POPd 1.00BETA documentation

COLLABORATORS			
	TITLE : POPd 1.00BETA documentation		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		January 8, 2025	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

POPd 1.00BETA documentation

1.1 POPd 1.00BETA

POPd 1.00BETA

POP3 daemon for AmiTCP

Copyright ©1994 by Jonathan Gapen

1.2 introduction

** NOTE ** POPd 1.00BETA is a beta release. It has only been tested on my A4000/030 running AmigaOS 3.0 and AmiTCP 3.0b2. A number of bugs may still remain, though they haven't appeared while in use on my system. POPd has been tested with AmiPOP 1.11 and NuPOP on MS-DOS. I'm releasing it to the Amiga community since I have no other means of testing.

POPd 1.00 is an implementation of the server end of the POP3 protocol for AmiTCP 3.0b2. The POP3 protocol is defined in Internet document RFC1081 by Marshall Rose in November of 1988.
The need for POP3 is outlined in the introduction to RFC1081, as follows:

Introduction

On certain types of smaller nodes in the Internet it is often

impractical to maintain a message transport system (MTS). For example, a workstation may not have sufficient resources (cycles, disk space) in order to permit a SMTP server and associated local mail delivery system to be kept resident and continuously running. Similarly, it may be expensive (or impossible) to keep a personal computer interconnected to an IP-style network for long amounts of time (the node is lacking the resource known as "connectivity").

Despite this, it is often very useful to be able to manage mail on these smaller nodes, and they often support a user agent (UA) to aid the tasks of mail handling. To solve this problem, a node which can support an MTS entity offers a maildrop service to these less endowed nodes. The Post Office Protocol - Version 3 (POP3) is intended to permit a workstation to dynamically access a maildrop on a server host in a useful fashion. Usually, this means that the POP3 is used to allow a workstation to retrieve mail that the server is holding for it.

1.3 requirements

The minimum requirements to run POPd:

- One (1) or more Amiga computers.

- AmiTCP 3.0b2

- Enough free RAM to hold the largest e-mail processed by POPd.

For maximum effectiveness, the computer running POPd needs some way to recieve e-mail for POP3 users. The mailbox files must be in UUCP format, so AmigaUUCP would work. AmigaSMTP 0.60 (included as the file SMTPd.amitcp in the INetUtils-1.3tcp.lha archive) by Michael B. Smith is recommended.

1.4 disclaimer

This software is provided as-is. No claim or warranty is made with regards to its ability to perform as stated. The user of this software bears sole responsibility for any effects of its use. The author will not be held responsible for any lost e-mail or damage to your system caused by any use of this software. Additionally, the author will not be held responsible for any sociopathic behavior by your Amiga resulting from its use, nor is he at all responsible for the bad coffee harvests in South America. You can't prove it.

This software is freely distributble as long as it remains in the original distribution archive. It may be included on the Aminet CD-ROM and in the Fred Fish collection. All other distribution for profit must be approved by the author. Even in Germany. It is NOT public domain, the copyright is retained by the author.

1.5 installation

Installation consists of three easy steps:

- Copy the POPd binary to AmiTCP:serv/

- Add this line to AmiTCP:db/services:

```
pop3      110/tcp     postoffice v3
```

- Add this line to AmiTCP:db/inetd.conf:

```
pop3      stream      tcp nowait root AmiTCP:serv/POPd in.popd
```

POPd is hard-coded to look for the mailbox files in UUMail:, so this assignment must be made to the place where they are stored. If you are running AmigaElm v3 or AmigaUUCP, this assignment should already be in place. Each user has an individual mail file named for their login name stored in this directory. These files should be in UUCP format, compatible with AmigaUUCP, AmigaSMTP 0.60, AmigaElm v3 and even AmiPOP 1.11.

POPd needs AmiTCP 3.0b2 for the usergroup.library, which provides a means of using a secure password system for POP3 requests. Each POP3 user needs to have a login name and password defined in AmiTCP:db/passwd. To limit users to POP3 access only, the 'nologin' keyword in the shell field of the user info will prevent access to the machine.

1.6 usage

The inetd program must be running in order to use POPd. Refer to AmiTCP documentation for information on inetd.

As POPd is a fully RFC1081-compliant POP3 daemon, it is used in a manner identical to that of any other POP3 server. Simply configure the POP3 client to use the IP address or name of the Amiga running POPd and enter the user name and password as defined in AmiTCP:db/passwd.

For information on using POP3 client software, refer to the client's own documentation.

1.7 implemenation

As I was fooling around with SAS/C 6.51, I decided to look at the examples included in the AmiTCP-api-30b2.lha archive of daemons initiated by inetd. The included example showed that creating such a daemon was incredibly simple. So I did. POP3 is a fairly simple protocol and the Amiga lacked an implementation so I got RFC1081 and went to work.

POPd is written in C++ using only ANSI C stdio functions. In fact, the only C++ part about the whole thing is the mailbox object. I did this to keep the binary size low and make POPd portable. Only the inetd init needs to be changed to port it to other machines.

The code that loads and manages the actual mailbox file is implemented as a C++ class object, so that the implementation of the loading routines is very transparent to the rest of the program. Currently, the object scans the UUCP mail file, maintaining a list of file positions for each message, loading individual messages on demand. (Thanks to Marx Marvelous on ISCA BBS for this

suggestion!) This keeps memory use low, but can lead to disk thrashing. I think this is the best solution, especially if a POP3 user crosses some one on Usenet and gets a core dump in the mail.

This low-memory approach also helps greatly, as POPd uses no static or global variables, it is pure and can run multiple instances concurrently.

1.8 Bugs and Limitations

In a perfect world, there'd be no bugs. Alas.

- Dropped connections hang POPd. (Biggest bug so far.)

- POPd is currently hard-coded to use UUMail: as the spool directory.

- The USER and PASS commands *must* have an argument.

- More error checking should be done internally.

- Mail must be less the 4 gigabytes. 32 bit machine limitation, not POPd.

- Mysterious, extraneous X appeared in one mailbox I tested. After poring over the binary with CPR, I think it must be a stdio bug. The X did not affect anything, it was just annoying.

1.9 history

- 1.0 (8/24/94) First public release.

1.10 author info

Send bug reports, praise, comments and suggestions to:

Jonathan Gapen (jagapen@students.wisc.edu)

Please include as much information as possible in bug reports, such as your machine and software configuration, plus the steps by which I can re-create the bug, if possible.

If you actually use this software, please send me some e-mail stating that you do and what you use it for. That's not too much to ask?