Periscope v1.4 Documentation

Contains VALUABLE information!

Periscope facilitates the transfer of data file from Unix machines to Macintosh computers without the use of a telnet ftp session and subsequent file type coercion with ResEdit on the Mac. The user may connect to a Unix machine, without login protocol, and transfer a file to the Mac with explicit Macintosh filetypes and signatures. Persicope consists of two parts: a double-clickable Mac application which acts a client, and a Unix server to process requests from clients

Requirements:
The server should be run as background process on any Unix machine a user wishes to download files from.
The client will run on any Macintosh capable of accessing an internet Unix network via MacTCP.

About the server:

The originally intended model for the server was as a single Unix system process to be run in the background, serving multiple clients. However, due to security and other concerns the model was changed. Under the curent model, each user logs onto the Unix machine in question and starts their own server process (you may want to nohup it or configure it to run on login). The server therefore has only the permissions of its user. To accomodate multiple processes per Unix machine a variable port numbering system was implemented. On startup, the server has a base port number and a range number of ports. The server creates a socket and then tries to bind it to a port, starting with the base port number, and proceeding upwards through contiguous port numbers until it exhausts its range or successfully binds to a port. If all ports are occupied, the server terminates. The default values are:

port = 32200
range = 10

so the server will start at port 32200 and search up to and including 32209 unless command line arguments specify otherwise. The arguments are:

-p<base port>
-r<range>

The server command name is pscope

ex. **pscope -p6000 -r100&**

The server simply waits for connections and handles requests as they come in. To ensure that a client connects only to its matching server process, and not someone else's,

a handshake protocol was implemented. The client's first transmission to a server is expected to contain the Unix account name of the user. The server gets this name from the Unix environment and compares it with the value sent by the client. If the two do not match, the client is informed that the server is not a match and the connection is terminated. The client then tries the next port in a manner similar to that of the server startup. Once the handshake succeeds, the server will handle requests from the client. Note that the server is actually capable of handling multiple connections, and it is possible for one user to connect from multiple Macs or for different users to share a single server if they provide the correct account name from the Mac (this could be a security problem if, say, root started a periscope server, but it's not a problem in the context for which this app was developed, the average user doesn't have superuser privelege).

The client:

The client is a Macintosh application. Currently, it has no true concept of a document, since its only purpose is downloading files. This may change in the future. The interface consists of a menu bar and various dialog boxes.

After double-clinking the application, a File Selection Dialog is posted. You may select a Preferences (or 'Servers') file to be loaded intially. Hitting **Cancel** will start the you with a blank slate. The 'Server' files have self explanatory icons.

The **FILE** menu

Open Servers... (cmd-O)

Allows you to load any preferences file, clearing any previously configured info. In this context, a 'server' is a particular Unix machine to which you want to connect.

Save Servers... (cmd-S)

Allows you to save a set of configured server information.

Quit (cmd-Q)

You guessed it!

The **EDIT** menu

Its just there to look nice. I haven't done much with it.

The **CONNECT** menu

This is where the action is. There are three sections in this menu, divided by separators.

section 1-

The first section, which can be empty, lists the servers the client knows about (you have configured them). A server with a check-mark next to it means you have a connection to it and can make requests. By selecting a server without a check-mark, you are asking for a connection to it. A dialog box will notify you of success or failure. You may select from connected servers to set the current server, whose name will appear in a special position in the second section of the menu. This way, you can tell which server a request is being sent to.

section 2-

Get File (cmd-G)

A dialog box will appear asking you for the unix file name (specified relative to the Unix directory from which the server is started or root), a Mac filetype, and a Mac signature. The tilde (~) abbreviation for /home directories does NOT work!

Useful file info:

Application	Type	Signature
Microsoft Word	TEXT	MSWD
Any Midi Thingy	Midi	PSCP
StuffitClassic	SITD	SIT!

You can get the necessary info for any file type by using ResEdit's Info function on a file of the type you are interested in.

Disconnect (cmd-D)

Will close the current connection. If any other connections are open, the current connection will be set to one of them.

The last item in this section is the name of the current connection, which is always grayed out.

section 3-

Add Server *(cmd-A)*

The client can deal with up to 16 servers in the menu at one time, and this is how you specify a new server for use by the client. A dialog box will ask for a number of items:

The complete, correct internet name of the Unix machine
The starting port number to search for the server process
The range of consecutive port numbers to try
A short name for access through the menu
The account name of the user on the Unix machine in question

It is important to be sure all these items are valid before hitting **OK**, since the app doesn't do validation. If it detects a ridiculous value for any field, it simply ignores the action, and doesn't do anything (I'm lazy, OK!). IF everything is swell, the short name you specified should appear in the menu. You can save the contents of the menu with **Save Servers...**

Remove Server (cmd-R)

You can specify any server name appearing in the menu, and it will be removed.

Marc S. Cooperman
April 15, 1992