2
Well defined......................................................................................................3
Complete........................................................................................................4
Parseable........................................................................................................5
Available.........................................................................................................6
Protocol Types..............................................................................................7
Protocol Design Issues..................................................................................8
POP3 Protocol..............................................................................................9
Command Format..........................................................................................10
States.............................................................................................................12
Commands....................................................................................................13
Optional POP3 Commands ..........................................................18

Reference


CS 580, Spring 1997 lecture notes,
http://www.eli.sdsu.edu/courses/spring97/cs596/notes/protocol/protocol.html

Copyright ©, All rights reserved. 2002 SDSU & Roger Whitney, 5500 Campanile Drive, San Diego, CA 92182-7700 USA. OpenContent (http://www.opencontent.org/pl.shtml) license defines the copyright on this document.
Protocol

Communication between client and server

Good protocols are hard to design

Requirements for a "good protocol":

• Well defined
• Complete
• Parsable
• Extendable
• Available protocol document
Well defined

Every bit of data sent in either direction has to have its place in the protocol description.

Protocol is a Language

Common formal description:

BNF and Augmented BNF

Format of the description language needs to be part of the protocol document.

Examples are important
Complete

The protocol must cover all possible situations.

• Garbage data
• Old client or server (different protocol versions)
• Illegal requests
• Boundary conditions
• Etc.
Parsable

Both clients and servers are computer programs.

A computer program's IQ is generally 0.

Design goals:

• Distinct information packets or messages
  
  Allow parsing independent of semantics

• Consistency
  
  Allow for code reuse

• Flexibility
  
  For example name-value pairs
Available

Different groups may write clients and servers at different times.

Central registry for Internet protocols

Self regulating:

• RFC - Request For Comment
• IETF - Internet Engineering Task Force

Official:

• ISO
• ANSI
Protocol Types

Two basic types

- Synchronous
- Asynchronous

Typical synchronous

- Client sends request to server
- Server responds with a reply

Examples

HTTP, POP, SMTP, GOPHER, XMODEM

Typical asynchronous

Client and server both send information to each other concurrently.

Examples

TELNET, RLOGIN, ZMODEM

A hybrid protocol is also possible
Protocol Design Issues

Protocol design is difficult!

Learn from examples

Some issues

• Protocol extendibility and versioning

• Byte order used for sending values

• ASCII vs. Binary protocol
  
  Easy of debugging
  Efficiency

• Synchronous vs. Asynchronous

  Protocol overhead
  Roundtrip delays

• State

  Who is writing, who is reading?

• Timeouts

  Timeouts vs. Synchronous protocols
POP3 Protocol

Purpose: Allow PC's, Macs, etc. to download mail from server

Port number 110

Protocol uses ASCII only
Command Format

Format of commands to server

keyword blank argument₁ [ blank argumentₖ ] CRLF

| keyword | = 3, 4 characters
| argument | <= 40 characters

keyword and arguments are separated by single space character

Server Response

Status keyword additionalInfo

Status is either "+OK" or "-ERR0.3."

A single line response ends in CRLF

If response requires more than one line:

- Each line ends in a CRLF
- The response ends in CRLF.CRLF
- If a line starts with a "." prepend a "." to it
Timeouts

A POP3 server may have an autologout timer

A server must wait at least 10 minutes before timing out a client

The POP3 server on cs.sdsu.edu times out in 2 minutes
States

AUTHORIZATION

Must log in with password before entering transaction state

TRANSACTION

Client can request actions of server, get mail for example

UPDATE

Updates mail box to reflect actions taken in transaction state
Commands

AUTHORIZATION

Server acknowledges connection from client with

+OK "message"

+OK UCB Pop server (version 2.1.2-R3) at sciences.sdsu.edu starting.

Commands: USER, PASS, APOP, QUIT
USER PASS

Combination is used to progress to transaction state

USER must come first
PASS or QUIT must come after USER

Example
  Ti 38->telnet cs.sdsu.edu 110
  Trying 130.191.226.116...
  Connected to cs.sdsu.edu.
  Escape character is '^]'.
  +OK QPOP (version 3.1.2) at sciences.sdsu.edu starting.
  USER whitney
  +OK Password required for whitney.
  PASS typeYourPasswordHere
  +OK whitney has 116 visible messages (0 hidden) in 640516 octets.
TRANSACTION

Commands: STAT, LIST, RETR, RSET, QUIT

STAT

Arguments: none
Returns "+OK" numberOfMessages SizeOfMail
Example

STAT
+OK 22 45595

LIST

Arguments: a message-number ( optional )
Returns: size of message in octets
Examples:

LIST 2
+OK 2 3064

LIST
+OK 116 visible messages (640516 octets)
1 2980
2 3064 ( message 3 - 116 deleted to save space )
116 1290
.
RETR

Arguments: a message-number
Returns: the message
Example:

RETR 21
+OK 825 octets

this is a test
..
the end
---
Roger Whitney       Math & Computer Science Dept.
whitney@cs.sdsu.edu   San Diego State University
http://www.eli.sdsu.edu   San Diego, CA 92182-7720
(619) 594-3535
(619) 594-6746 (fax)
**DELE**

Arguments: a message-number to delete  
Returns: a confirmation of deletion  
Marks a message to be deleted

**NOOP**

Arguments: none  
Returns: a positive response  
Does nothing

**QUIT**

Arguments: none  
Returns: a positive response  
Send POP3 server to UPDATE state

**UPDATE State**

Updates mail box to reflect transactions taken during the transaction state, then logs user out

If session ends by any method except the QUIT command during the transaction state, the update state is not entered
Optional POP3 Commands

TOP

Arguments: a message-number and number of lines to return
Returns: Requested lines of indicated message
State allowed in: transaction

UIDL

Arguments: a message-number (optional)
Returns: a unique-id listing for message
State allowed in: transaction

Examples:

UIDL 1
+OK 1 826312760.001

UIDL
+OK uidl command accepted.
1 826312760.001
2 826312760.006
3 826493796.004
e tc.
.
APOP

Arguments: a mailbox and a MD5 digest string

State allowed in: authorization

Action: If MD5 string is correct move to transaction state