A-Shell

LPD

Reference

This edition published February 2007

Publisher

MicroSabio
22704 Ventura Blvd., #476
Woodland Hills, CA 91364 USA
Web site: www.microsabio.com
Email: info@microsabio.com
voice: 818-710-8437
fax: 818-704-4882

Authors

Jack McGregor has been developing and documenting A-Shell since 1994, and his authorship continues through the date of this publication. Ty Griffin was responsible for editing, formatting, spell-chucking and publishing.

Documentation

The MicroSabio web site documents page contains current documentation on A-Shell and related software products from MicroSabio. The documents in the A-Shell collection are available in multiple formats: HTML, Windows Help (CHM) and PDF, to name the most common. If you do not find the document format you prefer, contact MicroSabio and inquire.

The primary A-Shell documentation set consists of the following:

- Set-Up Guide
- Command Reference
- Development Guide
- XCALL Reference

See "Help Menu Links" in the *A-Shell Setup Guide* for information on accessing A-Shell documentation from within A-Shell.

Support

MicroSabio provides extraordinarily good support for A-Shell and its other software products.

The first place to go with a question is the A-Shell bulletin board ("BBS"). You can search the BBS by keyword, quite possibly finding the answer(s) you need. If not, then post your question there; you have to create an account to post a question, but doing so is painless and free. All of the forums are moderated by MicroSabio, which means that if another user or developer does not answer your inquiry, MicroSabio staff will.

The main MicroSabio web site contains a variety of documentation, utility, and other support pages and downloads.

The A-Shell Advanced Support Network contains updates, utilities, technical notes, and various other A-Shell resources, mostly of a somewhat technical nature. It also requires that you create an account, but, like the BBS, doing so is painless and free.

If you are not getting satisfaction via these support avenues, send MicroSabio an email with your question and it will be answered promptly.

Legal Notice

Trademarks

Contents

Introduction

Overview

AshLPD is an A-Shell/Windows application that implements a network print server that offers some advantages over standard network print servers for applications running A-Shell. These include:

- Support for the A-Shell/Windows GDI printing directives (allowing A-Shell/UNIX applications to take full advantage of A-Shell/Windows printing features).
- A single AshLPD server can support any number of logical and physical printers without user invention.
- Eliminates the need to go "outside A-Shell" to administer printing—i.e., you don't have to set up any UNIX or Linux print services.
- Support for a backup server (if the PC running AshLPD goes down, it can automatically attempt printing to another PC).
- Configurable archiving of print files on the AshLPD server.
- GUI interface on the PC provides real-time status display, access to log of files printed, ability to display and/or reprint files (including option to change printers).
- AlphaBasic source code provided, allowing easy customization.

Terminology and Definitions

It is easy to get confused when talking about PC-based print servers because the machine at each end acts as both a client and a server, depending on the context. The document will try to reduce that confusion by consistent use of the following terms:

Application Server: A machine that the application runs on; this is typically a UNIX or Linux machine, or possibly a Windows telnet server. From the standpoint of the network printing protocol, this machine is actually a client.

Print Server: A program, or hardware device with such a program embedded, that handles print requests via a defined protocol, typically from remote machines over a network.

AshLPD Print Server: An instance of AshLPD, running under A-Shell/Windows on a PC. The same PC may also act as a telnet client to the application server. An AshLPD print server instance may support multiple logical and physical printers, and a single PC may run multiple AshLPD print servers (although the latter is probably redundant.)

LPR / LPD: A specific protocol for remote printing commonly used in UNIX and often supported by print servers. LPR is the client side (issues the print request), and LPD is the server side (receives and processes it). AshLPD may be considered an enhanced version of LPD, but is not currently compatible with LPR clients. (It requires a special ASHLPR client.)

Requirements

- Application server must be running A-Shell.
- You must have TCP connectivity between the application server and the AshLPD server.
- No special terminal emulator or emulation is required.
- A-Shell 4.9.945+ on the AshLPD side, and A-Shell 4.8.840+ on the application server side.

Package Contents

Since AshLPD is an A-Shell/Windows application, the installation package includes a minimal runtime A-Shell/Windows installation, plus the following.

File	Description
ASHLPR.SBX	Print filtering subroutine to be invoked in printer init files to implement the client side of the protocol (i.e. to forward print requests to the AshLPD server).
ASHLPR.INI	Sample configuration file for the application server (e.g. UNIX) side
ASHLPD.LIT	Print server application to run be run on the PC, typically from the system tray.
ASHLPD.INI	Sample configuration for the print server (e.g. PC) side
ASHLPD.CHM	This document

Installation

- Run the installation EXE (typically called ashlpd-a.a.aaa-bbb.exe, where a.a.aaa is the A-Shell/Windows version and bbb is the edit level of ASHLPD.LIT). By default, it installs beneath \Program Files\MicroSabio\AshLPD, although you can put it anywhere.
- Copy the ASHLPR.SBX to the BAS: directory on the application server (e.g. UNIX machine).
- Copy the ASHLPR.INI to the OPR: directory on the application server.
- Create one or more printer init files on the application server which invoke the ASHLPR client hook (COMMAND=SBX:ASHLPR)
- Create printer init files on the print server (e.g. PC) which match the names of the printer init files on the application server (e.g. UNIX machine.)

The following sections provide more detail for some of the steps or topics introduced in the bullets above.

ASHLPR.INI

This file, which resides in the OPR: directory of the application server (e.g. UNIX or Windows Telnet Server machine) tells the ASHLPR.SBX module where to send the print requests. A typical ASHLPR.INI will look something like this:

```
SERVER=192.168.0.5:31515    ; primary print server
SERVER=196.168.0.6:31515    ; backup print server (optional)
```

Another possibility is shown here:

```
SERVER=TELNET:31515         ; server is same as telnet client
```

In the above example, we used a special option, TELNET, to indicate that the print server is on the same machine as the telnet client. (Each telnet client would then need its own copy of ASHLPD running.)

Printer Init Files (Application Server)

A printer init file on the application (e.g. UNIX) server need only specify the ASHLPR subroutine, using the following syntax:

```
COMMAND=SBX:ASHLPR{,FLAG{,SERVER:PORT}}
```

The FLAG parameter, if specified, may be set to 1 to cause the print file to ALSO be printed on the specified printer on the application server (in addition to being printed via ASHLPD). In this case, the printer init file must also contain a DEVICE statement, e.g.:

```
COMMAND=SBX:ASHLPR,1
DEVICE=laser1
```

The above example would send the printout to the laser1 print queue on the application server, in addition to the ASHLPD server.

The SERVER:PORT parameter may be used to override the SERVER parameter in the OPR:ASHLPR.INI file. This is handy when certain individual logical printers have their own ASHLPD server (such as at a particular remote location), for example:

```
COMMAND=SBX:ASHLPR,0,192.168.200.100:31515
```

Printer Init Files (Print Server, i.e. Windows PC)

There must be a printer init file on the print server (e.g. Windows PC) side to match every printer init file on the application server that references `COMMAND=SBX:ASHLPR`. For example, if you create a spool queue named JINX on the application server that you want to connect to a PC printer via AshLPD, you would have an `ASHCFG:JINX.PQI` file on the application server side and also one on the PC side, e.g.:

Application Server (e.g. UNIX) – `ASHCFG: JINX.PQI:`

```
COMMAND=SBX:ASHLPR
```

Print Server (e.g. Windows PC) – `ASHCFG: JINX.PQI:`

```
DEVICE= Jinx-Jet Supreme | \\wserver\jinx
PASSTHROUGH=OFF
```

> The example above is a fairly minimal A-Shell/Windows printer init file. Consult the *A-Shell Setup Guide* for more details. The "Printer Defs" button within the AshLPD main window will help you get set up the printer init file(s) on the AshLPD server.

Operations

Client (application server) Usage

From the standpoint of the application server, using AshLPD (i.e. printing to an AshLPD-controlled Windows printer) is pretty much the same as printing to any printer. The process starts when the application makes a request to print a file (typically via SPOOL.SBR, EZSPL.SBR, or PRINT.LIT). If the printer name is not specified in the request, then A-Shell retrieves the default printer name from the MIAME.INI file. It then looks for a printer initialization file matching the printer name, either SYS:<printer>.INI or ASHCFG:<printer>.PQI. (The latter is the "approved" location, but the former is preserved for backwards compatibility.)

As A-Shell scans the printer initialization file, it will come upon the `COMMAND=SBX:ASHLPR` statement, which leads it to call the ASHLPR.SBX subroutine to process the request. ASHLPR.SBX in turn reads the OPR:ASHLPR.INI file to find the IP address and port of the AshLPD server (unless it was passed directly as an argument to ASHLPR), and then tries to connect to the server to transfer the file.

If the connection is accepted, then the file is transferred and the process is considered complete, at least from the standpoint of the application. (The print request returns to the application, which continues on its merry way.)

If the connection is not accepted, then ASHLPR retries a couple of times, at one second intervals (in case the AshLPD server is temporarily busy). If still no success and the AshLPD server IP was not specified on the ASHLPR command line, then it checks to see if there was a backup server defined in the ASHLPR.INI file. If so, then it tries to connect to it; again, up to 3 times spaced 1 second apart.

> Recently we have noticed that many Windows boxes can be very slow to react to a client attempting an invalid connection. This is probably a security measure to discourage automated programs from attempting to connect at every open port, but it has the side effect of making the above procedure of retrying quite slow. A subsequent version of ASHLPR will probably get around this by moving the entire operation to a background task.

If still no connection after all these tries, it pops up a message box to inform the user of the problem, and to present the choice to R)etry, Q)ueue for later, or C)ancel.

The Cancel option causes the request to be discarded and the process to return to the application.

The Queue for later option causes the file to be stored in the ASHLPR: working directory, along with an entry in a special ASHLPR.LOG file indicating the details of the request. The next print request that succeeds will automatically check the working directory and log to see if there are any queued requests, and if so, will transmit them at that time.

Server (PC side) Usage

AshLPD usage on the PC side is much more interesting than the application server side, and is discussed in the following several topics.

Launching AshLPD

The AshLPD server must be launched and left running on the PC server. Typically, you would run it in background, with only an icon showing in the system tray by launching it with a command similar to:

```
<install dir>\bin\ashw32.exe –i <install dir>\miame.ini –zi ashlpd
```

(The –zi switch causes it to launch in background from the system tray; otherwise it will launch as a normal window.)

> The installation program automatically adds a shortcut to the Startup folder with the above command so that it will start when the PC is booted.

When AshLPD first launches, it reads the configuration file `OPR:ASHLPD.INI` to get configuration options, the most important of which is the port to listen on. (This, and the IP address of the print server must match up with the configuration specified in `OPR:ASHLPR.INI` on the application server.)

> See the AshLPD User Interface section below for information on how to edit the configuration file.

AshLPD Licensing

AshLPD is an A-Shell/Windows application and does not, by itself, require a license, but to run it on a machine that doesn't already have A-Shell, you'll eventually need an A-Shell license, although it will also run in demo mode. When AshLPD is launched, if the underlying copy of A-Shell is not licensed, the main window will become visible, and a "**Register**" option will appear on the menu bar. Click the option to go to the licensing screen, which will display, among other things, the license status and the MAC address of the machine. The minimum license which supports AshLPD is equivalent to an ATE local PC license, which is tied to the local PC by its MAC address. Contact MicroSabio with the name and MAC address to get a real or evaluation license.

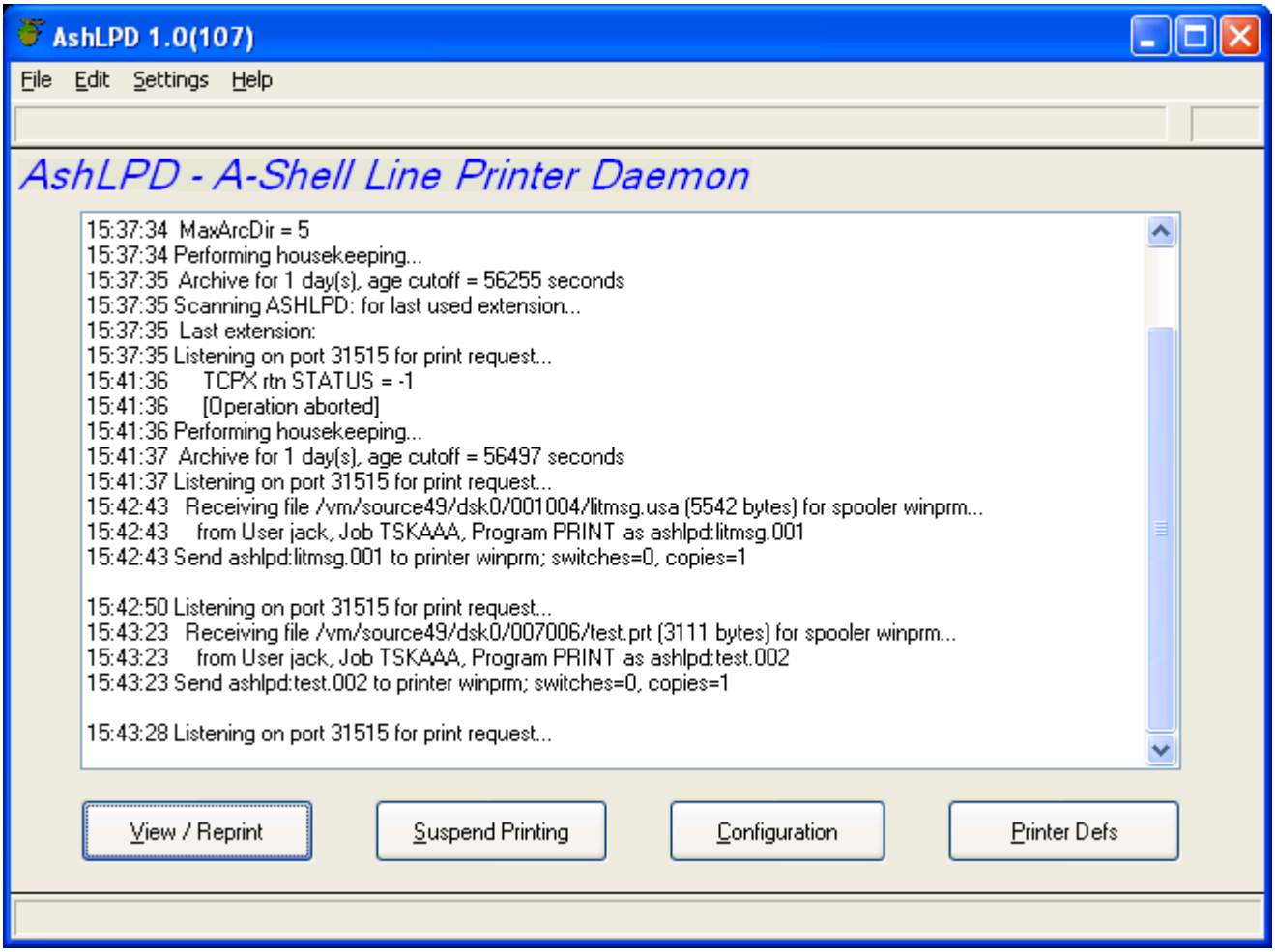
AshLPD Background Operation

Under normal operations, AshLPD waits for connections on the port specified in the `OPR:ASHLPD.INI` file. When a connection is accepted, it waits for a print request command from and receives the file to be printed. The print file is given a unique numeric (sequential) extension and copied to the ASHLPD: working directory, and an entry is made in the `OPR:ASHLPD.LOG` file. Finally, the file is then sent to the A-Shell/Windows print processor, which translates the file as needed and passes it to the printer.

Regardless of whether the print operation succeeds, the file remains in the ASHLPD: working directory until the archive process deletes it. (See the Archiving topic below.) This allows you to view and/or reprint the file, among other things.

AshLPD User Interface

To access any of the more advanced features of AshLPD, you need to bring up the user interface. If it is running from the system tray, you can do this by double-clicking on the acorn icon, which should bring up something like this:

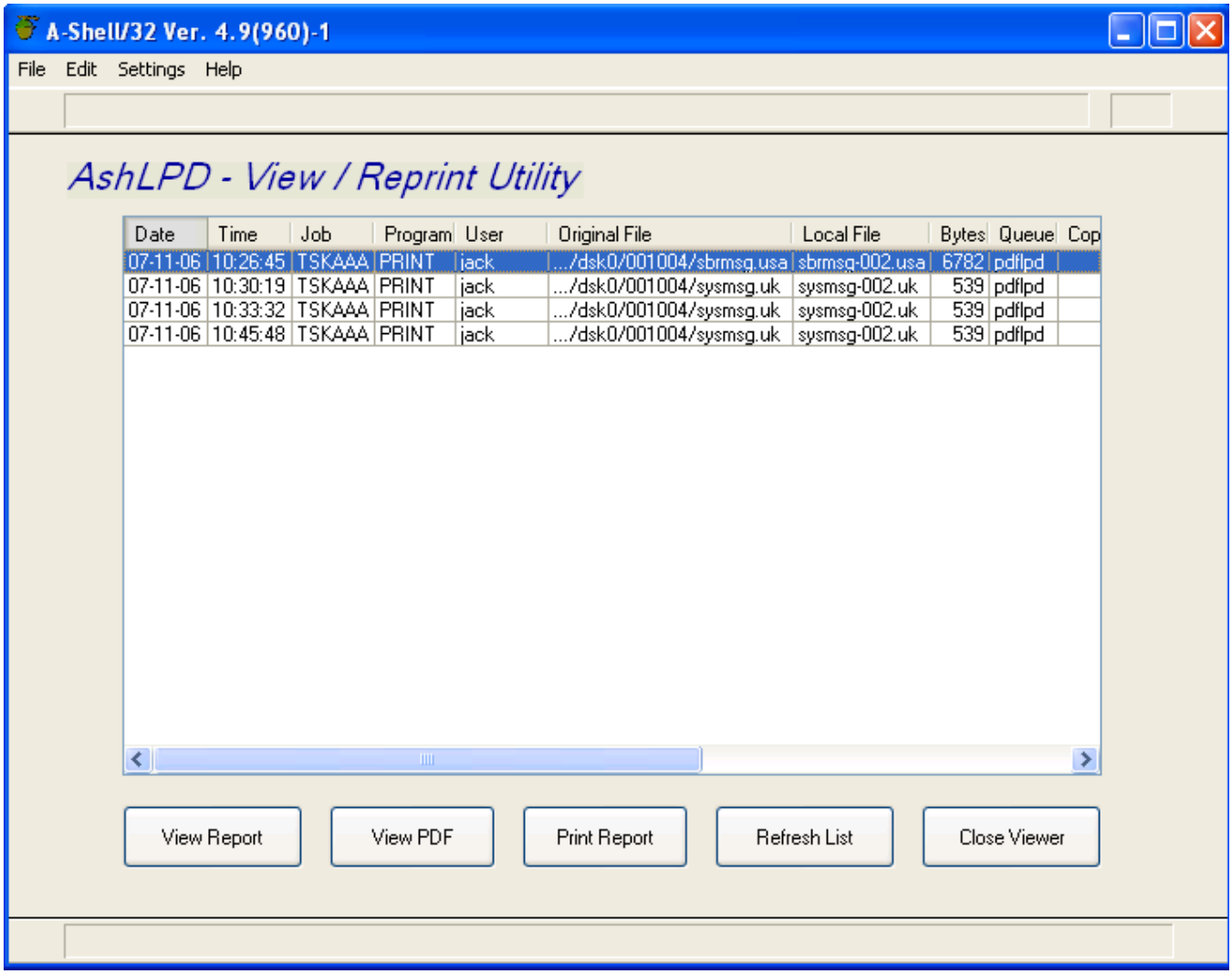


The main part of the display is a scrolling message log which records the last few dozen events, mainly so that you can see that it is working, when the last print request came through, etc.

The buttons along the bottom access the most common user operations, and are discussed in separate topics below.

View / Reprint

The View / Reprint button brings up another dialog (shown below) which lists all of the files currently held in the working directory, including details such as the time of the print request, job-program-user that make the request, original filename, local filename, size, print queue name, copies, and switches:



You can select any of these files to either view or reprint. (The print button gives you the option of choosing any printer visible to the local PC.)

Suspend Printing

The Suspend Printing button (on the main AshLPD dialog) acts as a toggle to enable/disable printing. Even when suspended, AshLPD continues to receive and store print requests as it normally does, so the application side is not aware of the suspension. Any files that were received during the suspension could be later reprinted by using the Print Report button on the View / Reprint dialog.

Configuration

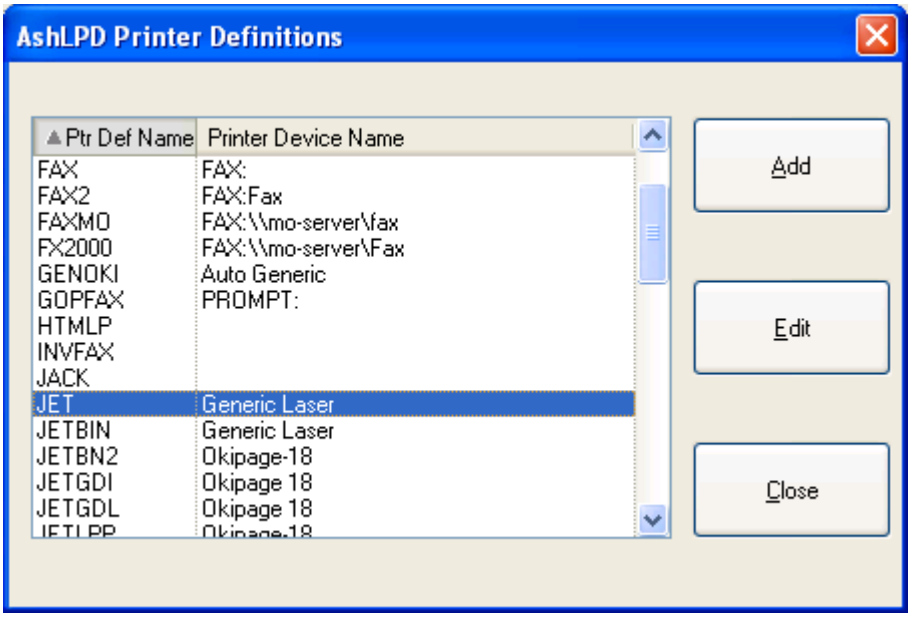
The Configuration button on the main AshLPD dialog opens up a text editor where you can modify the configuration settings for AshLPD. A typical configuration file looks something like this:

```
[ASHLPD]
PORT=31515
VERBOSE=0
;# of days to archive files in ASHLPD: before deleting
ARCHIVE=1
```

The most important of the above settings is the PORT number, which must match the port specified in the ASHLPR.INI configuration file on the application server. VERBOSE may be set to a value greater than 0 to increase the amount of information displayed in the scrolling event log. The ARCHIVE option determines how long files remain in the ASHLPD: working directory before they are deleted.

Printer Definition

The Printer Defs button on the main AshLPD dialog leads to another dialog where you can add, edit and delete printer definitions:



The purpose of a printer definition is mainly to match up an A-Shell named printer (as used in the application) with a PC printer device. A secondary purpose is to allow for the possibility of configuring options on the Windows printer (such as font size, bin selection, etc.).

Click the Add button to add a new printer definition, which will prompt you for the name of the printer definition and then allow you to choose a Windows printer from the list of installed printers. It then creates a minimum printer definition file and launches Notepad to allow you to further adjust it if so desired. (Refer to the A-Shell Setup Guide for a reference to all of the printer configuration options; in particular, pay attention to the PASSTHROUGH option.)

To delete a printer definition, move the selection bar to it and hit the DEL key.

Archiving

AshLPD currently supports only the simple option of keeping all the print files for a specified number of days. For many people this is enough (and is more than a typical LPD print server will do for you). If you want to implement a more sophisticated archival scheme (say, moving the files each week to a named subdirectory, from whence they can be written to CDROM), you can modify the source code to AshLPD, or you can set up an external process which gathers the files from the ASHLPD: work directory and processes them per your desires.

Problem Solving / Scenarios

This section discusses a variety of scenarios with an eye towards problem solving.

Printing over the Internet

The biggest problem here is likely to be the firewall at one end or the other. At the application server, ASHLPR will need to gain outbound access to whatever port the AshLPD server is listening on (default 31515). In most cases, this will not be a problem, as most firewalls are configured to stop incoming access but allow outgoing connections.

The AshLPD side is more likely to have a firewall (or just the Windows software firewall or some other "intrusion protection"). It is impossible to cover all the possibilities here, but the general requirement is simply that you open up inbound access to the port in question (e.g. 31515).

Another issue is the need to specify the IP address of the AshLPD server (i.e. the remote PC) to the application server. If the remote PC running AshLPD has a fixed IP, you can just enter it into the ASHLPR.INI file. But if it is dynamic, or if you have several remote PCs (such as home users) that want to use the same print queue name, then the solution is to add the TELNET:<port> option to the end of the COMMAND statement in the printer init file on the application server. For example, to create a printer called REMOTE that will forward print requests to the AshLPD server which is servicing port 31515 on the same remote workstation that is originating the telnet session, you would create the following `ASHCFG:REMOTE.PQI` file:

```
COMMAND = SBX:ASHLPR,0,TELNET:31515
```

Note that the ",0" is needed to hold the place of the print flag and to indicate that the file is not to printed on the application server (it is only to be forwarded to the AshLPD server). If you wanted to print the file ALSO on the application server, you would change the 0 to a 1 and add a DEVICE statement to indicate the name of the printer device.

Font Sizing

There are two commonly used methods for setting the font size in "legacy" AlphaBasic reports. One method involves embedding printer-specific ESCAPE codes (such as PCL commands) into the file itself. Assuming that you have you have a compatible printer at the AshLPD end, this method just requires that you add PASSTHROUGH=ON to the printer definition file on the AshLPD side.

The other common method is to not embed any font commands in the print file itself and instead to just rely on the user to have the printer loaded with appropriately sized paper (or to manually set switches on the printer for the desired font size). In this case, you can take advantage of A-Shell/Windows' GDI printing feature to auto-set the font size, by adding the following to the printer definition file on the AshLPD side:

```
PITCH=AUTO
CPP=*
```

The PITCH=AUTO option causes the print processor to scan the first 100 lines of the printout to determine the longest line length. Based on this maximum line length, and the fact that the CPP (columns per page) option is set to *, A-Shell/Windows will request a suitable font so that the longest line fits in the width of the page.

If you prefer that the print size not vary so much, and would rather have all reports use just one of two fonts (say, one for 80 column reports and one for 132 column reports), then you could use a variation of the CPP statement like this:

```
CPP=80,132
```

This would generate a font suitable for 80 columns, for any report whose longest line was less than or equal to 80 columns. Otherwise it would generate a font suitable for 132 columns. (You can adjust those to provide extra margin space, or set the second value to * to indicate a custom font for any line length beyond 80 columns.

Filtering / Special Processing

If you want to do some filtering or special processing while (or in lieu of) printing, you can do that on the PC (AshLPD server) side by specifying a COMMAND in the AshLPD printer definition file. As one example, let's say that the print files on the application server are hard coded for a PCL printer, but you want to be able to print them via AshLPD to a Windows-only ink jet printer. To do this effectively, you will need to strip out the PCL code. To do this, you could implement a printing SBX, let's say called FILTER, that you insert into the printer definition file on the PC side as follows:

```
;<AshLPD printer definition>
COMMAND = SBX:FILTER
DEVICE = Cheap Ink Jet | \\pserver\cheapjet
```

When the application sends one of these files to be printed, it would first be transferred to the AshLPD workstation, where it would be passed to the FILTER.SBX file which could strip out the PCL code (and even replace it with corresponding GDI commands), before sending it to the cheap ink jet printer.

As another example, let's say you want to email copies of all reports going to a certain printer. On the application server side, you would set up a printer init file something like this:

```
;<application server printer definition for emailing>
COMMAND = SBX:ASHLPR,1
DEVICE = prt1
```

The ,1 in the `COMMAND` statement above causes the print file to be sent to the print queue defined by the `DEVICE` statement, in addition to being sent to the AshLPD server. On the AshLPD side, the matching printer init file would have to use a different SBX to reformat the file for emailing. For interactive emailing, you could use the ready-made EMAILP.SBX, as follows:

```
;<AshLPD printer definition for interactive emailing>
COMMAND = SBX:EMAILP
```

Otherwise, for program-controlled email, you could write your own filter SBX, perhaps making use of the EMAILX.SBX routine, e.g.:

```
;<AshLPD printer definition for program-controlled emailing>
;(note: EMAILZ.SBX to be custom written, using EMAILX.SBX?)
COMMAND = SBX:EMAILZ
```

A third example might be where you just want to make a PC-based copy of all files sent to a particular printer (without printing at all). To do this, just use the pseudo device DISK:<path>, e.g.:

```
;<AshLPD printer definition to just store a copy>
PRINTER = DISK:c:\print-archives
```

Note that the above printer AshLPD printer definition, in conjunction with the standard ASHLPR printer init file on the application server side, essentially turns the PRINT command into a file transfer command. That is, the application server command:

```
PRINT <printer>=<file>
```

Would just copy <file> from the application server to `c:\print-archives\<file>` on the PC (and also log it to the OPR:ASHLPD.LOG file.)

Graphics

One straightforward advantage of using AshLPD as a printing service is that it allows you to use the A-Shell/Windows GDI printing directives under UNIX. No special tricks are needed. Just define a printer initialization file on the application server with the standard COMMAND = SBX:ASHLPR and create a printer definition on the AshLPD side that includes PASSTHROUGH=OFF. Any GDI printing directives in the file will be automatically processed during printing on the PC side.

Printing to an Unknown Windows Printer

This is a common situation, where you want to allow for the possibility of several different remote users to print selected reports back to a printer which they select only at the time of printing. The concept is similar to using emulated auxiliary port printing in an emulator such as ZTERM, with the printer choice set to "Prompt for printer".

To accomplish this, set up the application server side just as described for Printing over the Internet (above). On the AshLPD side, create a matching printer definition which uses the pseudo device PROMPT:, i.e.:

```
;<AshLPD printer definition for to prompt for printer>
DEVICE = PROMPT:
```

Known Limitations

- Server is only single-threaded, and is unable to respond to requests while performing other operations. (This is not true of all operations, since some, such as the View / Reprint dialog, launch an additional instance.)
- The ASHLPD.LOG file does not note whether a file was successfully printed or not.
- Print requests that are queued for later on the application server (because AshLPD wasn't available) are mixed together in the ASHLPR: holding directory, even if for different printers. Then, when a successful transfer is made, the queued up files will end up being transferred using the current instance of ASHLPR.SBX, which could result in delivery to the wrong AshLPD print server if there were multiple servers and the file in question was submitted to a printer which specified a different server address.
- The configuration file and printer definitions are currently edited via NOTEPAD rather than via custom dialogs.