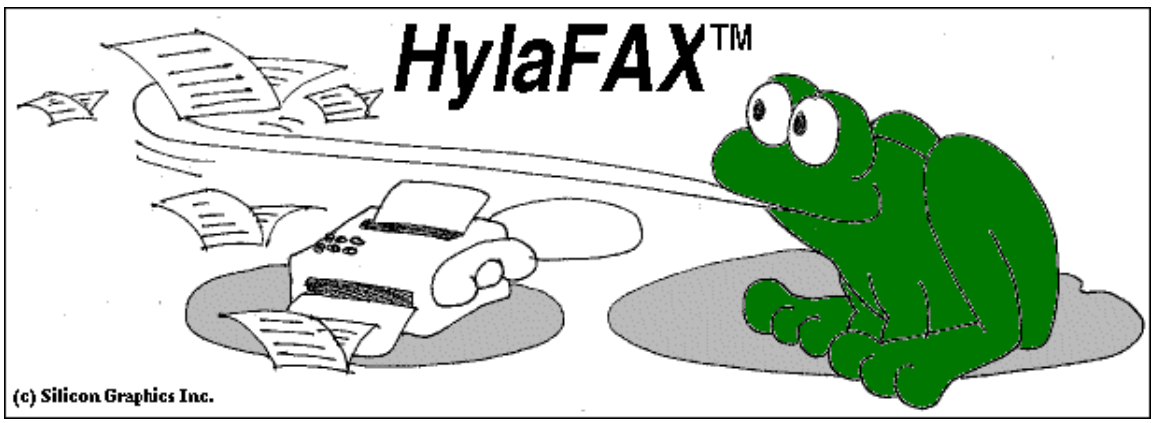

Latest *Distribution*: v4.0beta020
Latest *Release*: v3.0pl1 (Version 3.0 patch level 1)
Master FTP Site: ftp.sgi.com (192.48.153.1), directory sgi/fax
Home Page: http://www.vix.com/hylafax/
HylaFAQ Page: http://www.vix.com/hylafax/FAQ/
Mailing List: flexfax@sgi.com (archives are kept at ftp.celestial.com)

This is the home page for the HylaFAX™ telecommunication software.

- *If you want to use the software and are impatient*, try this synopsis.
- If you have a problem with the software, check out the troubleshooting chapter.
- If you have a question, consult the HylaFAQ or search the mailing list archives.
- If you are looking for a source or binary distribution, read these instructions.
- If you want an annotated list of all online documentation, check out the table of contents.

If this is your first encounter with this software you should start with the overview section and follow the button at the bottom of each page to get a top-to-bottom reading of the most important material. You will also find that each page has a button that takes you to an annotated table of contents for this documentation.

HylaFAX Table of Contents

This is the table of contents for the documentation included with the HylaFAX software distribution. The directory ftp://ftp.sgi.com/sgi/fax/doc. contains single file versions of this documentation in various formats including HTML and PostScript (suitable for printing). If you are looking for ancillary materials such as the FAQ or mailing list archives, consult the section on supporting materials

---

Basic information

- HylaFAX overview

An introduction to the software's features and some general information on its structure and the organization of the documentation.

- Where to get the HylaFAX distribution
  This is the definitive answer to where to get a distribution, be it source or binary.
- How to tell which version of HylaFAX you have
  Instructions on how to concisely identify the software version from a source or binary distribution; critical when reporting problems.
- Which modems can be used with HylaFAX
  General information on what types of fax modems can be used with the software and a table of specific modems that have been tried. This section also includes modems that are *recommended for use with this software*.
- About the Class 1 modem support
  Important information about limitations in the use of this software with Class 1 modems.
- About the Class 2 modem support
  Information about the support and how to deal with a common deficiency in these modems.
- About the Class 2.0 modem support
  Some quick comments about these new-style modems.
- About the source distribution
  Introduction to the source distribution and the requirements for building the software.
- About the binary distribution
  Information on where to locate binary distributions and how to interpret their packaging.
- Upgrading from HylaFAX Version 3.0
  Guidance for users upgrading from the last HylaFAX release to the current distribution; specifically what has changed in incompatible ways that you should be aware of.

BUILDING FROM SOURCE

- How to build the software from source code
  The critical chapter to read when working with the source distribution. The information in this chapter may also be useful to folks working with binary distributions.
  - Before You Start
    If you do not read this section before you run configure then you may do extra work and get pretty frustrated.
    - TIFF
    - gcc
    - C++ runtime libraries
    - ghostscript
    - gmake
    - gawk
    - sed
    - /bin/test
    - ps2fax binary for IRIX systems
  - The Build Procedure
    When the software builds without a hitch you can ignore this stuff; but if not be prepared to spend some time reading the fine print.
    - Build Trees

        - Configuration Files
        - Configuring Packages
        - Configuring the HTML Documentation
        - Upgrading From FlexFAX
        - A Sample Configuration Session
        - Configuration Parameters
    - Troubleshooting Build Problems
      Building this software is bulletproof; you'll never need this stuff...right!
    - System-specific Guidance
      Information about pitfalls for specific systems are provided; some of this stuff is just too hard to handle automatically in the normal build support.
- Building Ghostscript for use with HylaFAX
  Quick help on what to do to create a version of Ghostscript suitable for use as the PostScript Raster Image Processor (RIP) that HylaFAX uses to image outbound facsimile documents.

SETUP INFORMATION

- Basic server configuration
  Step by step instructions on setting up a server machine once the software is installed from a source or binary distribution. This material describes only the basic work required to get a system running.
    - Selecting a Facsimile Modem
      So you got this software but no modem; have no fear.
    - Checking your Modem
      A primer on what a fax modem looks like; in case you somehow did not follow the instructions on selecting a modem.
    - Selecting a Flow Control Scheme
      You may think you know all about modems and flow control but when it comes to facsimile you might be surprised!
    - Choosing a TTY Device
      This stuff should be straightforward but some vendors do their best to confuse the issue.
    - Using *faxsetup* to Configure a Server Machine
      The heart of the setup procedure; this is where HylaFAX tries to make sure you get things installed and setup correctly.
    - Using *faxaddmodem* to Configure Modems
      The tool you will want to use to initially setup and configure a modem for use.
    - Starting Outbound Service
      Most of this stuff is done when you run *faxsetup* but you will want to know what is happening for other times.
    - Setting up Inbound Service
      Arranging to handle inbound calls requires some manual configuration of system files because every system does this differently.
    - Settting up Client Access
      How to run the client-server protocol process that all the client applications talk to and how to configure user access.
    - Setting up Periodic Maintenance Work

If you do not arrange for some scripts and programs to be run periodically on your server system then you will be sorry.

- Modem Configuration Issues
  Some stuff you should be aware of when running HylaFAX; how it might affect your other system applications.
- System-specific Setup Advice
  System-specific tidbits about setting up a HylaFAX server machine.
  - IRIX-specific Guidance
  - SCO-specific Guidance
  - Solaris-specific Guidance
  - SunOS-specific Guidance
  - SVR4-specific Guidance
  - Ultrix-specific Guidance

- Advanced server configuration
  Information about optional facilities provided by HylaFAX. The material here is typically short and introductory with more complete information found in the reference documentation (manual pages). The last few sections give some background information about how the various modem-related configuration parameters fit together so that you can understand how the large set of tunable parameters can be most effectively used.
  - Local Identifier Support
    How to setup a string that identifies yourself to folks that receive your facsimile.
  - Dial String Rules
    How HylaFAX processes cient-specified phone numbers and the like.
  - Tagline Support
    Setting up that line of text across the top of each transmitted page of a facsimile.
  - Adaptive Answer Support
    Hints on configuring a modem for combined fax and data use.
  - Caller-ID Support
    How to enable the caller-ID support and use it to screen inbound calls.
  - Distinctive Ring Support
    How to distinguish the type of inbound calls based on distinctive ring patterns.
  - Copy Quality Checking
    Page quality checking of received facsimile documents.
  - Continuation Cover Pages
    Enabling the automatic generation of cover pages for facsimile that are retried after a communication error.
  - Time-of-Day Usage Scheduling
    How to constrain outbound calls based on the destination and the time of day.
  - Per-destination Controls
    The general per-destination control facilities provided by HylaFAX.
  - Page Chopping Support
    How to get HylaFAX to truncate facsimile pages that have lots of trailing white space.
  - Modem Classes
    How to setup groups of modems for outbound use.
  - Modem Priorities
    How to prioritize modems for outbound use.
  - Transcoding of Received Facsimile
    Automatically converting the data format of received facsimile for optimal storage.

- Automatic Processing of Received Facsimile
  How to automatically process (deliver and/or print) received facsimile.
- Rejecting Junk Facsimile
  How to screen inbound facsimile calls based on the sender's purported identity.
- UUCP Lock Files
  A quick overview of the UUCP locking scheme and how to tailor it for non-standard systems.
- Modems that Lie about their Capabilities
  Some tricks to use when modems are not quite what they claim to be.
- Configuration Parameter Usage During Modem Setup
  How the configuration parameters get used when a modem is initialized for use.
- Modem Parameter Usage for Outbound Calls
  How the configuration parameters get used during facsimile transmission.
- Modem Parameter Usage for Inbound Calls
  How the configuration parameters get used during an inbound call.

- Alpha-numeric pager support
  Setting up the optional facility for sending alpha-numeric messages to pager devices using the IXO/TAP communication protocol.
  - Setting up SNPP
    SNPP is the protocol used by client applications to submit requests to send pages.
  - Setting up Pagesend
    This is the backend program that does the actual delivery of the a page request by talking IXO/TAP over a phone connection to a paging service provider.
- How to setup a client.
  What to do to setup a system that is going to run only the client software.
- About server operation.
  Quick description of what goes on on a server machine and what administrative tasks must be managed for a server. The material in this section is mostly a reiteration of information in other sections from an administration perspective.
- How to setup a mail to fax gateway
  It is simple to arrange for electronic mail messages to automatically get submitted as facimile. HylaFAX includes fairly extensive support for controlling the content and delivery of facsimile submited in this way.

TROUBLESHOOTING

- Troubleshooting HylaFAX Problems
  If you have a problem this is where to start. This chapter discusses the facilities provided by HylaFAX for diagnosing problems and gives specific examples of how to use them. There are also comments about the most frequently encountered problems. The material in this chapter is supplemented by the HylaFAX Frequently Answered Questions document (HylaFAQ) which is updated more frequently than this documentation and often contains information relevant to problems with specific distributions.
  - Client Basics
    If you are running a client command line application such as *sendfax* or *faxstat* and do not know where to begin then this is the place.

- Client Access Control Problems
  What to do when a server will not let you in but you think should be allowed. Be sure that basic communication with the server works before trying the advise given here.
- Server Basics
  Basic information that you need to get started debugging problems that might occur on a server machine. You will find information here about the tracing logs that are the key to understanding server problems.
- Scheduler Operation
  When the central scheduler will not process your job or it is not doing what you expect read this section.
- Session Tracing
  An introduction to the session tracing logs maintained for all phone calls serviced by HylaFAX.
- PostScript Document Preparation
  What to try when you just can not figure out why the PostScript document you submitted for transmission keeps getting rejected.
- TIFF Document Preparation
  TIFF documents are handled just like PostScript documents; sort of. If something goes wrong sending a TIFF document and you do not understand then look here.
- Communication Problems
  The most common area of frustration; some guidance and suggestions on what to do when a job does not go through.
- Getty Problems
  If you encounter problems arranging for inbound data calls to be handled look here.
- Problems With UUCP, cu, tip, etc.
  When HylaFAX and the normal system programs do not play together you may find this information useful in figuring out what is wrong.
- Alpha-numeric pager support
  A bit of advise on common problems encountered when using the IXO/TAP support.

SUPPORTING MATERIALS

- HylaFAX Frequently Asked Questions (HylaFAQ)
  The FAQ is updated more frequently than this documentation and is also a good place to look for answers to problems specific to a particular distribution.
- HylaFAX mailing list archives
  This is a repository of the FlexFAX/HylaFAX mailing list that has fallen into disrepair in recent times (the repository, not the mailing list). You can also find more conventional archives of the mailing list at ftp://ftp.celestial.com/pub/mailing-lists.
- HylaFAX user survey form
  If you are running a HylaFAX server then filling out this form will help me track which systems people are using the software on and what modems are in use. Read the survey to more fully understand its purpose.

- FlexFAX Modem Bakeoff report
  This is a very old paper that is included just so folks will not ask me where it is.

- FlexFAX/HylaFAX modem information
  Limited information about specific modems; this was supposed to be a general repository for information but it never happened.
- ZyXEL modem voice support information
  Reference information, now old, on the ZyXEL voice command support.
- Supra/Rockwell Class 1 documentation
  An old document from Supra about their modem.
- Supra/Rockwell Class 2 documentation
  An old document from Supra about their modem.
- Telebit T3000/WorldBlazer FAX commands
  A reference card for fax commands supported in older Telebit modems.
- Hayes technical reference for modem users
  Yet another old document that has some information that might be interesting.
- Peter Chen's "What you need to know about modems"
  Some good introductory material for novices.
- FAX modem command scorecard
  An old project to collect information about what modems implement what Class 2 commands; necessary given the nonstandard nature of the Class 2 command set.

END MATTER

- About contributed software
  Some folks have contributed stuff to go with HylaFAX; check here for pointers.
- Acknowledgements
  Lots of folks have helped make this package useful; you will find some of them listed here together with attribution to the various packages that HylaFAX makes use of.
- Other documentation
  Some pointers to documentation that can be found elsewhere on the web.
- HylaFAX mailing list and other final words
  How to sign up to be on the mailing list.
- Use and copyright
  The rules for using this software. Hint: there are not many!
- TIFF information
  Information on the other major software package HylaFAX depends on that I give away.

HYLAFAX OVERVIEW

HylaFAX is a telecommunication system for UNIX systems. It supports:

- sending facsimile
- receiving facsimile
- polled retrieval of facsimile
- transparent shared data use of the modem

- sending alpha-numeric pages

Facsimile can be any size (e.g. A4, B4), either 98 or 196 lpi, and transmitted/received as either 1D-encoded or 2D-encoded facsimile data (2D-encoded data is frequently more compact and hence takes a shorter time to communicate). Any modem that supports one of the standard interfaces for facsimile operation can be used; i.e. any Class 1, Class 2, or Class 2.0 modem.

Outgoing documents can be any format; the sendfax program uses a rule-based definition file similar to the System V **/etc/magic** file to deduce document types and to decide how to convert each document to a form suitable for transmission (either PostScript or TIFF/F). Automatic cover page generation is supported and users can easily tailor cover pages to their environment. A simple text-based phonebook database is supported by the sendfax program. Information is also provided on how to trivially setup an email to fax gateway service.

Incoming facsimile are stored in a receiving area as TIFF/F (read ‘‘*TIFF Class F*’’) files and may be automatically delivered by mail and/or printed. A fax server status program, faxstat, can be used to monitor the send and receive queues, as well as the state of facsimile servers.

Fax modems are shared with outgoing data communication applications that honor the *UUCP locking protocol*. These applications typically include: *cu*, *tip*, *kermit*, *uucp*, *slip*, and *ppp*. The software can also be configured so that incoming data calls cause HylaFAX to invoke the standard system getty program.

The software is structured around a client-server architecture. Fax modems may reside on a single machine on a network and clients can submit outbound jobs from any machine that can communicate with the machine on which the modems reside. Client software is designed to be lightweight and easy to port; imaging can be offloaded to the server or done on the client. (Imaging is however, typically done on the server because it simplifies administration.) An access control mechanism is included to control which users on which machines may access a server. Clients and servers communicate using well-defined publicly specified protocols: for facsimile the HylaFAX Client-Server Protocol and for alpha-numeric pages the Simple Network Paging Protocol (SNPP) specified by RFC 1861.

Multiple modems on a single server machine are effectively scheduled for high throughput. Broadcast faxing is well-supported through optimal imaging of transmitted documents and the effective scheduling of modem resources. Support is provided for scheduling jobs during off-peak hours based on the destination phone numbers (e.g. long distance calls may be scheduled for off-peak phone rates). An access control mechanism can be used to restrict the class of phone numbers called so that, for example, calls to emergency services such as 911 can be rejected out of hand.

The server requires a PostScript to facsimile imaging utility for useful operation (otherwise, only pre-imaged facsimile may be transmitted.) A Display PostScript-based imager is provided for IRIX 4.x- and 5.x-based systems. For other systems, a Ghostscript-based version can be built from the GNU sources.

HylaFAX is freely available under copyright in complete source form. It may be used in commercial applications in part or in whole without charge.

Documentation

HylaFAX comes with extensive documentation in two forms: this HTML-based documentation that is designed for on-line use and general guidance, and a complete set of UNIX manual pages that contain reference information in a more terse but precise format. The HTML documentation contains *links* to the manual pages providing a complete hypertext connection between the two forms of documentation.

The HylaFAX documentation is intended to support users of binary distributions; it is complete enough that access to the source code is not needed.

Esoterica

The name of this software package is ''*HylaFAX*'', not ''*hylafax*'', ''*Hylafax*'', or anything else. Also, do not call this software by its old name ''*FlexFAX*'' because that name is a trademark of another fax product and the folks that own that trademark are possessive. Please also note that ''*HylaFAX*'' is a trademark of Silicon Graphics and it should be treated as such when used in documentation.

Regarding the name, it is derived from the word *hyla* which is defined as ''*Any of a genus of (Hyla) of frogs, especially the tree frog.*'' Hence the logo found on the home page for this software.

Finally, please recognize that this is free software that represents the work of many people. The section of ''Acknowledgements'' lists those people that have made significant contributions.

Where to get the HylaFAX Distribution

HylaFAX is typically obtained by public FTP on the Internet. It is also available on a number of public domain and freeware-style CD-ROMs. The master distribution site for HylaFAX is the host **ftp.sgi.com**. The following information is included here:

- Obtaining the Software by FTP on the Internet
- Obtaining the Software by Electronic Mail
- Obtaining the Software Within Silicon Graphics

All the HylaFAX documentation is online on the World Wide Web. A quick synopsis of how to build and install the software from the source distribution is provided in the section on ''Installation from a source distribution''. Binary distributions come with their own installation instructions that are tailored to their format and the needs of the target system. Otherwise the HylaFAX home page at http://www.vix.com/hylafax/ is the place to go for *all* the HylaFAX documentation.

There are two mailing lists for HylaFAX users; information on how to subscribe is found in the chapter ''HylaFAX Mailing Lists and Other Final Words''.

Using public FTP on the Internet

From ftp.sgi.com (master distribution site)

The *source code* is available for public FTP on

```
ftp.sgi.com                     sgi/fax/source/
    (192.48.153.1)
```

For example,

```
% ftp -n ftp.sgi.com
....
ftp> user ftp
... <type in password>
ftp> cd sgi/fax/source
ftp> binary
ftp> get hylafax-v3.0beta099-tar.gz
```

Software is available in *released form* (*none presently available*), and in *beta form*, (v3.0beta036). Beta software differs from released software mainly in the level of testing that has been done on non-Silicon Graphics platforms. Any files in the **source** directory of the form:

```
hylafax-<version>-patch-<XX>
```

are shell scripts that can be used to apply patches to the specified <*version*>.

If you are uncertain about what a patch script does just look at its contents; there is a comment at the top that says what version of the software the script is intended to patch. The HylaFAX home page also contains a link to release notes for the latest patch file.

The <*XX*> sequence numbers in the patch filenames indicate the order in which the patches must be applied; e.g. hylafax-v3.0-patch-01 must be applied *before* hylafax-v3.0-patch-02.

Binary distributions and alternate sites

Many binary distributions are available on the master FTP site:

```
ftp.sgi.com                     sgi/fax/binary
    (192.48.153.1)
```

These include:

- Silicon Graphics *inst'able images*
- SVR4 *pkg images* for x86 systems
- HPUX-9.01 for HP9000/7xx systems
- Ultrix 4.4 for DecStation-5000 systems
- SunOS 4.1.x for SPARC systems

- AIX 3.2.5 for IBM RS/6000 systems
- Solaris 2.4 for SPARC systems

[*This list may be out of date, check the FTP directory.*]

Alternate sites that either mirror the master FTP site or that have specific binary distributions are:

Some of these sites had binary distributions of FlexFAX or HylaFAX 3.0, they will presumably have HylaFAX 4.0 distributions soon.

FreeBSD binary distributions are available from:

ftp.freebsd.org (165.113.58.253) pub/FreeBSD/ports-current/comms/hylafax/

NetBSD binary distributions are available from:

ftp.netbsd.org (206.86.8.12) pub/NetBSD/packages/binaries/NetBSD-current/i386

If you are unable to locate the materials you want at one of the above sites, consult the archie resource location service to find copies close to your system.

Obtaining the Software by Electronic Mail

If you cannot use FTP at all, there is a service called ‘‘*ftpmail*’’ available through a variety of systems including **decwrl.dec.com**: you can send electronic mail to one of these machines and the host will retrieve the requested files using FTP and then send you the files via electronic mail. To find out more about the ftpmail service, send a message to ftpmail@decwrl.dec.com whose body consists of the single line ‘‘*help*’’ or check out this sample response to the help query.

Obtaining the Software Within Silicon Graphics

Internal to Silicon Graphics there are inst’able images on the host **dist.engr** in the directory **/sgi/hacks**. Thus you can do something like:

```
% inst -f dist.engr.sgi.com:/sgi/hacks/hylafax
```

to install the latest version of the software on your machine.

How to Tell Which HylaFAX Version You Have

If you have need to refer to this specific software, you should identify it as:

```
HylaFAX v<version><n>
```

where <*version*> is found in the file **VERSION** and <*n*> is the number recorded in the file **dist/hylfax.alpha** of the source distribution. This string is also prominently displayed when you run the configure script to setup the software for compilation and each time the faxq scheduler process is started (look in the file where system messages are recorded).

If you have a binary distribution for a Silicon Graphics machine, <*version*> and <*n*> are displayed in `"versions -n hylafax.sw"`; for example:

```
I = Installed, R = Removed

   Name                 Version     Description

I  hylafax              1006004075  HylaFAX Facsimile Software, Version 3.0beta
I  hylafax.sw           1006004075  HylaFAX Software
I  hylafax.sw.client    1006004075  HylaFAX Client Software
I  hylafax.sw.server    1006004075  HylaFAX Server Software
```

Here <*version*> = 3.0beta and <*n*> = 075 (the right three digits in the Version column), so this is HylaFAX v3.0beta075.

Note that HylaFAX is distributed in two versions: a *released version* with patch levels to fix small bugs or problems and a *beta version*. Released versions represent software that has been carefully tested on all supported platforms and is believed to compile, install, and function correctly (except where noted in the documentation). Released versions take a long time to create. The last few released versions have been at least 12 months apart. Beta versions are test versions of the software that are distributed more frequently than released versions. These versions include new facilities and/or support for new systems. They are typically tested on one or two systems, but not as thoroughly as released versions and not on all supported systems. The current released and beta versions can always be found on the HylaFAX home page.

Which Modems can be used with HylaFAX

HylaFAX is intended to be used with fax modems. Fax modems are not the same as data modems though most contemporary data modems also include support for fax communication. HylaFAX works with modems that support standard host-modem command interfaces. Specifically, HylaFAX should work with any:

- Class 1 fax modem: modem supports the TIA/EIA-578 "Class 1" specification (see the section ‘‘Class 1 Support’’ for **important information on restrictions on this support**).
- Class 2 fax modem: modem supports the TIA/EIA-592 draft SP-2388-A of August 30, 1991 (see the section ‘‘Class 2 Support’’ for caveats).
- Class 2.0 fax modem: modem supports the TIA/EIA-592 "Class 2.0" specification (see the section ‘‘Class 2.0 Support’’).

Modems that support multiple classes can be configured to use any single class but not multiple classes.

Wherever possible HylaFAX works around known modem problems or restricts modem usage in order to provide a functioning system.

The table below lists modems that have been used with HylaFAX. There are likely others modems that have been tried and/or are being used with HylaFAX; this list shows only those that have some form of support in the distribution or that have been publicly reported as *functional* (though beware that what is functional to one person may not be functional to you). Consult the Modems section of this documentation for known modem bugs and gotchas or follow the links in the table to find information for a particular modem.

Modems in the list below that are marked with a 🙂 are *recommended for use with this software*. This is not to say that modems not listed below or not marked as recommended will not work well; only that the noted modems have performed reliably in extended use with the software. Beware however, that modems only work as well the firmware that controls them; be sure to use up to date firmware in a modem.

If you are selecting a modem to use with this software you will do best if you use one of the recommended Class 2 or Class 2.0 modems; rather than a Class 1 modem.

Note that modems based on off-the-shelf parts (as opposed to custom DSP-based implementations) such as those produced by Rockwell are usually plug compatible when it comes to fax usage. HylaFAX includes configuration files for modems based on modem parts from Rockwell: RC96AC, RC144AC, RC144DP, RC32ACL, RC224A, and RC288DPI parts; Exar; and Cirrus Logic. Writing a configuration file for a modem that is not directly supported is straightforward.

Manufacturer	**Model**	**Class1**	**Class2**	**Class2.0**	**Notes**
AIWA	PV-BF288	Yes	Yes	No	
Archtec	SmartLink V.32Te	Yes	Yes	No	
	SmartLink 1419AV	Yes	Yes	No	
AT&T Paradyne	DataPort 14.4	Yes	Yes	No	🙂
	DataPort 14.4 Express	Yes	Yes	No	
BAUSCH	MOD 144	Yes	Yes	No	
Boca Research	M1440E	Yes	Yes	No	
	M1440E/RC32ACL	Yes	Yes	No	@
	MV.34I	Yes	Yes	No	
BEST Communication	BEST 28800 EC	Yes	Yes	No	
CPI	ViVa 14.4/FAX	No	Yes	No	
Digicom	Scout+	Yes	No	No	🙂
Dynalink	Dynalink 1414VE	No	Yes	No	
Elsa	Microlink 14.4TL	No	Yes	No	

E-Tech, Inc.	P1496MX 5.06-SWE	No	Yes	No	
	Bullet E-288 MX	No	Yes	No	
Everex	EverFax 24/96D, 24/96E	No	Yes	No	
GVC	MaxTech 28800	Yes	Yes	No	
Hayes	Optima 144, Optima 28800	Yes	No	No	
	Optima 2400+Fax96	No	Yes	No	
	Acura 144, Accura 288	No	Yes	No	
Intel	SatisFAXtion 400e	Yes	No	No	
JVC	F-1114V/R6	Yes	Yes	No	
Logicode	Quicktel Xeba 14.4	No	Yes	No	
Maestro	Vfast	No	Yes	No	
Microcom	DeskPorte FAST ES 28.8	Yes	Yes	No	
Motorola	Lifestyle 28.8	Yes	No	No	
	Power 28.8	Yes	No	No	
Multi-Tech	MT1432BA, MT224BA, MT2834BA	No	Yes	No	☺, *
	MT1432BG	No	Yes	No	☺
	MT1932ZDX, MT2834ZDX	No	Yes	No	☺
NetComm	M7F	No	Yes	No	
Nuvo	Voyager 96424PFX	Yes	No	No	
Olitec France	Olicom Poche Fax Modem 2400	No	Yes	No	
Power Bit	Power Bit	Yes	Yes	No	
Practical Peripherals	PM14400FXMT, PM14400FXSA	No	Yes	No	
	PM28800XFMT, PM288MT II	No	Yes	No	
Supra	SupraFAX v.32bis	Yes	Yes	No	
	SupraFAX v.32bis/RC32ACL	Yes	Yes	No	@
	Express 288	Yes	Yes	No	
Telebit	T3000, WorldBlazer	No	Yes	No	☺
	QBlazer	No	Yes	No	
Telejet	14400	Yes	Yes	No	
TKR	Terbo Line 19K2/005-Q	No	Yes	No	
Tricom	Tornado28/42	Yes	No	No	

Twincom	144/DF	Yes	Yes	No	
UDS	FasTalk Fax32	Yes	No	No	
USRobotics	Courier v.Everything	Yes	No	Yes	
	Courier HST	Yes	No	No	
	Sportster 28.8	Yes	No	Yes	
	Sportster 14.4	Yes	No	No	
Well/Xtrum	AT-2814SAM	Yes	Yes	No	
Yocom	1414E	No	Yes	No	
Zoom	VFX, VFP 14.4V, 14.4X	Yes	Yes	No	
Zero One Networking	ZyXEL U1496	Yes	Yes	Yes	☺
	ZyXEL Elite 2864, Elite 2864I	Yes	Yes	Yes	

Legend:

* There are apparently many variants of the MT1432BA, the following models are known to work: MT1432BA, MT1432BA/A, MT1432MK, MT1432PCS.

@ /RC32ACL refers to second-generation products made with the Rockwell RC32ACL part and different firmware.

☺ *Modem is recommended for use with this software.*

Class 1 Modem Support (caveat emptor)

HylaFAX includes a driver for modems that support the EIA-578 "Class 1" programming interface. These modems export a very low level interface that requires that the CCITT/ITU Recommendation T.30 facsimile communication protocol be implemented almost entirely in the host. Robust support for this protocol places two requirements on the host system:

- low latency for serial line input
- near realtime response

In a UNIX environment both these requirements can be problematic. In particular, many UNIX systems increase the latency for data received on a serial port in order to reduce system overhead. That is, input data are typically held in a low level device driver for some period of time before they are passed to the user so that bursts of input data can be delivered to the user together. This behaviour is known to occur under the Silicon Graphics IRIX operating system; it may also occur under other systems. It is important for the proper operation of the Class 1 driver that input data be delivered to the facsimile server as quickly as possible. This may require making a non-standard call of some sort to the operating system. For IRIX systems this call is automatically done.

The response time requirements are important for insuring that T.30 protocol messages are received in a timely fashion. On a loaded system the protocol process may be preempted by other activities and not be scheduled fast enough to satisfy the protocol timing requirements. This can usually be guarded against by assigning the facsimile process a high scheduling priority. Unfortunately most UNIX systems do not support such facilities and so even if it is possible to receive serial line input with the minimum of delay, protocol timing requirements may not be met because of delays in scheduling the execution of the fax server process. For this reason the facsimile server processes attempt to raise their scheduling priority while actively sending or receiving facsimile. At other times, such as when doing queue management operations, processes run at a normal priority. On Silicon Graphics systems the ''high priority'' is a nondegrading scheduling priority that is above the priorities of the normal system processes. On SVR4-derived systems the ''high priority'' is a real-time scheduling priority. On other systems the server currently always runs at the same (normal) scheduling priority. For more details consult the `setProcessPriority` routine in **faxd/ModemServer.c++**.

In summary, If you want to use a Class 1 modem with this software and your system does not provide support for low latency serial line input you are likely to have troubles. If your system does not provide a mechanism for raising process scheduling priority (note that this is not the same as the UNIX ''nice'' parameter), then you may see problems when the server is under load. Exactly how much load will cause trouble is dependent on the system configuration and processing power.

Class 2 Modem Support

The system includes a driver for modems that conform to TIA/EIA-592 draft SP-2388-A of August 30, 1991. Unfortunately, because this was only a draft, there are many modems that claim to conform to this document, but which diverge in sometimes subtle ways.

The Class 2 driver has been extensively tested with a wide variety of Class 2 modems. The software attempts to work around many incompatibilities and mistakes in modem firmware. A common problem that requires special attention is that some modems are incapable of accepting an `AT+FDIS` command to set the session parameters between the time a call is placed and a page is transmitted. Modems with this sort of problem may ignore the `AT+FDIS` command which causes incorrect session parameters being used to interpret the page data (often resulting in ''squished pages''), or abort the session, returning a `+FHNG:` status message to the host. If you encounter this problem use the `Class2DDISCmd` configuration parameter to enable workaround support in the fax server; e.g.

```
Class2DDISCmd:  AT+FDIS
```

Class 2.0 Modem Support

There is a driver for Class 2.0 modems that was developed initially using a USR Courier modem and later using a ZyXEL 1496E modem. Since this standard is very young do not be surprised if there are firmware problems and/or incompatibilities between the driver and modems with Class 2.0 firmware.

HylaFAX Source Distributions

The source distribution for HylaFAX contains *all the code* in the system except a small bit of code used to build the Display PostScript-based imager for Silicon Graphics machines (this code is useless unless you have a **developers agreement** with Adobe for Display PostScript). To build HylaFAX from sources you must have the TIFF software distribution installed on the system where HylaFAX is to be built.

The system is written almost entirely in contemporary C++. It uses nested types and in several places assumes the compiler generates code according to the ANSI C++ Reference Manual (ARM). The normal development environment for the software is based on compiler products from Silicon Graphics. Every effort is made to insure the code builds correctly under all contemporary C++ compilers, including the GNU gcc compiler distribution. Beware however that not all compilers will compile this software.

Versions of gcc prior to 2.6.1 will not correctly build this software; gcc 2.6.3 and libg++ 2.6.2 are the recommended versions of the GNU tools to use though later versions work also.

To build this software your system must have the following functionality or be capable of emulating those items that are missing:

- ANSI C compiler and runtime support (e.g.strftime).
- C++ compiler and runtime support. The C++ compiler must have support for nested types and ARM-style handling of temporary variables.
- ANSI C system include files with proper function prototype declarations (though some missing function declarations will be automatically handled).
- A make program that supports include files.
- Support for BSD-style sockets and TCP/IP for network communication, including the select interface for synchronous multiplexing of I/O streams.
- Support for FIFO special files.
- Support for BSD-style file locking (flock) or equivalent functionality from which it can be emulated (fcntl, lockf).
- POSIX 1003.1-style interfaces, including termios for manipulating tty devices.

When the software is configured for building any function that is not located will be emulated if it is known how. If a required facility is not found and no emulation or workaround is known, then the configuration procedure will abort.

The Class 1 modem driver requires sub-second timer facilities and a minimum latency interface to serial input. The server uses the BSD-style setitimer calls for the timer support and system-specific calls to enable low latency input from serial ports. If your system does not support the BSD timer calls, then the server will fall back to using the alarm system call that only has 1 second granularity (and the driver is unlikely to work reliably). If your system buffers serial input and does not provide a mechanism for defeating this, then the Class 1 driver will not work reliably. See the section above for specific requirements of the Class 1 support.

There is support in the fax server for handing inbound data calls to the system getty program. This support must be configured according to the requirements of the target platform; either System V-style or BSD-style. Only one of these two styles may be configured for use; the one that is appropriate to the target system should automatically be selected when the software is configured for building.

HylaFAX Binary Distributions

There are a number of pre-built binary distributions of the HylaFAX source code that are blessed as *safe for installation*. These distributions are available by public FTP from various sites on the Internet.

Beware of taking binary distributions from sites that are not on this list. HylaFAX includes several programs that are either started by the super-user or are installed as setuid root; thus installing untrusted distributions can be risky.

Many binary distributions are available on the master FTP site:

ftp.sgi.com (192.48.153.1) sgi/fax/binary

These currently include:

- Silicon Graphics *inst'able images*
- SVR4 *pkg images* for x86 systems
- HPUX-9.01 for HP9000/7xx systems
- Ultrix 4.4 for DecStation-5000 systems
- SunOS 4.1.4 for Sparc
- Solaris 2.4 for Sparc

[*This list may be out of date, check the FTP directory.*]

Alternate sites that either mirror the master FTP site or that have specific binary distributions are:

These sites had binary distributions of FlexFAX or HylaFAX 3.0, they will presumably have HylaFAX 4.0 distributions soon.

FreeBSD binary distributions are available from:

ftp.freebsd.org (165.113.58.253) pub/FreeBSD/ports-current/comms/hylafax/

NetBSD binary distributions are available from:

ftp.netbsd.org (206.86.8.12) pub/NetBSD/packages/binaries/NetBSD-current/i386

The binary distributions come with their own instructions. The following guidelines describe how to interpret the filenames to understand the format and contents of each distribution.

Distribution filenames are of the form: `hylafax-`<*target*>`-`<*version*>`-`<*format*>; where <*target*> describes the target system, <*version*> specifies the HylaFAX version the distribution was built from, and <*format*> specifies the format of the distribution as follows:

- a `.Z` suffix indicates the file has been compressed with the compress(1) program
- a `.gz` suffix indicates the file has been compressed with the GNU gzip(1) program
- a `.tar` component indicates the file was created with the tar(1) program
- a `-inst` component indicates the file contains images for the IRIX inst(1) program
- a `-pkg` component indicates the file contains images for the SVR4 pkgadd(1M) program

So, for example, `hylafax-irix5-v3.0beta099-inst.tar` is a binary distribution for machines running IRIX 5.x and the distribution is provided as IRIX inst images collected together in a single archive created by the tar program.

In addition to the binary distributions, there may also be two other files present for each target system. Files of the form `INSTALL-*-*` are installation instructions for a specific binary distribution. Files of the form `gs*-*-tar.gz` are binary distributions of Ghostscript, the Ghostscript fonts, and any associated .ps files; configured for use with HylaFAX.

Upgrading From HylaFAX Version 3.0

If you were previously using HylaFAX Version 3.0 beware that a number of things have changed in incompatible ways. The following is a list of potential things you should be aware of when setting up HylaFAX 4.0 on a server:

- You must remove any entries in the **inetd.conf** file for the old faxd.recv program and stop any running faxd.recv server processes.
- The new client-server protocol operates at port 4559 (+2 from the old ‘‘fax’’ port which defaulted to 4557). You may want to add a new service definition to a YP/NIS database (though you should be able to function without it.
- The new client-server protocol server, hfaxd(1M), is built by default with support for the old client-server protocol but the support must explicitly be enabled. If hfaxd is started up through the default shell script run by *init*(1M) and *not* by *inetd*(1M) then this happens automatically. Otherwise enabling support for the old protocol may require some manual configuration work. Consult the hfaxd(1M) manual page.
- faxq no longer accepts a `-m` option to specify modems to use in a send-only configuration. The faxmodem program should be used instead; consult the chapter on ‘‘Server Setup and Basic Configuration’’ and the manual pages for faxmodem(1M) and faxq(1M).
- The **etc/hosts** file on the server may contain passwords in v4.0 and hfaxd will not trust any information there unless the file is mode 0600. Since previous versions did not care what mode the

file was it probably is publicly readable and hfaxd will deny client access until the mode is corrected.
- Completed jobs are removed asynchronously with a new faxqclean program; you will want to setup a cron entry to do this.
- The new faxsetup(1M) script must be run before a server machine can be used.

The remaining items are relevant to client and server installations:

- The old syntax ‘‘host:modem’’ for specifying a server and modem is no longer supported; the only syntax supported is ‘‘modem[@host[:port]]’’.
- The order of options on the command line to sendfax and sendpage is now important; consult the manual pages for details.

BUILDING HYLAFAX FROM SOURCE CODE

This chapter contains the information needed to configure and build HylaFAX from the source distribution. The following sections are available here:

- Before You Start
- The Build Procedure
- Build Trees
- Configuration Files
- Configuring Packages
- Configuring the HTML Documentation
- Upgrading From FlexFAX
- A Sample Configuration Session
- Configuration Parameters
- Troubleshooting Build Problems

There are some build-related issues that are dependent on the operating system present on the machine. Some system-specific guidance is sprinkled throughout these materials; the remainder is in a section on system-specific guidance that is found at the end of this chapter.

Before You Start

Before you build the software you **may need to obtain certain other software distributions**. All the software packages described here are available by public FTP from a variety of Internet hosts; consult the archie resource location service to find copies close to your system.

TIFF

HylaFAX requires version 3.4 or later of the freely available TIFF software distribution. The master

FTP site for this distribution is ftp://ftp.sgi.com/graphics/tiff. Note that the TIFF distribution must be installed before building HylaFAX from source code or HylaFAX must be specially configured to search for TIFF include files and libraries in non-standard locations.

gcc

You need a contemporary C++ compiler to build this system. *gcc version 2.6.3 is the current recommended version to use; though later versions are also known to work.* Versions of gcc prior to 2.6.1 will not work. When installing gcc beware of the make install step. On some systems and/or with some versions of gcc, the fax software may not compile properly if the include files are wrong (function prototypes that would normally cause type casts to be done may be missing).

C++ runtime libraries

HylaFAX requires only minimal C++ runtime support; basic dynamic memory allocation facilities and support for creating global static objects; it does not use any of the newer C++ runtime facilities such as exception handling. Most C++ compiler packages include everything that you need in the way of a runtime library. When using GNU gcc you may also need the libg++ distribution which is packaged separately from gcc. On some systems where gcc is used these facilities can be satisfied without linking against `-lg++` or `-liberty`, but on others one or both of these libraries are needed for certain library routines that are normally found in the C runtime library. The configuration procedure will abort if libg++ is needed but is not present on the build system. If libg++ is required you should use the version that is appropriate for the compiler. *When using gcc version 2.6.3, libg++2.6.2 is the recommended version to use.*

ghostscript

If you are not on a Silicon Graphics machine (and maybe even if you are), then you will need the Ghostscript PostScript interpreter software to build a PostScript imaging engine to be used by the facsimile server. (This imaging engine is frequently referred to as a Raster Image Processor or RIP.) Ghostscript comes in three flavors, two of which are important to us: GNU PostScript, a version governed by the GNU Public License (GPL), and Aladdin Ghostscript, a version covered by the Aladdin Free Public License. Both versions are copyrighted work and are not shareware or in the public domain. Aladdin Ghostscript is free for use by end users but does not allow commercial distribution; to do this you must arrange for a commercial license. Versions of GNU Ghostscript are distributed approximately a year following the corresponding Aladdin Ghostscript version. For more information on these and other issues consult the information located at http://www.cs.wisc.edu/~ghost/ghostscript/

Ghostscript Version 2.6.1 and later are known to work. *If you use version 2.6.1 however, be certain to apply patches 1-4.* If you do not apply the patches you will encounter a bug in the clipping code that is triggered by the default cover page distributed with this software. Also beware that if you use a version of Ghostscript prior to 3.12 (inclusive) that there is a tiffg32d driver that writes incorrect 2D-encoded facsimile data; either do not configure this driver for use or disable its use by setting **Use2D** in the HylaFAX scheduler configuration file so it does not try to generate 2D-encoded facsimile data for outbound jobs.

gmake

The make files are extensive and work untouched with the system make under many systems. If your make does not understand them, then you should be able to use the GNU make (gmake) instead. If you decide to use gmake, be sure to get version 3.63 or newer; otherwise you may encounter problems (especially with the rules that automatically generate source code dependency information).

gawk

Several of the shell scripts included in this software make use of awk. These uses are reasonably simple, but will not work if your awk is old enough that it does not support functions or the `-v` command line option for setting variable values before the `BEGIN` action is executed. If you encounter problems using the standard awk on your system, try the GNU awk: gawk.

When using gawk be sure to use version 2.15 or later; HylaFAX is unable to detect versions of gawk that do not work correctly.

sed

Many of the shell scripts included in this software make use of sed. Certain systems are known to have versions of sed that do not handle the shell scripts. If you encounter problems, the latest GNU sed should be substituted.

/bin/test

The software configuration shell script (configure) and the modem configuration shell script (faxaddmodem) make heavy use of the test program. On some systems the **/bin/test** program does not support options such as `-c` (test if a file is a character special device). Source for a contemporary, public domain, test program is available by public FTP from ftp.uu.net.

ps2fax binary for IRIX systems

The Display PostScript-based imager for Silicon Graphics systems is included in the binary distribution images provided on ftp.sgi.com. If you choose to work from the source code on an IRIX system you may want a copy of the ps2fax program: it is available separately on ftp.sgi.com in the HylaFAX source distribution area: ftp://ftp.sgi.com/sgi/fax/source. *Note that this is a COFF executable for IRIX 4.x and 5.x systems and requires that the IRIX **dps_eoe** package be installed for proper use.*

The Build Procedure

To build the software you need to first run the configure shell script that is located in the top level of the source directory. This script probes the target system for necessary tools and functions and constructs a build environment in which the software may be compiled. Once configuration is done, you simply run `make` to build the software and then `make install` to do the installation; for example:

```
hyla% cd hylafax-v3.0beta099
hyla% ./configure
    ...lots of messages...
hyla% make
```

```
    ...lots of messages...
hyla% su                # NB: installation must be done by the super-user
hyla# make install
```

In general, the software is designed such that the following should be ‘‘*make-able*’’ in each directory:

Target	Purpose
`all`	build everything that is to be installed (default)
`depend`	build dependency information
`install`	build and install everything normally installed
`clean`	remove .o files and cruft, but not executables
`clobber`	remove everything that can be recreated
`distclean`	remove *absolutely everything* that can be recreated

Note that after running "`make clobber`" or "`make distclean`" the configure script must be run again to create the Makefiles and other build-related files.

Build Trees

There are two schemes for configuring and building the software. If you intend to build the software for only one target system, you can configure the software so that it is built in the same directories as the source code.

```
hyla% cd hylafax-v4.0beta011
hyla% ls
COPYRIGHT       config          dist            faxrm           sendfax
Makefile.in     config.guess    distrules       faxstat         sendpage
README          config.h.in     etc             hfaxd           sgi2fax
README.ultrix   config.site     faxalter        html            util
TODO            config.sub      faxcover        man
VERSION         configure       faxd            port
afm             defs.in         faxmail         rules.in
hyla% ./configure
```

Otherwise, you can configure a build tree that is parallel to the source tree hierarchy but which contains only configured files and files created during the build procedure.

```
hyla% cd hylafax-v4.0beta011
hyla% mkdir obj obj/mycpu
hyla% cd obj/mycpu
hyla% ../../configure
```

This second scheme is useful for:

- building multiple targets from a single source tree
- building from a read-only source tree (e.g. if you receive the distribution on CD-ROM)

Beware that if you choose to use the second scheme for configuring the software you must not use an absolute pathname when you run configure (i.e. a pathname that begins with ‘‘/’’) and the make that you use to build the software must correctly support the VPATH facility.

HP-UX 9.05: *The standard make incorrectly processes VPATH; either use gmake or configure builds in the source tree.*

Solaris 2.3: *The standard make does VPATH processing incorrectly for files passed to the make dependency generator script (the last file in the list is not converted to a pathname relative to the source directory); this causes lots of messages that can be ignored.*

Configuration Files

The configuration process is critical to the proper compilation, installation, and operation of the software. The configure script runs a series of tests to decide whether or not the target system supports required functionality and, if it does not, whether it can emulate or workaround the missing functions. This procedure is fairly complicated and, due to the nonstandard nature of most UNIX systems, prone to error. The first time that you configure the software for use you should check the output from the configure script and look for anything that does not make sense for your system. A sample configure run is shown below together with an explanation of the work that is done.

A second function of the configure script is to set configuration parameters for the software. Many of these values are just default settings that can be changed after configuration through files that programs read at runtime, but some settings cannot be changed without rerunning the configure script. For example, by default the software is configured for installation in the **/usr/local** hierarchy; this cannot be changed without rerunning the configure script.

The configure script reads two *configuration files* to get configuration parameters: a *site-wide configuration file* that has settings that are to be applied to all systems at a particular site, and a *target-specific configuration file* that has settings for a specific system. Site-wide configuration files are named **config.site** and are automatically searched for first in any directory specified on the command line to configure with the `-site` option, or if that fails, in the directory in in which the configure script is located. Target-specific configuration files are named **config.local** and are looked for first in the top-level build directory, or, if that fails, in the directory in which the configure script is located.

*configure reads any **config.site** file first, and then any **config.local** file. This permits target-specific definitions to override site-wide definitions.*

Configuration files are actually just shell scripts (written in the Bourne shell syntax) that define well-known shell variables that are used in the configuration process. For example, the following file might be used on a BSD/OS system to configure the software for installation in the **/usr/contrib** area and to make use of the Adobe Font Metric files that are already distributed as part of the BSD/OS 1.1 distribution:

```
#
# Parameters suitable for BSD/OS 1.1
#
DIR_AFM="/usr/contrib/lib/flexfax/afm"  # reuse existing files
DIR_BIN="/usr/contrib/bin"              # directory for client apps
DIR_SBIN="/usr/contrib/bin"             # directory for system apps
DIR_LIBEXEC="/usr/contrib/bin"          # directory for libraries&hidden apps
DIR_LIBDATA="/usr/contrib/lib/hylafax"  # directory for client data
DIR_MAN="/usr/contrib/man"              # directory for manual pages
DIR_SPOOL="/var/spool/hylafax"          # directory for spooling area
DIR_HTML="/usr/contrib/html/hylafax"    # directory for HTML documentation
DIR_CGI="/usr/contrib/html/cgi-bin"     # directory for CGI scripts
```

The complete list of configuration parameters is given in the sample **config.site** file provided in the source distribution. Also, the sections below on ‘‘Configuring Packages’’ and ‘‘Configuration Parameters’’ have information on the most important or obscure parameters.

Finally, take note that there are two other configuration-related files that may be useful. The file named **config.cache** has all the configuration parameter settings that were selected the last time configure was run. This file is automatically read in by configure when it starts up to speed the configuration process. **config.cache** may be safely removed at any time. Cached settings are also automatically ignored if new settings are set through command line arguments to **configure** or through a **config.local** or **config.site** file.

The other file of interest is **config.log**. This file contains information and messages related to the various tests that configure does when constructing a build environment. If the configuration process fails for some reason or configure does something unexpected this file should be used to debug the problem.

Beware that the configure script is designed to be similar to scripts generated by the GNU autoconf facility but it is not 100% compatible. Use the `-help` option to configure for a list of the available options.

Configuring Packages

HylaFAX uses the notion of a *package* to control which pieces of software are included in a build. Some packages are optional and are installed only as needed, or only if specifically requested at the time the configure script is run. Other packages may select one of several incompatible choices that must be made when the software is built. Packages can be specified in a **config.site** or **config.local** file, or by using a `-with-`*`package`* option when invoking configure; e.g. `configure -with-AFM` to request the AFM package. Most packages are automatically configured according to the characteristics of the system where HylaFAX is being built; this may not be appropriate when preparing binary distributions for multiple systems in which case they can be explicitly enabled or disabled.

Name	Default	Description
Adobe Font Metrics	`AFM=auto`	This package contains Adobe Font Metric files that came from the public dvips distribution. AFM files are required by the textfmt(1) and faxmail(1) programs for converting ASCII text to PostScript. The AFM files should reflect the characteristics of the fonts that are used for imaging PostScript on the fax server. However if the PostScript RIP used to image text does not come with AFM files for its fonts, then this package can be substituted with only minimal degradation in the formatting of the imaged text. By default this package is configured for installation only if font metric files do not appear to be present on the build machine.

Display PostScript RIP	`DPS=no`	Whether or not to build and install a DPS-based PostScript imager for IRIX 4.x and 5.x systems from sources present in the HylaFAX build tree. This option is intended for people building binary distributions.
Dynamic Shared Object Support	`DSO=auto`	This package controls whether or not to configure the building of *Dynamic Shared Objects* (DSOs) for utility routines used by both client and server applications, and for a nucleus of common code used by server applications. Use of DSOs can significantly reduce the disk space needed for the HylaFAX software. If DSOs are not used then the code is statically linked into each application that requires it. By default this package is configured only if the build system appears to suport DSOs in a way that fits into the normal build scheme. If DSO support is *explicitly enabled* and there is no support for using DSOs in the expected way then DSOs are not used. Beware that DSOs can only be used when the compiler and runtime system support the runtime instantiation of global C++ objects that have constructors; something that few systems support.
Getty Support	`GETTY=auto`	This package controls whether or not to configure BSD- or System V-style getty support for handling incoming data connections. By default this package is configured according to the requirements of the target system.
Ghostscript RIP	`GS=no`	Whether or not to build and install a PostScript imager from Ghostscript sources present in the HylaFAX build tree. This option is intended for people building binary distributions.
Impressario RIP	`IMP=no`	Whether or not to build and install an Impressario 2.1 plug-in DSO from sources present in the HylaFAX build tree. This option is intended for people building binary distributions.
HTML Documentation	`HTML=no`	This package contains this HTML documentation about HylaFAX. By default this package is not configured for installation. HTML documentation can be viewed directly from the source directory; otherwise all the documentation can be viewed from the main WWW site at http://www.vix.com/hylafax/. This option is intended for people building binary distributions.
PostScript RIP	`PS=auto`	This package selects which RIP to use for imaging PostScript on the server. Possible values are: `gs` for Ghostscript, `dps` for the Display PostScript-based RIP provided for IRIX 4.x and 5.x systems, or `imp` for a plug-in DSO for Impressario 2.1 under IRIX. By default the DPS support is selected for IRIX systems; otherwise Ghostscript is used.

Regular Expression Support	`REGEX=yes`	This package controls whether or not to use the regular expression package included with the software. On systems where there is POSIX regular expression support in the standard system library use of this package may be disabled. Note that if this package is disabled the location of the include files and library containing the regular expression support must be specified; see the `REGEXINC` and `LIBREGEX` configuration parameters.
SGI RGB Image Converter Support	`SGI2FAX=auto`	This package controls whether or not to configure the sgi2fax program that converts SGI RGB images to TIFF/F (for direct transmission as facsimile). By default this package is configured for installation only if the build system appears to have support for the SGI Image library that sgi2fax uses to read RGB images. If this package is not installed then SGI RGB images must be converted to a form suitable for transmission using some other mechanism.
System V Init Support	`SYSVINIT=auto`	This package controls whether or not to configure the System V-style support for automatically starting the HylaFAX queuer process from init(1). By default this support is configured for installation only if the target system appears to use a System V-style init (`auto`); i.e. the **/etc/rc0.d** and **/etc/rc2.d** directories exist.
UTMP/WTMP Support	`UTMP=auto`	This package controls whether or not system accounting work should be done using the normal **utmp** files and data structures or the extended **utmpx** facilities. By default the appropriate support is chosen based on whether or not the build system has the **<utmpx.h>** file (`auto`). Explicit support can also be chosen by setting this parameter to `utmp` (normal accounting) or `utmpx` (extended accounting).
zlib Support	`ZLIB=yes`	This package controls whether or not to use the zlib compression package included with the software. On systems where the zlib distribution has already been installed this package may be disabled. Note however that if this package is disabled the location of the include files and library for the zlib distribution must be specified; see the `ZLIBINC` and `LIBZ` configuration parameters.

Configuring the HTML Documentation

HylaFAX comes with extensive documentation written in HTML, the language used to author many documents found on the World Wide Web (WWW). These materials can be viewed at the main WWW site http://www.vix.com/hylafax/, they can be installed on a local WWW server, or they can be viewed directly from the source area. Viewing the documentation through a WWW server is preferred because

the materials include links to reference manual pages that are accessed through CGI scripts, and these links will not be available when the documentation is accessed directly from the source distribution.

To configure the build environment so that it will install the HTML documentation, you need to request the ‘‘HTML package’’ when running the configure script. This is done by specifying `-with-HTML` as a command-line option to **configure** or by setting `HTML=yes` in a **config.site** or **config.local** file. When the HTML package is setup for installation there are four other parameters that should be defined according to local conventions:

Parameter	Default	Description
`DIR_HTML`	**/var/httpd/hylafax**	The directory where HTML materials should be installed.
`DIR_CGI`	**/var/httpd/cgi-bin**	The directory where CGI scripts should be installed.
`CGIPATH`	**/cgi-bin**	The pathname to use in HTML documents to reference scripts installed in `DIR_CGI`.
`HTMLPATH`	**/hylafax**	The pathname to use in HTML documents to reference materials installed in `DIR_HTML`.

(NB: the default values are suitable for the NCSA HTTP server).

Finally, to complete the installation the HTML documents must be edited to *fix pathnames* that are found in the HTML files. All instances of `@HTMLPATH@` and `@CGIBIN@` must be replaced with the values of `DIR_HTML` and `DIR_CGI`, respectively. (This was done automatically in previous versions of HylaFAX but must now be done manually.) A shell script to do this is:

```
#! /bin/sh
ROOT=${1-/hylafax}
CGI=${2-/cgi-bin}

patch()
{
   chmod +w $1
   ed - $1<<EOF
g;@HTMLPATH@;s;;$ROOT;g
g;@CGIBIN@;s;;$CGI;g
w
q
EOF
    chmod -w $1
}

FILES=`grep -l "$PATTERN" *.html */*.html`
test "$FILES" && {
   for i in $FILES
   do
       echo $i; patch $i;
   done
}
```

There are a few other issues to consider when installing the CGI scripts:

- The `manpage` script that provides access to the UNIX manual pages may require some tailoring if manual pages are installed in non-standard locations or if the normal system man(1) command does not support the expected options. If you have problems consult the script.
- If local support is to be provided, it may be worthwhile to alter the electronic mail addresses

embedded in several of the CGI scripts.

Upgrading From FlexFAX

If you were previously using FlexFAX beware that several things have changed in incompatible ways. The following is a list of things you should be aware of when configuring the latest distribution:

- The default server operation parameters set in **config.h** have changed; for example the default kill time for a fax job is now 3 hours instead of 24 hours. Verify the parameters in **config.h** are what you want.
- Numerous pathnames may have changed. The source-based installation procedure does not remove previously installed files unless they are to be overwritten; this can result in multiple versions of applications such as faxaddmodem being installed. Verify the parameters used for installation; you may want to tailor them to your environment by creating a **config.local** file with local configuration parameters.
- HylaFAX includes an improved **install.sh** script that is used to install files on non-IRIX systems. Certain files in the distribution are treated as *configuration-specific files* and are not overwritten if they appear to have local modifications. Instead the **install.sh** script will install the new file with a **.N** suffix and print a warning message. Be sure to reconcile these conflicts by comparing the old and new file contents with a program like diff(1); in most cases you should incorporate any local modifications into the new file and install it for use.

There are other issues with upgrading from FlexFAX; they are discussed in the chapter that describes setting up the server.

A Sample Configuration Session

This section shows a sample configuration session and describes the work done. The session on the left in a `fixed width font` with user-supplied input in a **`bold font`**. Commenttary is shown on the right in a normal or *italic* font or across both columns. Blank space is present in the session output where it is needed to fit comments on the page; in normal operation the output from configure does not include this white space.

This session was collected on a 486 machine running BSD/OS 1.1.

```
wullbrandt% mkdir fax
wullbrandt% cd fax
wullbrandt% ln -s /hosts/oxford/usr/people/sam/fax src
```

A build tree separate from the source tree is used here. In fact, in this case the distribution is accessed from a read-only NFS-mounted filesystem.

```
wullbrandt% src/configure

Configuring HylaFAX (tm) (aka FlexFAX) v3.0beta095.

If configure does the wrong thing, check the file config.log for
information that may help you understand what went wrong.
```

Note that the version and the deduced target configuration (`i386-unknown-bsdi1.1`) are printed. The file **config.log** should

```
information that may help you understand what went wrong.

Reading site-wide parameters from src/config.site.
Fee, fie, foe, this smells like a i386-unknown-bsdi1.1 system.
```

be checked if something goes wrong during configuration; it has output from the tests that configure does.

```
Using /usr/local/bin/gcc for a C compiler (set CC to override).
Looks like /usr/local/bin/gcc supports the -g option.
Looks like an ANSI C preprocessor.
... but __ANSI_CPP__ is not automatically defined, will compensate.
Looks like /usr/local/bin/gcc supports the -M option for make dependencies.
```

configure checks the normal shell search path for ANSI C compilers. A compiler is selected if it properly compiles a small test program. A specific compiler may be selected by setting the `CC` environment variable to the appropriate pathname, by supplying the parameter on the command line, e.g. `-with-CC=gcc`, or by setting `CC` in a configuration file.

Note that an ANSI C compiler is required to build the software. If a C compiler requires options to enable ANSI C compilation, these options can be specified with the `ENVOPTS` parameter.

```
Using /usr/local/bin/gcc for a C++ compiler (set CXX to override).
Looks like /usr/local/bin/gcc supports the -g option.
Using " -g" for C++ compiler options.
Looks like an ANSI C++ preprocessor.
... but __ANSI_CPP__ is not automatically defined, will compensate.
```

A C++ compiler is selected using a scheme similar to the one used to find a C compiler. Likewise there is a `CXX` parameter that can be set to explicitly select a C++ compiler.

```
Using /usr/bin/make to configure the software.
Using ".include <file>" syntax for Makefiles.
```

The make selected must support include files. configure can deduce several different syntaxes for specifying include files. Note that a large number of configuration parameters can be changed by editing the top-level **defs** file that is included by each Makefile in the distribution. A `MAKE` parameter can be used to control which make to use.

```
Using /bin/bash to process command scripts.
```

All the HylaFAX shell scripts are written using Bourne shell syntax. The standard shell on many systems is incapable of processing these scripts so configure prefers **bash** to **ksh** to **sh**.

```
Looks like -lutil is needed for wtmp file logging.
Looks like -lm is the library for math functions.
```

Various system-specific libraries that may or may not be needed are checked for. If your system requires a library that is not automatically included it can be specified by setting the `MACHDEPLIBS` parameter.

Creating port.h. The **port.h** file is included by all the C and C++ code in the system. It includes definitions for functions and types that are missing from system include files, `#defines` to enable or disable system-specific functionality, and other odds and ends.

```
Creating port.h with necessary definitions.
... open FIFO files read-only
... use (sig_t) for signal handler type
... use (sig_t) for sigvec handler type
... use (sig_t) for sigaction handler type
... configure use of mmap for memory-mapped files
... configure use of fchown
... add function prototype for cuserid
... configure use of fchmod
... add function prototype for setutent
... add function prototype for endutent
... add function prototype for pututline
... configure use of <locale.h> (internationalization support)
... configure use of <paths.h>
... configure use of <sys/select.h>
... configure use of logwtmp (BSD-style wtmp logging)
... add function prototype for logwtmp
... configure use of logout (BSD-style utmp support)
... add function prototype for logout
Done creating port.h.
```

This file can take a long time to generate so configure creates the file only when it is needed, either because the file does not exist or because a different target or compiler is to be used. Note that running "`make distclean`" in the top-level directory of the build tree will remove the **port.h** file (along with all the other files generated by configure).

Selecting emulated library functions. Certain functions used by HylaFAX are not present on all systems but can be emulated using other system functionality. configure checks for the presence of required functions and if they are missing, will configure emulation code from the **port** directory to use instead. Building HylaFAX on unsupported systems may require adding to the code to the **port** directory.

```
Checking system libraries for functionality to emulate.
... emulate cuserid
Done checking system libraries.
```

If a routine must be emulated and configure does not automatically check for it, the routine name can be specified using the `PORTFUNCS` parameter. To add emulation support for a new function `foo`, create a file **port/foo.c** that contains the emulation code and then set `PORTFUNCS=foo` in a configuration file or modify the configure script to automatically check for the missing function.

```
Checking for TIFF support.
Using TIFF include files from /usr/local/include
Using pre-built TIFF library -L/usr/local/lib -ltiff
Done checking for TIFF support.
```

HylaFAX uses the TIFF library extensively. By default the TIFF library and tools are assumed to be instaled in the default location used by the TIFF distribution.

If the TIFF library is installed as a DSO then the test done by configure will fail unless the DSO is in a well-know location or is explicitly searched for; e.g. with the `LD_LIBRARY_PATH` environment variable.

```
Checking for Dynamic Shared Object (DSO) support.
Done checking for DSO support.
```

If the `DSO` package is enabled (`DSO=auto` or `DSO=IRIX`), then configure will verify the system and compiler are capable of constructing DSO's in the expected way. Note that while a system may support DSO's the compiler may not be capable of generating the required position-independent code and/or the compiler may not pass the needed options through to the loader.

Selecting utility programs. configure locates various system utility programs that are used by the software. Note that absolute pathnames are selected because several scripts and programs excute with super-user privileges.

```
Selecting programs used during installation and operation.
Looks like /usr/bin/awk should be used to process command scripts.
Looks like /usr/sbin/sendmail should be used to deliver mail.
Looks like /usr/bin/mkfifo creates FIFO special files.
Looks like /bin/mv supports the -f option to force a move.
Looks like /bin/ln supports the -s option to create a symbolic link.
Done selecting programs.
```

Several things to note: The awk program selected for use must support functions using the POSIX-style syntax (i.e. using `function` and not the older `func` keyword). configure will not select an awk program that does not support the syntax used by the awk scripts included in the distribution. Mail delivery is done using sendmail; a different mailer cannot be substituted without changing certain scripts because a number of sendmail-specific command line options are used.

Selecting default configuration parameters. The remainder of the work done by configure involves setting up configuration parameters that are mainly used during the *operation* of the fax software. These parameters can be set during configuration or, in most cases, they can be set through runtime configuration files.

```
Using uid uucp and gid dialer for controlling access to fax stuff.
Using uid bin and gid bin for installing programs.
Using LSB2MSB bit order for your i386 cpu.
Looks like you need BSD getty support.
Looks like /usr/libexec/getty is the program to exec for a data call.

Looks like /bin/vgetty as the program to exec for a voice call.
Looks like /bin/egetty as the program to exec for an extern call.
Looks like you use binary-style UUCP lock files.
Looks like UUCP lock files go in /var/spool/uucp.
Looks like font metric information goes in /usr/contrib/lib/flexfax/afm.
Looks like the gs imager package should be used.
Looks like /usr/contrib/bin/gs is the PostScript RIP to use.
Looks like manual pages go in /usr/local/man.
Looks like manual pages should be installed with bsd-nroff-gzip-0.gz.

HylaFAX configuration parameters are:

[ 1] Directory for applications:        /usr/local/bin
[ 2] Directory for lib data files:      /usr/local/lib/fax
[ 3] Directory for lib executables:     /usr/local/lib/fax
[ 4] Directory for system apps:         /usr/local/bin
[ 5] Directory for manual pages:        /usr/local/man
[ 6] Directory for HTML documentation:  /usr/local/doc/fax
[ 7] Directory for spooling:            /var/spool/fax
[ 8] Directory for font metrics:        /usr/contrib/lib/flexfax/afm
[ 9] Directory for uucp lock files:     /var/spool/uucp
[10] Uucp lock file scheme:             binary
[11] PostScript imager package:         gs
[12] PostScript imager program:         /usr/contrib/bin/gs
[13] Manual page installation scheme:   bsd-nroff-gzip-0.gz
[14] Default page size:                 North American Letter
[15] Default vertical res (lpi):        98
[16] Location of getty program:         /usr/libexec/getty
[17] Location of voice getty program:   /bin/vgetty
[18] Location of sendmail program:      /usr/sbin/sendmail
[19] Location of TIFF tools:            /usr/local/bin

Are these ok [yes]?


Creating defs from src/defs.in
Creating rules from src/rules.in
Creating Makefile from src/Makefile.in
    ...stuff deleted...
```

*The **egetty** and **vgetty** programs are not part of HylaFAX; they are ''hooks'' for developers to integrate programs that implement voice-oriented capabilities.*

At this point you can interactively modify any of the displayed parameters. Hitting a carriage return or typing `yes` will accept the current parameters. Typing one of the numbers displayed along the left hand side causes configure to prompt for a new value of the specified parameter. Typing anything else causes configure to prompt for a new value *for each parameter*. In general hitting carriage return will accept the current value and typing anything that is unacceptable will cause a help message to be displayed. See below for descriptions of the parameters.

Once parameters are setup configure generates all the files that depend on these parameters. Note

```
Setting up make dependency files.
Done.
```

that certain files may or may not be created based on the selection of optional packages and/or the functions supported by target system.

Configuration Parameters

This section gives a brief description of the less obvious configuration parameters. Consult the distributed **config.site** for a *complete* list of parameters. The list here is sorted alphabetically.

The default setting for most configurations parameters is usually correct. Be especially careful about enabling or overriding parameters that control workarounds for system bugs; certain problems can cause server processes to go into infinite loops!

Note that HylaFAX is composed of *client* and *server* applications. Client applications are programs that normal users invoke to send facsimile, query the status of facsimile servers, etc. Server applications are programs that reside *only on the machine where the fax modems are present*.

Parameter	Description
AROPTS	The options passed to ar when creating an archive. Note that configure will automatically check to see if ar supports an `s` to create a symbol table instead of using ranlib.
CONFIG_ABORTBUG	On some systems the logic used by the server processes to poll for messages on a FIFO special file while actively doing work can cause the process to go into an infinite loop. Setting this parameter to `yes` causes the server processes to **not** poll for messages. Enabling this workaround means that the faxgetty process will not recognize messages to abort a facsimile while it is being received.
CONFIG_BADEXECVEPROTO	On some systems the prototype function declaration for the `execve` system call is wrong (it does not list certain parameters as `const`). Setting this parameter to `yes` causes the software to work around this problem.
CONFIG_BADEXECVPROTO	On some systems the prototype function declaration for the `execv` system call is wrong (it does not list certain parameters as `const`). Setting this parameter to `yes` causes the software to work around this problem.
CONFIG_BADGETOPTPROTO	On some systems the prototype function declaration for `getopt` is wrong (it does not list certain parameters as `const`. Setting this parameter to `yes` causes the software to work around this problem.
CONFIG_BADSELECTPROTO	On some systems the prototype function declaration for `select` passes sets of file descriptors as `int*` instead of `fd_set*`. Setting this parameter to `yes` causes the software to assume the `int*` form is required.

CONFIG_FIFOBUG	Whether or not to enable a workaround for a kernel bug that causes select system calls to return prematurely after a process closes the ‘‘client side’’ of a FIFO special file. Setting this parameter to `yes` causes server processes to close and reopen FIFO special files after each received message. Note that aside from the additional overhead incurred in the server processes, this problem can also cause client programs to be temporarily unable to reach a server process. To minimize this possibility, the client code tries to reach a server process up to 5 times before giving up; this can introduce noticeable delays in applications such as faxstat.
CONFIG_NOREOPEN	Whether or not to reopen the tty device that a modem is connected to after raising and lowering the `DTR` signal (to reset the modem). This is done because on some systems lowering `DTR` causes the device to be placed in an unusable state. It may be desirable to disable this action however; for example when a modem is connected to an Ethernet-based terminal server. Setting this parameter to `yes` causes the modem to be reopened after reset. By default this parameter is set according to the target system.
CONFIG_NOSTDINDUP	On some systems, if the standard output is redirected to be the same as the standard input, then the stty program will emit a warning message. This is necessary for some older systems because it is the only way to force stty to change parameters on a tty device that is not the controlling tty. Setting this parameter to `yes` causes the ondelay program to not set stdout to stdin. Enabling this workaround can break the faxaddmodem program; it should be used only on systems where stty supports the `-f` option to select a tty device other than the controlling tty.
CONFIG_OPENFIFO	The mode to use when opening a FIFO special file in a server process. Normally this should be `O_RDONLY`, but on some systems this must be `O_RDWR` to avoid kernel bugs that cause select system calls to return prematurely after a process closes the ‘‘client side’’ of a FIFO. By default this parameter is set according to the target system.
CONFIG_SELECTBUG	On some systems where the FIFO select bug (see above) exists the select system call may return bits set in the read, write, and exception masks even if they were not requested; this in turn can cause the Dispatcher code to abort. Setting this parameter to `yes` causes the Dispatcher to ignore any bits set in bit vectors returned by select if the bits were not requested. Enabling this workaround adds additional overhead to each server process.
CONFIG_SOCKARGLENTYPE	On some systems the type of call-by-reference parameters that specify the size of variable-length buffers in socket-related calls is wrong (the type is something other than `int*`). This parameter can be set to the C type that should be used for length parameters; e.g. ‘‘`unsigned long`’’; otherwise `int` is assumed. Setting this parameter incorrectly should result in compilation errors.

CONFIG_TIOCMBISBYREF	Whether to pass the argument to the TIOCMBIS ioctl by value or by reference. On most systems the value is passed by reference; consult termio(7) for your system if you are unsure how your system works. Setting this parameter to `yes` is the default and generates code for passing the argument by reference.
CONFIG_WINSZHACK	On some systems the `TIOCWINSZ` ioctl is defined, but its use requires the inclusion of two non-standard files. Setting this paramter to `yes` causes the inclusion of these files.
CXXFILE	The options to the C++ compiler required to get the compiler to process a file with a **.c++** suffix as C++ source code. configure automatically sets this parameter to `"-x c++"` for gcc and to `"-+"` for the IBM xlC compiler under AIX.
DEFVRES	The default vertical resolution in lines/inch that clients should use for submitted facsimile. Choices are 98 and 196.
DIR_AFM	The directory where client applications should find Adobe Font Metric (AFM) files. These files are used by various HylaFAX programs when converting text to PostScript.
DIR_BIN	The directory where client applications should be installed; by default this is **/usr/local/bin**.
DIR_CGI	The directory where CGI scripts used by the HTML-based HylaFAX documentation should be installed; by default this is **/var/httpd/cgi-bin** (usually appropriate for the NCSA HTTP server).
DIR_HTML	The directory where HTML-based HylaFAX documentation should be installed; by default this is **/var/httpd/htdocs/hylafax** (usually appropriate for the NCSA HTTP server).
DIR_LIBDATA	The directory to install data files that are required by client applications; by default this is **/usr/local/lib/fax**.
DIR_LIBEXEC	The directory to install executable programs that are invoked by other applications and not from the command line (e.g. the textfmt program invoked by sendfax to convert text to PostScript when submitting a fax job); by default this is the same as `DIR_SBIN`.
DIR_LOCKS	The directory in which to create UUCP lock files.
DIR_MAN	The top-most directory of the manual area where client/server manual pages should be installed.
DIR_SBIN	The directory where system applications such as servers should be installed; by default this is **/usr/local/sbin**.
DIR_SPOOL	The directory in which the HylaFAX server spooling area should be setup; by default this is **/var/spool/fax**.

DSODELAY	When DSO's are built, the option to specify to CC and/or CXX to note that a library should be loaded only as needed. This option is used to delay the loading of the TIFF library for applications that may use it infrequently.
DSOOPTS	When DSO's are built, the options to specify to CC and/or CXX to create a DSO.
DSOSUF	When DSO's are built, the filename suffix for a DSO. If this is set to "a" then statically linked archives are used.
ENVOPTS	Options to pass to CC and CXX to force ANSI C compilation.
FAXGID	The group ID to use for the fax user; by default this is selected according to the target system.
FAXUID	The user ID to use for the fax user; by default uucp.
FILLORDER	The order of bits in a byte on the server machine; either LSB2MSB or MSB2LSB. This is normally selected according to the target system.
GCOPTS	Special options to pass the C compiler. If this parameter is set, then configure may append other options to this list.
GCXXOPTS	Special options to pass the C++ compiler. If this parameter is set, then configure may append other options to this list.
INSTALL	The pathname of the install program to use. Note that this program must emulate the command line interface used by the IRIX install program.
LIBMALLOC	Whether or not to use -lmalloc in building the software. By default (auto) configure will probe the system to see if the library is present; if it is then the software will be configured to use it when building. Other possible values are: yes and no to enable and disable use, respectively. On systems that support DSOs it may be important to disable the use of -lmalloc to avoid conflicts with the normal memory alocation routines present in the standard C library.
LIBPORT	The pathname of the library that holds code to emulate missing system functionality. Normally this parameter is set by configure based on whether or not emulation code is required for the target.
LIBSUN	Whether or not to use -lsun in building the software. By default (auto) configure will probe the system to see if the library is present; if it is then the software will be configured to use it when building. Other possible values are: yes and no to enable and disable use, respectively.
LIBREGEX	The command line parameter(s) to use to link the library containing regular expression support. This parameter is set by default to reference the software included in the distribution.
LIBTIFF	The command line parameter(s) to use to link against the TIFF library. This parameter is set by default to the default installation location used by the TIFF software distribution.

LIBZ	The command line parameter(s) to use to link against the zlib library. This parameter is set by default to reference the software included in the distribution.
LLDOPTS	Extra command line options passed to CC and CXX when linking an executable. This option is usually set only when DSO support is enabled (to force the executable to search for the HylaFAX-specific DSO's in non-standard locations in the filesystem.)
LOCKS	A specification of the default type of UUCP lock files to use. This parameter is usually one of ascii, binary, -ascii or +ascii; consult the config(4F) manual page for a description of the **UUCPLockType** configuration parameter.
MACHDEPLIBS	Target-dependent libraries that should be used when linking applications. Note that if this parameter is specified configure will append to the list of libraries.
MAKECXXOVERRIDE	A special parameter used by configure when constructing Makefiles. This parameter is required when compiling the software with the SunPRO C++ compiler; it works around limitations in the SunPRO compilation environment.
MAKEDEPINCLUDE	The keyword used to specify the inclusion of a make dependency file when constructing Makefiles. This parameter is set to MAKEINCLUDE if make dependendcies are to be constructed by the software; otherwise it is set to # so that no make dependency rules are included (and used).
MAKEDSOINCLUDE	The keyword used to specify inclusion of a file containing rules for building DSO's. This parameter is set to MAKEINCLUDE if DSO's are to be built; otherwise it is set to # so that nothing is included.
MAKEINCLUDE, MAKELQUOTE, MAKERQUOTE	The syntax used to specify an include file for make; e.g. @MAKEINCLUDE@ @MAKELQUOTE@*file*@MAKERQUOTE@. These parameters are normally deduced by configure depending on the capabilities of the make program to use.

MANSCHEME	The scheme to use when preparing and installing manual pages. Schemes are constructed according to: <*organization*>-<*formatting*>-<*compression*>[-<*suffix*>] where: <*organization*> is either `bsd` for BSD-style section organization (e.g. file formats in section 5) or `sysv` for System V-style organization (e.g. file formats in section 4). <*formatting*> is either `nroff` to force installation of formatted materials (using nroff) or `source` to get the nroff source installed. <*compression*> is either the name of a program to compress the manual pages (gzip, compress, pack) or `cat` for uncompressed data. <*suffix*> is either the file suffix to convert installed pages to (e.g. 0.gz for gzip-compressed pages under BSD) or `strip` to force the normal ".4f" suffix to be converted to ".4" (or ".5" if using the BSD organization). If no -<suffix> is specified then filenames are not converted when they are installed.
PAGESIZE	The default page size that client applications will use for submitted facsimile. Page sizes are specified by name and checked against the pagesizes database installed in the `DIR_LIBDATA` directory. See also pagesizes(4F).
PATH_DPSRIP	The absolute pathname of the Display PostScript-based imager program.
PATH_EGETTY	The absolute pathname of suitable getty program to exec to deduce the type of an inbound call. Note that this parameter must be set correctly before the software is built; there is no mechanism for doing this through a runtime configuration file. *This program is not part of HylaFAX; it is provided as a ‘‘hook’’ for developers to integrate voice capabilities.*
PATH_GETTY	The absolute pathname of suitable getty program to exec to handle an inbound data call. Note that this parameter must be set correctly before the software is built; there is no mechanism for doing this through a runtime configuration file.
PATH_GSRIP	The absolute pathname of the Ghostscript-based imager program. Beware that this must be a version of the **gs** program that includes the tiffg3 device driver.
PATH_IMPRIP	The absolute pathname of the Impressario 2.1-based imager program available for IRIX 6.2 (and beyond) systems.
PATH_SENDMAIL	The absolute pathname of the sendmail program on the server machine; sendmail is used by HylaFAX to post mail messages for a variety of reasons.
PATH_VGETTY	The absolute pathname of suitable getty program to exec to handle an inbound voice call. Note that this parameter must be set correctly before the software is built; there is no mechanism for doing this through a runtime configuration file. *This program is not part of HylaFAX; it is provided as a ‘‘hook’’ for developers to integrate voice capabilities.*

PORTFUNCS	A list of non-standard functions that should be emulated. Normally this list is constructed by configure based on checks it does. If this parameter is set, configure will append to the specified list.
PROTOTYPES	Options that should be passed to the C compiler to force it to check for function prototype mismatches. This parameter should not be needed because ANSI C compilers should do this automatically.
REGEXINC	The pathname of the directory containing include files for the regular expression package. By default this parameter is set to reference the package included with this distribution.
SCRIPT_SH	The absolute pathname of the shell program to use when building HylaFAX and to use to execute certain command scripts. This shell must support the Bourne shell command syntax. Beware that many versions of bash have bugs in them that make them unsuitable for use.
SETMAKE	If make does not automatically set `$MAKE` to the name of the make program to invoke for subdirectories, then configure will create an explicit definition. If this parameter is set, then it will be used instead.
SHDLIBC	A special library to use in place of the standard `-lc` implicitly provided by `CC` and/or `CXX`.
SIGHANDLERTYPES	The type signatures to use when checking for the correct type for a signal handler. If this parameter is set then configure will check these types before it checks a builtin list of well-known values.
SYSGID	The group ID to use for installing non-setgid programs; by default this is chosen according to the target system.
SYSUID	The group ID to use for installing non-setuid programs; by default `bin` is used.
TIFFBIN	The pathname of the directory where the TIFF software distribution tools have been installed.
TIFFINC	The pathname of the directory where the TIFF software include files have been installed.
ZLIBINC	The pathname of the directory containing include files for the zlib distribution. By default this parameter is set to reference the package included with this distribution.

Troubleshooting Build Problems

Most compilation problems are because the configure script did not properly setup the software for building. If you have problems compiling the software on a *supported system* check the output from configure and the contents of the **port.h** file generated by configure to make sure it makes sense. Another byproduct of the configuration procedure is the **config.log** file that is found at the top of the

build tree. This file contains information about the tests done by the configure script; it should also be studied to check for problems.

The most common problem encountered during building, which is frequently not recognized until later when a program does not work correctly, is when the system include files are lacking proper ANSI C function prototypes. In particular beware of compilation warnings about ‘‘*passing an object as an argument to a function*’’, this usually means that a function was defined without a function prototype and then a C++ object was passed through the interface as an argument. For example, if the C library `strchr` function is declared as

```
extern char* strchr();
```

instead of the expected

```
extern char* strchr(const char*, int);
```

then the C++ compiler will not know to do the implicit cast of the first parameter to a `const char*`; almost certainly causing problems. Unfortunately many older systems do not have system include files that include ANSI C function prototype declarations; if this is true of your system then you are unlikely to get HylaFAX to build correctly.

System-specific Guidance

This section contains some build-related issues that are dependent on the operating system installed on the build machine. The information included here is by no means exhaustive; it reflects feedback from users accumulated over multiple HylaFAX versions and/or operating system releases.

AIX Guidance

To use the IBM xlC C++ compiler you must have a version that supports the `-+` option so that filenames with a .c++ extension are processed properly by the compiler.

Under at least AIX 4.2 some socket-related function declarations have been changed to use non-standard types. To workaround compilation problems the `CONFIG_SOCKARGLENTYPE` configuration parameter should be specified to reflect the appropriate type of the call-by-reference length parameter passed to `accept`, `getpeername`, and `getsockname`.

BSD/OS Guidance

Beware that the standard BSD/OS distributions place many files and applications in non-standard locations. This can cause confusion when HylaFAX is installed because the old applications distributed with BSD/OS may be accidentally run and do the wrong thing. Under BSD/OS 2.1 the following **config.local** file may be used to configure a build tree that installs over top of the standard BSD/OS distribution of HylaFAX.

```
DIR_BIN="/usr/contrib/bin"              # client apps
DIR_MAN="/usr/contrib/man"              # manual pages
```

```
DIR_AFM="/usr/contrib/lib/hylafax/afm"  # AFM files
DIR_LIBDATA="/usr/contrib/lib/hylafax"  # client data
DIR_LIBEXEC="/usr/contrib/lib/hylafax"  # libs&hidden apps
DIR_SBIN="/usr/contrib/bin"             # system apps
DIR_SPOOL="/var/spool/fax"              # spooling area
PATH_GSRIP="/usr/contrib/bin/gs"        # Ghostscript RIP
```

Note that you will also need to specify the location of the TIFF software because the version distributed with BSD/OS 2.1 is too old to use.

Dell SVR4 Guidance

On some releases **<sys/mkdev.h>** includes static functions with non-ANSI C definitions; these must be manually corrected before building the software. (The GNU gcc installation procedure attempts to correct this file but does not.)

The *ttymon* program is executable only by root; this may cause the configure script to not select it as the getty program that is started up for inbound data calls. To work around this problem override the default selection through the interactive prompts or by creating a **config.local** file that explicitly sets the **PATH_GETTY** configuration parameter.

FreeBSD Guidance

FreeBSD 2.0 does not include the file **<osfcn.h>**. If the software does not compile without this file a copy can be obtained from the GNU libg++ distribution: ftp://prep.ai.mit.edu/pub/gnu/libg++*.tar.gz, libg++-2.6.2/libg++/src/osfcn.h.

HP-UX Guidance

Under HP-UX 9.05 the standard make incorrectly processes VPATH; either use gmake or configure the software to be built directly in the source tree.

Some versions of HP-UX define the `select` system call with parameters that are of type `int*`; if so set `CONFIG_BADSELECTPROTO=yes` to work around the problem.

Some versions of HP-UX have the include file **<utmpx.h>** but do not support the extended accounting facilities defined by this file. This confuses the **configure** script; to work around the problem set `UTMP=utmp` when configuring the build environment.

Linux Guidance

Under various versions of Linux; when linking with the `-g` option; if the linker complains about needing **libc.so.4**, then it is necessary to setup symbolic links in **/usr/lib** so that: **libg.a** -> **libc.a** and **libg.sa** -> **libc.sa**.

Some recent versions of Linux come setup so that when C++ programs are compiled the standard C functions `strchr` and `strrchr` (and possibly others) have multiple definitions based on the types of their arguments. This overloading causes compilation problems because HylaFAX assumes the ANSI C definition which specifies a single definition for these functions. To workaround this problem either correct the include file **<string.h>** or re-install gcc and/or libg++ (the standard distributions do not cause

problems).

IRIX Guidance

Older versions of the SGI C++ compiler do not correctly handle nested types. If configure complains that the compiler is unsuitable for use or if the software does not build correctly verify that you have an up to date version of the compiler.

When building the software to use DSOs it may be necessary to set `LIBMALLOC=no` to avoid conflicts between the memory allocation routines in the standard C library and those in `-lmalloc`.

Under IRIX 5.3 **/bin/sh** has a bug that causes the faxsetup script to run correctly but terminate with an error. It may be preferrable to set `SCRIPT_SH=/bin/ksh` though it is not necessary.

SCO Guidance

Under SCO OS 5.0.0 compilation problems have been reported because of a missing declaration for the C typedef `id_t` used in the file **faxd/ModemServer.c++**. A workaround is to disable the use of the realtime scheduling support by setting the define `HAS_PRIOCNTL` to 0 in the file **port.h**.

Solaris Guidance

Under (at least) Solaris 2.3 the standard make does VPATH processing incorrectly for files passed to the mkdepend script (the last file in the list is not converted to a pathname relative to the source directory); this causes lots of msgs that can be safely ignored.

Some versions of Solaris include an old version of the TIFF library with the OpenWindows software. This can cause problems when running the configure script unless the `LD_LIBRARY_PATH` environment variable is explicitly set to reference the location of the newer TIFF library DSO or the path is explicitly set with a `-R` flag in the `LIBTIFF` configuration parameter.

Ultrix Guidance

[*Ed: this information comes from Tom Lislegaard.*]

Extensive work is required to get the software built and working. The following comments apply to at least Ultrix 4.4. These notes relate to building and running HylaFAX Version 3.0 with gcc-2.6.3 and libg++-2.6.2.

Two header files cause problems: **termios.h** and **sys/socket.h**.

termios.h defines tcsetattr, tcgetattr, etc in terms of macros that map to ioctl's; this causes problems with the configure script. As the real functions are present in libc, working around this problem is just a matter of bringing the include file up to date. (When compiling with GNU gcc the file to patch is normally **/usr/local/lib/gcc-lib/mips-dec-ultrix4.4/2.6.3/include/sys**):

```
123,125d122
< #if !defined(_POSIX_SOURCE)
```

```
< #define tcsetattr(fildes,action,termios_p)    \
<       ioctl(fildes,action,termios_p)
127,133d123
< #define tcgetattr(fildes,termios_p)   \
<       ioctl(fildes,TCGETP,termios_p)
< #else
< extern        int tcsetattr();
< extern        int tcgetattr();
< #endif
<
147,152d136
< #if !defined(_POSIX_SOURCE)
< #define tcdrain(fildes)               \
<       ioctl(fildes,TCSBRK,-1)
< #else
< extern        int tcdrain();
< #endif
161,166d144
< #if !defined(_POSIX_SOURCE)
< #define tcflush(fildes,queue) \
<       ioctl(fildes,TCFLSH,queue)
< #else
< extern        int tcflush();
< #endif
176,181d153
< #if !defined(_POSIX_SOURCE)
< #define tcflow(fildes,action) \
<       ioctl(fildes,TCXONC,action)
< #else
< extern        int tcflow();
< #endif
183,189d154
< /*
<  * Get input and output baud rates.
<  */
< #if !defined(_POSIX_SOURCE) && !defined(_XOPEN_SOURCE)
< #define OBAUD (CBAUD << 16)
< #define cfgetospeed(termios_p)                \
<       (((termios_p)->c_cflag & OBAUD) >> 16)
191,198c156,160
< #define cfgetispeed(termios_p)                \
<       ((termios_p)->c_cflag & CBAUD)
< #else
< extern        speed_t cfgetospeed();
< extern        speed_t cfgetispeed();
< extern        int     cfsetispeed();
< extern        int     cfsetospeed();
< #endif
---
> extern int tcsetattr(int, int, const struct termios*);
> extern int tcgetattr(int, struct termios*);
> extern int tcdrain(int);
> extern int tcflush(int, int);
> extern int tcflow(int, int);
199a162
> extern  int tcsendbreak _PARAMS((int, int));
```

/usr/include/sys/socket.h needs to be protected against multiple inclusion:

```
2a3,4
> #ifndef _SOCKET_
> #define       _SOCKET_
237a240
> #endif /* _SOCKET_ */
```

Many of the tools provided with Ultrix are outdated, functionally crippled, or just buggy. Affected here are sh, make, expr, and sed.

configure must be run by ksh; e.g.

```
% ksh configure
```

There is no hope for using **/bin/sh**, it does not support shell functions. The good news is that the current **/bin/ksh** handles all the scripts as well, so it is not necessary to install bash; simply use

```
SCRIPT_SH=/bin/ksh
```

A recent version of GNU make must be used (I have used version 3.67).

A replacement expr program is found in the GNU Shell Utilities package; it is needed in place of the default system version. Likewise for sed; I have used GNU Version 2.05.

The syslogd and printf programs are not available on Ultrix. On a host running only client software (no local modem) these may be ignored. On a server machine these are required.

The standard syslog support can not handle the HylaFAX server processes (or any other modern software for that matter). It should be replaced with a newer syslogd. Several alternatives are available, the stuff I have used can be found at ftp://gatekeeper.dec.com:/pub/DEC/jtkohl-syslog-complete.tar.Z.

(A note here: on my system I have done a complete replacement, including putting new syslog functions in my libc.a. This means that I have not tried to build with the **port/syslog.c** support included with HylaFAX; though it should be equivalent).

BUILDING GHOSTSCRIPT FOR USE WITH HYLAFAX

If you are not using the Display PostScript-based imager on a Silicon Graphics machine, then you need to build a version of Ghostscript that includes the tiffg3 device driver. This driver was included with Ghostscript starting with version 2.6.1. Ghostscript and its fonts can be found at many public FTP sites around the Internet. If you are unfamiliar with Ghostscript you can find information about it at http://www.cs.wisc.edu/~ghost/ghostscript/

To setup Ghostscript do the following:

1. Create a **Makefile** following the Ghostscript documentation.
2. Add `tiffg3.dev` to the list of configured devices in the **Makefile** if it is not already present (recent versions of Ghostscript automatically include it).
3. Build a **gs** executable that includes the tiffg3 driver.
4. Verify that you have configured your **gs** executable correctly by running it with the `-h` option and looking for the tiffg3 driver in the list of available devices; e.g.

```
hyla% gs -h
Aladdin Ghostscript version 2.9.9 (6/23/1994)
Copyright (C) 1990-1994 Aladdin Enterprises, Menlo Park, CA.
Usage: gs [switches] [file1.ps file2.ps ...]
Available devices:
        x11 tiffg3 tiffg4
Search path:
    .
    /usr/local/lib/ghostscript/gs-2.9.7:/usr/local/lib/ghostscript/fonts
Most frequently used switches: (you can use # in place of =)
```

```
    -d[=]   define name as token, or true if no token given
    -dNOPAUSE            don't pause between pages
    -gx   set width and height ('geometry'), in pixels
    -q                   'quiet' mode, suppress most messages
    -r              set resolution, in pixels per inch
    -s=    define name as string
    -sDEVICE=   select initial device
    -sOutputFile=  select output file: embed 268501088 for page #,
                         - means stdout, use |command to pipe
    -                   read from stdin (e.g., a pipe) non-interactively
For more information, see the (plain text) file use.doc
  in the directory (null).
```

5. Install the **gs** executable and the associated runtime support code (e.g. ‘‘make install’’).
6. Install, if necessary, the Ghostscript fonts (these may come separately from the Ghostscript source distribution).

Make sure that HylaFAX is configured to use the version of Ghostscript that you have installed. HylaFAX assumes that Ghostscript is installed in the **/usr/local** hierarchy; if this is not where you placed it then you will need to specify the location with the `PATH_GSRIP` configuration parameter when setting up a HylaFAX build environment.

*Note that on machines with dynamic shared libraries (e.g. SunOS), if you link Ghostscript with the X11 device driver and use shared X11 libraries that are not in a standard location, then you need to link Ghostscript with arguments that force it to specifically search these directories, or augment the HylaFAX **util/ps2fax.gs.sh** script; e.g.*

```
LD_LIBRARY_PATH=/usr/local/R5/lib:/usr/openwin/lib
export LD_LIBRARY_PATH
```

Beware that while many versions of Ghostscript between 2.6.1 and 3.12 have a tiffg32d driver that claims to support 2D-encoded facsimile data, but does so incorrectly. Ghostscript versions 3.13 and later generate correct 2D-encoded data.

Finally, beware that recent versions of Ghostscript require several additional packages such as the Independent JPEG Group (IJG) distribution for the PostScript Level II support. These additional software packages may come bundled in archives separate from the basic Ghostscript distribution. Consult the Ghostscript documentation for specifics.

Server Setup and Basic Configuration

HylaFAX is composed of *client* and *server* applications. Client applications are programs that normal users invoke to send facsimile, query the status of facsimile servers, etc. Server applications are programs that reside *only on the machine where the fax modems are present*. HylaFAX is distributed so that the normal `make install` step done after the software is built will install both client and server applications. Client-only systems require a slightly different procedure that is discussed in the next chapter on ‘‘Client Setup.’’ This chapter discusses setting up a server machine for use.

A system acting as a HylaFAX server usually runs at least two server processes: the HylaFAX scheduler process *faxq* and the client-server protocol process *hfaxd*. Server systems may also use the HylaFAX front-door process *faxgetty* to monitor each modem on the system and possibly receive incoming

facsimile calls. A send-only system would run faxq and hfaxd but (probably) not faxgetty. A system that was not going to use the transmit capabilities would not run faxq.

In addition to the server processes that operate all the time HylaFAX comes with two programs that are intended to be run periodically. The *faxqclean* program is responsible for purging unwanted files from the spooling area on a server and the *faxcron* script monitors the spooling area and performs routine maintenance tasks such as truncating log files. These programs are usually invoked by the system *cron* program.

The remainder of this chapter discusses the basic steps required to setup a HylaFAX server machine:

1. Install the HylaFAX software.
2. Select a facsimile modem for use.
3. Check your modem is functional.
4. Select a flow control scheme to use for facscimile communication.
5. Select a TTY device to use.
6. Use *faxsetup* to configure a server machine.
7. Use *faxaddmodem* to configure modems.
8. Startup outbound service.
9. Setup the modem for inbound use [*optional*].
10. Setup client access.
11. Setup periodic maintenance work.

Note that many decisions in setting up a server machine are dependent on the operating system present on the machine. Some system-specific guidance is sprinkled throughout these materials, with additional information provided in a section on system-specific guidance. There is also a section on ‘‘Modem Configuration Issues,’’ while advanced configuration and setup issues are discussed in a subsequent chapter titled ‘‘Advanced Server Configuration.’’

Installing HylaFAX

Most installations of HylaFAX will be done from a source code distribution. In this case installation on a server machine is done by running:

```
hyla# make install              # NB: must be done by the super-user
```

from the top-level of the build area. Note that *this step must be done as the super-user or file ownerships will be wrong*.

When working from a binary distribution the technique required to install the software programs varies based on the format used to distribute the materials. Each binary distributions of HylaFAX should include detailed installation instructions that reflect the exact contents of the distribution and any distribution-specific requirements.

Selecting a Facsimile Modem

Selecting a modem usually means purchasing a modem that is capable of sending and receiving facsimile. Most any fax-capable modem can be used with HylaFAX but not all are equal. HylaFAX has drivers for Class 1, Class 2, and Class 2.0 facsimile modems, but there are caveats on the use of Class 1 modems. Class 2 modems are the most common, but due to the non-standard nature of the specification that they are implemented against compatibility can be a problem. Note also that the quality of Class 2 modems varies significantly. Class 2.0 modems follow the latest standard, a ratified version of the specification used in implementing Class 2 modems. There are significantly fewer Class 2.0 modems available, though the quality of these modems also varies significantly. There is a (constantly growing) list of modems that have been tried with HylaFAX and this list includes several modems that have been found to be reliable for use in sending and receiving facsimile.

If you are buying a modem to use with this software you will do best if you use one of the recommended Class 2 or Class 2.0 modems; rather than a Class 1 modem.

Checking Your Modem

Once you have a modem to use with HylaFAX first make sure that the modem works for data use. One can not say this enough. If you can not use a communication program such as *cu*, *tip*, or *kermit*, with your modem, do not try to configure it for use with HylaFAX. This means in particular that you should verify that you have a working cable between your host and modem and that this cable is suitable for use. That is, that the cable has the relevant signals for doing hardware flow control if that is necessary and that it passes the DCD and DTR signals appropriately.

Verify that the modem you are using is a fax modem. This can be done by communicating directly with the modem using a communication program such as cu; for example:

```
hyla% cu -l ttyf2
Connected
at+fclass=?
0,1,2
OK
~[hyla].
Disconnected
```

The `at+fclass=?` command asks the modem to report which *classes* it is capable of supporting. Class 0 is for data use. Class 1, Class 2, and Class 2.0 are for facsimile use. Other classes may be reported, for example, for modems that provide digitized voice support. HylaFAX can be used with any modem that supports Class 1, Class 2, or Class 2.0.

Selecting a Flow Control Scheme

Flow control refers to the mechanism used to control the transfer of data *between the host and the modem* (there may also be a flow control scheme used in modem-to-modem communication but the discussion here is specific to the scheme used between the host and a modem). The rules to use for selecting a host-modem flow control method for facsimile use are:

- If you have a Class 1 modem, then you can use either hardware or software flow control, but beware of using hardware flow control as the Class 1 specification only requires vendors to support software flow control (and many of the Class 1 modems tried so far do not properly support hardware flow control).
- If you have a Class 2 or Class 2.0 modem, then you can use either hardware or software flow control, but if you are going to communicate with the modem faster than 9600 baud then you should probably use hardware flow control.

Beware that although a modem may properly implement hardware flow control when doing data communication, it may not support hardware flow control during facsimile communication. Consult the modem information for specifics on some modems.

If a prototype configuration file for your modem is included in the HylaFAX distribution then the appropriate default flow control scheme should be defined for the modem.

Note that some operating systems do not support RTS/CTS flow control unless carrier is present. In particular, some versions of SunOS and Solaris requires patches to correct this mis-behaviour; consult the sections below for system-specific guidance.

Versions of IRIX prior to 6.2 have a bug in the device driver for the on-board serial ports on IP20 and IP22 systems that causes RTS/CTS flow control to be turned off by HylaFAX. Patch 475 (''RTS/CTS flow control busted when `CLOCAL` is set'') and its successors correct this problem and must be installed to use HylaFAX with hardware flow control. For complete details refer to the section below on IRIX-specific guidance.

When in doubt or having trouble, configure the modem to use software flow control for fax use.

Choosing a TTY Device

There are two things to beware of in selecting a tty device file to use with your modem: flow control usage and port locking mechanisms.

On many systems different devices are used to select different flow control schemes and/or whether or not the system will monitor the DCD signal. For example, IRIX systems use **ttym*** and **ttyf*** device names to identify devices that monitor DCD but only the latter support RTS/CTS flow control. Similarly, the FAS driver for SCO uses a different names as does the standard HP-UX terminal driver.

On some systems inbound and outbound port use is interlocked by using a pair of devices, one for inbound use and another for outbound use. Typically this scheme works by stopping programs that use the inbound device until an inbound call is received (and DCD is raised by the modem). Outbound usage is also interlocked against applications waiting for the inbound device. HylaFAX provides no direct support for this because this scheme requires that a modem auto-answer incoming calls (something that is hard to make work with multi-mode--i.e. fax and data--modems). When faced with a system that uses this scheme most people elect to avoid the inbound device and run both incoming and outgoing traffic on the outbound device, using the built-in interlocking mechanism provided by HylaFAX. In this case the appropriate device to use is typically named **/dev/cu***, but check local conventions.

Using faxsetup to Configure a Server Machine

Before any HylaFAX software can be used on a server machine the faxsetup script must be run. This interactive script verifies the installation of the HylaFAX software and also carries out a variety of one-time tasks that prepare the system for use. Running faxsetup is especially important when working from a binary distribution because it checks that parameters setup at the time the distribution was built are correct for the target machine where the software is to be run.

faxsetup carries out a variety of tasks and then writes configuration information to two files in the HylaFAX spooling area. The file **etc/setup.cache** in the spooling area contains the parameter settings used by HylaFAX command scripts while the **etc/setup.modem** file contains settings and shell functions used by command scripts that communicate with modems.

*The **setup.cache** and **setup.modem** files must be present for HylaFAX to function properly. If these files do not exist then HylaFAX server applications will terminate with an error message.*

The work done by faxsetup includes the items listed below. Note that faxsetup always prompts for permission before doing anything that might affect normal system operation (e.g. adding a new user to the password file).

- faxsetup verifies that the pathnames compiled into HylaFAX applications are correct and that the expected directory hierarchy used by client and server applications is present and setup correctly. If any of these checks fails then it is assumed that HylaFAX has not been installed or that there is a misconfiguration problem such as might occur when a binary distribution is loaded in an unexpected location.
- faxsetup verifies that the TIFF software distribution is properly installed on the server machine. HylaFAX uses certain of the TIFF tools in normal server operation and, if the TIFF library has been built as a DSO, requires the TIFF library DSO for proper operation.
- faxsetup verifies that the server machine is properly configured to support FIFO special files. Some operating systems are distributed without support for this feature and must be reconfigured before HylaFAX can be used.
- faxsetup verifies that the configured PostScript RIP is present and that it has the necessary functionality to use it with HylaFAX. Ghostscript users must be sure to configure the ‘‘tiffg3’’ device driver when building Ghostscript for use with HylaFAX. IRIX users must be aware that the DPS-based RIP distributed for use with HylaFAX is a COFF executable that cannot be used under IRIX 6.2 or later.
- faxsetup verifies that a ‘‘fax’’ user is registered in the password file or YP/NIS database. If one is not setup then one is created. The fax user is used in various places in HylaFAX and should be setup to have the same user ID as that of ‘‘uucp’’ so that UUCP lock files can be shared. (If you run IRIX the ‘‘fax’’ user will also be added to the **passwd.sgi** file so that it does not show up in the normal palette of users displayed during the login procedure.)
- faxsetup verifies that suitable entries for the ‘‘hylafax’’ and ‘‘snpp’’ services exists in the **/etc/services** file or in the equivalent YP/NIS database. If no entries are present then they may optionally be setup (the software will still function properly without them).
- faxsetup verifies that the hfaxd client-server protocol process is started up when the system is

brought up multi-user or that hfaxd is started by the inetd(1M) program. hfaxd is the process that HylaFAX client applications communicate with to submit jobs, query server status, etc. It operates most efficiently when run in standalone but may also be invoked through inetd. Without hfaxd HylaFAX is basically useless.

- faxsetup verifies that a ‘‘FaxMaster’’ entry is present in the aliases database. This alias is equivalent to the normal PostMaster alias that is used to deliver mail-related problems; HylaFAX directs notices about problems and received facsimile to this alias. The FaxMaster alias should list those system administrators that will handle HylaFAX-specific problems. If an alias is not present, then one is created.

Following the above work faxsetup then prompts to create a configuration file for the HylaFAX scheduler process and for any modems on the system that are to be used by HylaFAX. Finally the HylaFAX server processes are started up, or restarted if an existing installation is being updated or reconfigured, and any modem configuration work is performed. The following sections elaborate on this work and provide examples of how this work is done.

Using faxaddmodem to Configure Modems

Modems are configured for use with the faxaddmodem script. This is an interactive script that walks you through the configuration and installation of a new or existing modem. Even if you have a previous version of HylaFAX or FlexFAX installed it is a good idea to run faxaddmodem to update the configuration information for your modems after installing a new distribution.

faxaddmodem may be run directly from the command line or via the faxsetup script that is used to prepare a server machine for use with HylaFAX.

The remainder of this section shows a sample configuration session and describes the work done. The session is shown on the left hand side in a `fixed width font` with user-supplied input in a **`bold font`**. Comments are shown to the right in a normal or *italic* font, separated by horizontal rules. Blank space is present in the session output where it is needed to fit comments on the page; in normal operation the output from faxaddmodem does not include this white space. faxaddmodem displays the current/default setting for a configuration parameter enclosed in ‘‘`[]`’’; the current value is accepted by typing a carriage-return.

This session was collected on a Silicon Graphics Indy machine running IRIX 5.3 and communicating with a USR Courier v.Everything modem.

```
hyla# faxaddmodem ttyf2

Ok, time to setup a configuration file for the modem.  The manual
page config(4F) may be useful during this process.  Also be aware
that at any time you can safely interrupt this procedure.
```

If your modem is configured to communicate with the host at a fixed speed, then use the `-s` option to lock the host-modem baud rate. Note also that faxaddmodem must be run as root.

Collect server-specific configuration parameters. Per-modem configuration files contain parameters that pertain to the operation of the modem and parameters that control the function of the HylaFAX server software that controls the modem. The former parameters are termed modem-specific while the latter are referred to as server-specific. faxaddmodem collects server-specific parameters first and then modem-related parameters are setup based on the type of modem.

```
Reading scheduler config file /var/spool/fax/etc/config.
```

The configuration file for the HylaFAX scheduler is automatically read to get default settings for parameters such as the area code.

```
No existing configuration, let's do this from scratch.
```

If a configuration file already exists for a modem, the server-specific parameters will be carried over to the new configuration, but the modem-specific parameters will not.

```
Country code [1]?
Area code [510]?
```

The area code may be set to a null string or omitted in locations where one does not exist.

```
Phone number of fax modem [+1.999.555.1212]? +1 510 526-8788
```

This is the phone number associated with the modem. By default this number is passed as an *identity* to peer fax machines (see below) and it may also appear on tag lines created by the fax server.

Phone numbers should be supplied with a complete international dialing specification: ''+<*country code*> <*local part*>'' (the <*local part*> includes an area code where such a notion exists.)

```
Local identification string (for TSI/CIG) ["NothingSetup"]? ""
```

The local identification string is passed to peer fax machines during communication. If it is not specified, or set to a null string, then the canonical phone number of the fax modem is used instead.

Beware that the facsimile communication protocol restricts the local identification string to numbers, blank, and the ''+'' symbol. In practice however, most facsimile machines will accept any ASCII string.

```
Long distance dialing prefix [1]?
International dialing prefix [011]?
Dial string rules file (relative to /var/spool/fax)
   [etc/dialrules.sf-ba]?
```

The dial string rules file holds rules used to convert user-supplied dialing strings (i.e. phone numbers) to a canonical format and to prepare strings for delivery to a modem. The default rules do very little. Specific dialing rules may be useful for your site or locale based on how your modems are connected to the PTT. Consult the section in the Advanced Server Configuration chapter for more information.

```
Tracing during normal server operation [1]?
Tracing during send and receive sessions [11]?
```

These parameters control the logging done by server processes.

It is important that tracing during send and receive sessions include sufficient information to diagnose problems. For Class 1 modems this parameter is usually set to `0x4f` so that HDLC frames are included in the logs. For Class 2 and Class 2.0 modems the default setting of 11 is typically ok. Consult the descriptions of `ServerTracing` and `SessionTracing` in the config(4F) manual page.

```
Protection mode for received facsimile [0600]?
```

The default parameter selected here makes all received facsimile accessible only to the fax user. It may be desirable to set this parameter to `0644` so that anyone can look at received facsimile.

```
Protection mode for session logs [0600]?
Protection mode for ttym2 [0600]?
```

Note that if sensitive information such as credit card access codes are supplied by users that they will appear in the session logs kept on the server machine. For this reason the default protection mode for session logs protects them from public access.

```
Rings to wait before answering [1]?
```

This parameter is used during inbound call handling. It specifies the number of rings to wait before answering the phone. See the section on inbound call handling for a discussion of the different schemes that are supported for handling calls.

```
Modem speaker volume [off]?
Command line arguments to getty program ["-h %l dx_%s"]?
```

This parameter is also related to inbound call handling. It controls whether or not to enable support for inbound data calls and should not be set without first understanding how to setup your system for incoming data usage.

```
Pathname of TSI access control list file
   (relative to /var/spool/fax) [""]?
```

The TSI access control list file can be used to restrict inbound facsimile access. The default parameter causes all inbound calls to be accepted. This parameter is discussed more below.

```
Tag line font file (relative to /var/spool/fax) [etc/lutRS18.pcf]?
Tag line format string ["From %%l|%c|Page %%p of %%t"]?
```

Tag lines are an optional feature that cause each page of an outbound facsimile to be marked with a line of text. A proper description of the tag line support is provided in the Tagline Configuration section of the next chapter.

Note that in the United States some form of identification of the sender of a facsimile is required by law; properly configured tag lines are an acceptable form of identification. Beware that the default tag line includes the phone number defined above; make sure this number is correct!

```
Time before purging a stale UUCP lock file (secs) [30]?
```

This parameter controls the time that a HylaFAX server process will wait before removing a UUCP lock file whose owner appears to be gone. The default parameter forces these stale lock files to be left around for just 30 seconds. This is an appropriate value

to use on systems where applications that use UUCP lock files are known to be well behaved.

```
Percent good lines to accept during copy quality checking [95]?
Max consecutive bad lines to accept during copy quality checking [5]?
Max number of pages to accept in a received facsimile [25]?
Syslog facility name for ServerTracing messages ["local5"]?
```

More parameters associated with inbound facsimile jobs; consult the section below.

The syslog facility controls where HylaFAX directs server tracing-related messages. By setting this parameter to a non-standard value HylaFAX messages can easily be recorded in a file separate from the normal system messages. Consult the Troubleshooting chapter, for more information.

```
Set UID to 0 to manipulate CLOCAL [""]? yes
```

This parameter should only need to be set on Silicon Graphics systems. Consult the section on IRIX-specific guidance for a description of why this parameter might be used.

```
The non-default server configuration parameters are:

CountryCode:            1
AreaCode:               510
FAXNumber:              +1 510 526-8788
LongDistancePrefix:     1
InternationalPrefix:    011
DialStringRules:        etc/dialrules.sf-ba
SessionTracing:         11
RingsBeforeAnswer:      1
SpeakerVolume:          off
GettyArgs:              "-h %l dx_%s"
LogFacility:            local5
TagLineFont:            etc/lutRS18.pcf
TagLineFormat:          "From %%l|%c|Page %%p of %%t"
MaxRecvPages:           25
ClocalAsRoot:           yes

Are these ok [yes]?
```

This completes the collection of server-related parameters; the remaining steps identify and configure the modem for use. If the displayed parameters are unacceptable, typing something other than `yes` or a carriage return will cause faxaddmodem to prompt for new/changed parameter settings.

Setup modem-specific configuration parameters. faxaddmodem probes the modem to find out what type it is and what capabilities it has. The modem identity is then matched against a set of prototype configuration files to come up with modem-specific parameters that are appropriate to the modem and the style of flow control that is to be used between the host and modem. If the modem is not supported by an existing prototype configuration then faxaddmodem will prompt to get values for parameters.

```
Now we are going to probe the tty port to figure out the type
of modem that is attached.  This takes a few seconds, so be patient.
Note that if you do not have the modem cabled to the port, or the
modem is turned off, this may hang (just go and cable up the modem
or turn it on, or whatever).

Probing for best speed to talk to modem: 38400 OK.
```

Use the `-s` option to avoid probing.

```
This modem looks to have support for both Class 1 and 2;
how should it be configured [2]?
```

Modems that support multiple classes can be configured to use any supported class. By default Class 2.0 is considered better than Class 2 which is preferred over Class 1.

```
Hmm, this looks like a Class 2.0 modem.
Modem manufacturer is "USRobotics Courier V.Everything".
Modem model is "Product type           US/Canada External".

DTE-DCE flow control scheme [default]?
```

The flow control scheme requested is

one of `xonxoff` for software flow control, `rtscts` for hardware flow control, or `default` for a setting that is appropriate for the modem and the tty device.

Beware of using an improper flow control scheme for the selected tty device. On systems where faxaddmodem understands how tty device names reflect flow control characteristics, selecting a flow control scheme not supported by the device will cause faxaddmodem to prompt for confirmation and/or to change the device name or the flow control scheme.

```
Using prototype configuration file usr-2.0...

The modem configuration parameters are:

ModemFlowControl:       rtscts
ModemHardFlowCmd:       AT&H1&I0&R2
ModemNoFlowCmd:         AT&H0&I0&R1
ModemRate:              38400
ModemResultCodesCmd:    ATQ0X4
ModemSetVolumeCmd:      "ATM0 ATM1 ATM1 ATM1 ATM1"
ModemSetupAACmd:        AT+FAA=1
ModemSetupDCDCmd:       AT&C1
ModemSetupDTRCmd:       ATS13=1&D2
ModemSoftFlowCmd:       AT&H2&I2&R1
Class2BUGCmd:           AT+FBU=0
Class2CQQueryCmd:       !(0),(0)

Are these ok [yes]?
```

The modem-specific configuration parameters are obtained from prototype configuration files that reside in the **config** subdirectory of the HylaFAX spooling area. These parameters have been taken from working systems and should provide a functioning configuration based on the modem type and the selected flow control scheme.

If no prototype configuration file exists for a modem then faxaddmodem will prompt for settings. There are generic prototype configuration files for Class 1, Class 2, and Class 2.0 modems. Because there are many configuration parameters for modems it may be preferrable to use a normal text editor instead of faxaddmodem when constructing a configuration file for an unsupported modem, To do this simply accept the default parameters and then edit the generated configuration file before starting up a server for the modem. Once a working configuration file is created it is simple to create a prototype file from it; consult the faxaddmodem(1M) or config(4F) manual pages for information on doing this.

```
Creating new configuration file /var/spool/fax/etc/config.ttyf2...
Done setting up the modem configuration.

Checking /var/spool/fax/etc/config for consistency...
...everything looks ok; leaving existing file unchanged.

Don't forget to run faxmodem(1M) (if you have a send-only environment)
or configure init to run faxgetty on ttyf2.
```

At this point faxaddmodem compares the parameters setup for the modem against the parameters setup for the HylaFAX scheduler process. If any parameters are incompatible it prompts to see if they should be used to create a new file for the scheduler.

Once a modem has been setup with faxaddmodem the HylaFAX scheduler process must be informed of its presence. This can be accomplished in one of two ways. If a system is to be used only for transmitting facsimile then the scheduler is informed by running the faxmodem command. Otherwise, if a modem is to be used for both inbound and outbound use then a faxgetty process should be setup to manage the modem--this process will automatically inform the scheduler once it has initialized the modem for use. There are also valid reasons to use faxgetty on modems that are to be used strictly for data; this and other variations in configuring the software are discussed later. If a modem was previously in use nothing needs to be done; the HylaFAX server processes will notice the new configuration file and automatically use its contents. If faxaddmodem was invoked by faxsetup then faxsetup should complete the steps necessary to complete the setup of a HylaFAX server machine.

Starting Outbound Service

Outbound service is carried out by the HylaFAX scheduler process, the faxq(1M) program. There is one faxq process for all modems on a system. The faxq program learns about modems that can be used for outbound jobs by messages it receives on a FIFO special file located in the HylaFAX spooling area on the server machine. These messages come from two sources: from the faxmodem program that is used to manually enable a modem for use, or from faxgetty processes that are setup to run on each tty device where a fax modem resides.

Specifying modems with faxmodem is useful when HylaFAX is to be used in a *send-only configuration*. Doing this however limits the functionality of the scheduler because it will not know the true state of each modem; e.g. when a modem is in use by an outbound application such as uucp or tip. Instead faxq will assume that each modem is ready for use except when it is actively being used by HylaFAX to transmit a facsimile or alpha-numeric page.

A modem specified with faxmodem is identified by the tty device it is attached to. Thus, to notify the scheduler that two modems are available for use, the following might be used:

```
hyla# faxmodem tty01
hyla# faxmodem /dev/tty02
```

(note that devices may be specified with or without a leading **/dev/** prefix.) Modem specified as above are assumed to have a default set of *capabilities*: whether or not they support polled retrieval of facsimile documents, what speeds they support for transmitting facsimile, whether or not they handle high resolution facsimile, etc. It is a good idea to specify the correct set of capabilities for a modem when using faxmodem, otherwise you may not get best use of a modem. Not identifying when a modem has limited capabilities can also cause HylaFAX to do extra work or cause errors that might be avoided.

Modem capabilities are specified through faxmodem with the syntax used by Class 2 and Class 2.0 modems. This makes it easy to setup a Class 2/2.0 modem: all you need to do is make a simple query to the modem to get the capabilities string to pass to faxmodem. For example, for a Class 2.0 modem the following commands would be used:

```
hyla% cu -l ttyf2
Connected
at+fclass=2.0
OK
at+fcc=?
(0,1),(0-5),(0-2),(0-2),0,0,0,(0-7)
OK
```

This sets the modem in Class 2.0 and then asks for the set of communication capabilities. The resulting string is then passed to faxmodem:

```
hyla# faxmodem -c '(0,1),(0-5),(0-2),(0-2),0,0,0,(0-7)' ttyf2
```

(note the quote marks around the string so that the shell does not interpret the parentheses).

For a Class 2 modem the commands are slightly different:

```
hyla% cu -l ttyf2
Connected
at+fclass=2
```

```
OK
at+fdcc=?
(0,1),(0-5),(0-2),(0-2),0,0,0,(0-7)
OK
```

For a Class 1 modem an entirely different procedure is needed because the modem only implements a small portion of the facsimile protocol. This means that the capabilities are mostly dependent on the HylaFAX software and not on the modem. The only information needed from the modem is which signalling rates are supported for transmitting fax data; this is obtained with:

```
hyla% cu -l ttyf2
Connected
at+fclass=1
OK
at+ftm=?
24,48,72,73,74,96,97,98
OK
```

and from there a capabilities string can be crafted by understanding that the above list indicates the modem can transmit at speeds from 2400 bps (`24`), 4800 bps (`48`), 7200 bps (`72,73,74`), and 9600 bps (`96,97,98`). (Multiple values for a particular speed indicate support for multiple *modulation schemes*; if any one value is reported then the corresponding speed should be specified in the capabilities string.) Thus the capabilities string is ‘‘(0,1),(0-3),(0-2),(0-2),0,0,0,(0-7)’’ (note the second segment is 0-3 instead of the 0-5 used above which indicated that the modem supported 12200 and 14400 bps signalling rates). Consult the faxmodem(1M) manual page for more information.

When faxq is used in conjunction with faxgetty no modems need to be specified using faxmodem. If modems are specified however; faxq will just treat the modems as ready for use until it receives more up to date information from the faxgetty processes.

Setting up Inbound Service

To setup HylaFAX for inbound facsimile or data service a modem configuration file must be setup and a faxgetty program must be started to listen for input on the tty device. The configuration file setup is usually done at the same time that outbound service is configured; i.e. when faxaddmodem is run. A faxgetty server for the modem should be setup to be run by the init(1M) process according to local system conventions. For System V-based systems this is done by editing the **/etc/inittab** file to spawn faxgetty on the appropriate port. For example, if a modem is to be started on **/dev/ttyf2** the following line might be appropriate:

```
t2:23:respawn:/usr/local/sbin/faxgetty ttyf2
```

For systems that use a BSD-style setup the following line might be appropriate for the **/etc/ttyab** file.

```
cua0    "/usr/local/sbin/faxgetty /dev/cua0"    dialup  on
```

Note that faxgetty may be run on a modem port whether or not it is to provide inbound service. By setting the `RingsBeforeAnswer` configuration parameter to zero, faxgetty will not answer an incoming phone call unless it is explicitly commanded to by the faxanswer(1M) program. This may be desirable if a phone line is used, for example, as the primary line for voice calls.

In general it is desirable to run faxgetty because faxgetty informs the HylaFAX scheduler process

whenever modems are in use and because it identifies modems' capabilities (passing them on to the scheduler so that it can make informed scheduling decisions). faxgetty also includes support for screening calls based on caller-ID information (consult the section ''Caller-ID Support'') and for automatically routing calls based on distinctive ring when these services are provided by the local PTT (consult the section ''Distinctive Ring Support'') . For this additional functionality, and because faxgetty does a reliable job of reseting and configuring recalcitrant modems, it may even be desirable to run faxgetty on non-fax modems.

The following sections discuss HylaFAX support for servicing particular types of inbound calls.

Facsimile Service

In normal operation HylaFAX's faxgetty process will automatically answer inbound phone calls and receive facsimile. Received facsimile are written to files in the **recvq** directory in the spooling area on the server machine. Facsimile data is stored as a TIFF Class F (TIFF/F) file and protected according to the `RecvFileMode` configuration parameter specified in the modem configuration file. The `RecvDataFormat` configuration parameter can be used to control the encoding of data stored in these files; consult the section on ''Transcoding of Received Facsimile'' for more information on this facility. The maximum number of pages that will be received in a single call can also be controlled with the `MaxRecvPages` configuration parameter. Finally, HylaFAX provides an access control list mechanism for restricting recived facsimile according to the TSI string passed as part of the facsimile protocol; consult the section on ''Rejecting Junk Facsimile'' for more information.

Data Service

By default HylaFAX does not enable support for inbound data calls. Data service is not enabled so that naive users do not accidentally setup inbound access to their system before proper password controls are in place. To enable inbound data service the modem configuration file must be setup to accept data calls and to invoke the normal system getty program to process the incoming call. Normally this involves enabling the use of *adaptive-answer* functionality and the setup of the `GettyArgs` parameter in the configuration file.

Adaptive answer is the ability for a modem to determine whether an incoming phone call is for data, fax, or voice use. If a modem supports a good adaptive-answering facility then it should be enabled with the `ModemSetupAACmd` and the faxgetty process will automatically service fax, data, or voice calls *as identified by the modem*. Most Class 2 and Class 2.0 modems provide adaptive-answer support that distinguishes data calls from fax calls and the prototype configuration files that come with HylaFAX automatically enable it if it is provided by the modem. Most Class 1 modems do not provide adaptive-answer support, but HylaFAX provides adaptive-answer support in the server. Consult the section on ''Adaptive Answer Support'' in the next chapter for help on configuring adaptive-answer support.

Setting up the `GettyArgs` parameter requires an understanding of how to automatically startup the system getty program. HylaFAX will invoke the getty program when a data call is recognized and set up

the standard input, output, and error descriptors to point to the appropriate tty device. The getty program should not reopen or reinitialize the modem before doing its work. Some getty programs are incapable of this and are unsuitable for use with HylaFAX. The parameters passed to the getty program must also identify the speed to use to communicate with the local modem. Some getty programs want to automatically detect this rate based on the `CONNECT` message that a modem sends to the host when a connection is established; these programs are unsuitable for use with HylaFAX. A getty program used with HylaFAX must be able to handle a fixed speed for host-modem communication, with the speed specified on the command line.

For System V-style getty programs the appropriate parameters are typically of the form:

```
GettyArgs: "-h %l dx_%s"
```

where the `-h` parameter instructs getty to not hangup the device first, the `%l` parameter is translated to the device name (‘‘the tty line’’), and the `dx_%s` parameter identifies the **/etc/gettydefs** entry to use (`%s` is translated by HylaFAX to the speed used to communicate with the modem). Note that the exact parameters to supply depend on the getty program used; consult local documentation to understand what options should be used.

For BSD-style systems, `GettyArgs` is usually of the form:

```
GettyArgs: "std.%s -"
```

where `std.%s` refers to an entry in the **/etc/gettytab** file for a fixed speed port; e.g.

```
g|std.19200|19200-baud:sp#19200:
h|std.38400|38400-baud:sp#38400:
```

(Note that as before, the ‘‘`%s`’’ is replaced by the speed for host-modem communication.)

Setting up Client Access

HylaFAX client applications such as sendfax do not communicate directly with server processes such as faxq or faxgetty. Instead they communicate with the hfaxd(1M) client-server protocol process. This architecture insulates client applications from the internal structure of a server machine, provides a more robust operating environment, and scales better for many clients.

hfaxd is normally started up when the faxsetup program is run. faxsetup also arranges for hfaxd to be automatically started up each time a server machine is booted; either *standalone* by a script invoked by the init process or *indirectly* by the inetd process. The preferred way to run hfaxd is in a standalone mode as this gives optimal performance.

When hfaxd is started the command line arguments specify which of several client-server protocols it should offer. hfaxd currently has support for three protocols:

- the Version 4.0 HylaFAX client-server protocol,
- the old HylaFAX client-server protocol used in versions prior to 4.0, and

- the Simple Network Pager Protocol (SNPP) that is used to submit alpha-numeric text pager requests.

When operating in a standalone fashion the command line options specify the protocols to support and the ports at which service should be provided. For example, to startup hfaxd in a standalone mode supporting all three protocols the following might be used:

```
# /usr/local/sbin/hfaxd -i 4559 -o 4557 -s 444
```

This specifies that the Version 4.0 protocol is to be offered at port 4559, the old protocol at port 4557, and SNPP at port 444.

It is also possible to have the inetd program startup hfaxd. In this case only a single protocol can be requested since inetd advertises service and establishes the network connection. For example, the following entry might be used in the **inetd.conf** file to startup hfaxd to service SNPP requests:

```
snpp    stream  tcp     nowait  fax     /usr/local/sbin/hfaxd   hfaxd -S -d
```

The `-S` option specifies that hfaxd should service SNPP requests using the standard input and output descriptors and the `-d` option keeps hfaxd from detaching itself from the controlling tty.

It is possible to run hfaxd in a standalone mode as well as indirectly from inetd so long as this is done for separate protocols. Doing this however is of questionable value since it is much more efficient for a single standalone hfaxd process to support multiple protocols than to have multiple unrelated hfaxd processes.

Beware that hfaxd must either be started up by the super-user or be installed setuid-root for proper operation.

Besides arranging for hfaxd to get started up when a server machine is booted, it is necessary to specify which client machines and users can have access to a HylaFAX server machine. This is specified by the contents of the **etc/hosts** file in the HylaFAX spooling area on the server machine. The contents of this file is specified in the hosts(4F) manual page. The default **etc/hosts** file that comes with HylaFAX permits anyone to have access through the *localhost* network interface; i.e. the hosts file contains:

```
localhost
127.0.0.1
```

It is a good idea to refine the controls specified in this file before providing general access to the server. Access can be restricted both on a per-client-machine basis and by user. Passwords can also be required though support for this is presently somewhat awkward.

*The **etc/hosts** file must be owned by the fax user and be mode 0600 or hfaxd will not permit client access.*

Setting up Periodic Maintenance Work

HylaFAX comes with two programs that need to be run periodically on the server machine: faxqclean(1M) and faxcron(1M).

The faxqclean program is responsible for removing old document and job description files from the spooling area on a server. These files are created when outbound jobs are created but are removed in a delayed fashion to permit clients to resubmit jobs without retransmitting all the information to the server and to allow imaged documents to be reused in unrelated facsimile transmissions. faxqclean is also responsible for *archiving* outbound jobs that have completed.

The faxqclean program in version 4.0 does not support job archiving; but consult the manual page to verify this (in case someone has done some local improvements).

faxqclean must be run by the super-user. A sample entry for the cron program that runs faxqclean once each hour might be:

```
0   *    *    *    *    /usr/local/sbin/faxqclean
```

The second program that should be routinely run on each server machine is faxcron. This is a script that does maintenance such as truncating log files and purging old session logs and received facsimile. faxcron needs to be run by the fax user once a day or so and it is a good idea to capture its output since it includes information about failed phone calls that might warrant investigation. A sample cron entry to run faxcron might be:

```
25   23    *    *    *  sh /usr/local/sbin/faxcron | mail FaxMaster
```

Modem Configuration Issues

Beware that when faxgetty processes control a modem they may leave the modem in a state suitable for sending and receiving facsimile. That is, they may leave the modem running in Class 1, Class 2, or Class 2.0. This may have implications for data communication programs such as *tip*, *cu*, and *uucp*. For example, it may be necessary to force the modem into Class 0 (for data communication) when placing a call:

```
AT+FCLASS=0DT<phone number>
```

Exactly how things work depends on the contents of the modem configuration file. It is usually a good idea to setup the configuration parameters so that the modem is left idling in Class 0 (for data use) when HylaFAX is not actively using a modem. Note however that this may not always be possible as some modems require that a modem idle in Class 2 or 2.0 when doing adaptive answer.

Note that when HylaFAX places an outbound facsimile call it automatically forces the modem into Class 1, 2, or 2.0 ***before*** *issuing* `ModemDialCmd`*. Thus the old FlexFAX trick of changing the class in the* `ModemDialCmd` *parameter should not be used.*

System-specific Guidance

This section contains some setup-related issues that are dependent on the operating system installed on the target machine. The information included here is by no means exhaustive; it reflects feedback from users accumulated over multiple HylaFAX versions and/or operating system releases.

IRIX Guidance

On Silicon Graphics Indigo and Indy machines you can not use a Macintosh modem cable to connect your modem to the DIN-8 connector on the back of your host. A Macintosh cable uses a special wiring pattern to pass the RTS and CTS signals between the host and modem. This wiring is not compatible with the wiring used on SGI machines. While it may appear that the the modem and cable work, hardware flow control will not function properly and data will eventually be lost. Consult the serial(7) manual page for an explanation of how to wire up modem cables.

The tty device that is used must reflect whether hardware or software flow control is to be used. Under IRIX, modem devices (i.e. those that monitor DCD) come in two flavors: `ttyf*` devices support RTS/CTS flow control while `ttym*` devices support XON/XOFF flow control. If you want to use hardware flow control to communicate with your modem you should use a `ttyf*` device, otherwise use a `ttym*` device. If you fail to use the correct device you may still get the correct flow control (because later versions of IRIX actually permit flow control to be switched irrespective of the device used), but you are likely to collide with other modem users such as cu, uucp, ppp, and slip that still use the old-style device names (so UUCP lock files may be created for a different name than the one HylaFAX is using).

USR modems work under IRIX 5.x only when patch 475 or a successor is installed and `ClocalAsRoot` *is set to* `Yes` *in the modem configuration file.*

Versions of IRIX prior to 6.2 have a bug in the device driver for the on-board serial ports on several systems that causes RTS/CTS flow control to be turned off as a side effect of setting the `CLOCAL` flag on the associated tty device. Patch 475 (‘‘RTS/CTS flow control busted when `CLOCAL` is set’’) and its successors correct this problem and must be installed to use HylaFAX with RTS/CTS flow control (install the appropriate successor to patch 475). Also, when this patch is installed the `ClocalAsRoot` modem configuration parameter must be set to `Yes` for proper operation (see config(4F) for a detailed explanation of what this parameter does). If you do not have the appropriate patch installed on your system then you will either see flow control-related problems when transmitting facsimile or possibly some other problems related to modems dropping DCD when carrier is lost.

The DPS-based PostScript imager program distributed with HylaFAX is available only in COFF format. Because IRIX 6.2 does not support COFF executables this program cannot be used with HylaFAX. A Ghostscript-based RIP should be used under IRIX 6.2.

The font metric files required by client applications are contained in the **dps_eoe.sw.dpsfonts** image that is part of the standard IRIX distribution.

SCO Guidance

The standard SCO serial I/O driver (SIO) does nothing with modem control lines if `CLOCAL` is set on the tty device. The usual workaround is to use the FAS driver instead.

Solaris Guidance

Versions of Solaris prior to 2.5 require a patch to correct the handling of RTS/CTS flow control with serial ports built around the Zilog ZS8530 chip.

Some versions of Solaris (2.3 is known to do this) silently truncate or discard syslog messages longer than about 120 characters.

Use the **/dev/cua/*** devices and not the **/dev/term/*** devices.

When using *ttymon* to service inbound data calls set the `GettyArgs` parameter to something like the following:

```
GettyArgs: "-g -h -d /dev/cua/a -l 38400 -m ldterm,ttcompat"
```

Be certain you are not running a ttymon with sac when using HylaFAX. Disable all ports that are to be used by HylaFAX with admintool or pmadm(1m).

SunOS Guidance

Versions of SunOS prior to 4.1.4 require a patch to correct the handling of RTS/CTS flow control with serial ports built around the Zilog ZS8530 chip. These patches are available at:

- ftp://sunsolve1.sun.com/pub/patches/100513-05.tar.Z for SunOS 4.1.[123], and
- ftp://sunsolve1.sun.com/pub/patches/101621-04.tar.Z for SunOS 4.1.3_U1.

(choose the one appropriate to the system you are running).

SVR4 Guidance

The following **GettyArgs:** configuration parameter is suitable for many SVR4-based systems:

```
GettyArgs:      "-g -h -t 60 -l ff_%s"
```

Be sure entries for different baud rates are defined in the **/etc/ttydefs** file.

Ultrix Guidance

[*Ed: this information is old.*]

Hardware flow control does not work in HylaFAX Version 3.0, modems must be configured to use software flow control.

Serial ports are conventionally called **tty00**, **tty01**, etc; see MAKEDEV(8) if you need to create devices.

Advanced Server Configuration

This chapter describes how to configure some of the advanced features of HylaFAX. In addition it discusses how many of the modem-specific configuration parameters work together when sending and receiving facsimile. The set of per-modem configuration parameters provides a flexible mechanism for working around many modem and system limitations. This chapter includes the following sections:

- Local Identifier Support
- Dial String Rules
- Tagline Support
- Adaptive Answer Support
- Caller-ID Support
- Distinctive Ring Support
- Copy Quality Checking
- Continuation Cover Pages
- Time-of-Day Usage Scheduling
- Per-destination Controls
- Automatic Truncation of Whitespace
- Modem Classes
- Modem Priorities
- Transcoding of Received Facsimile
- Automatic Processing of Received Facsimile
- Rejecting Junk Facsimile
- UUCP Lock Files
- Modems that Lie about their Capabilities
- Configuration Parameter Usage During Modem Setup
- Modem Parameter Usage for Outbound Calls
- Modem Parameter Usage for Inbound Calls

A separate chapter discusses setup and configuration of the support for transmitting messages to alpha-numeric pager terminals.

Local Identifier Support

The facsimile communication protocols include messages that identify the sending and receiving devices using a 20-character string. The T.30 facsimile protocol specifies that these strings should use only numbers, the ‘‘+’’ symbol, and space. Many fax machines and modem however are capable of accepting a wide range of ASCII characters. By default HylaFAX will use the canonical form of the local phone number for the local identity when sending or receiving facsimile. This phone number is specified with the per-modem `FAXNumber` configuration parameter. If the `LocalIdentifier` parameter is specified, however, HylaFAX will use it instead. Unlike `FAXNumber` which is converted to a canonical form that conforms to the T.30 specification, the value of the `LocalIdentifier` can be any ASCII string; for example,

```
LocalIdentifier: "Errno Consulting"
```

Beware that not all Class 2 and Class 2.0 fax modems support arbitrary ASCII local identifier strings. If you encounter problems verify your modem supports this functionality by sending the following commands (if it is a Class 2 modem):

```
AT+FCLASS=2
OK
AT+FLID=?
(20)(32-127)
OK
```

The response to the `AT+FLID=?` command should indicate the range of ASCII characters the modem will accept in a local identifier string.

Dial String Rules

A dial string specifies how to dial the telephone in order to reach a destination facsimile machine, or other device. This string is supplied by a user with each outbound job. User-supplied dial strings need to be processed in two ways by the HylaFAX server processes: to craft a canonical phone number for use in locating the receiver’s capabilities, and to process into a form suitable for sending to a modem. In addition client applications may need to process a dial string to formulate an external form that does not include private information such as a credit card access code. Phone number canonicalization and dial string preparation are done according to *dial string processing rules* that are located in a file specified in the server configuration file;

```
DialStringRules: etc/dialrules
```

The generation of an externalized form for a dial string is done by rules that *optionally* appear in `DIR_LIBDATA`**/dialrules** on client machines (where `DIR_LIBDATA` is the pathname setup at the time the software is configured for compilation).

Dial string rules are an important part of HylaFAX as they permit local PTT conventions to be handled entirely on the HylaFAX server machine (e.g. the need to dial ‘‘1’’ before certain numbers but not others). Client applications are then free to use a canonical format for phone numbers that is location independent. Dial string rules combined with the `ModemDialCmd` also permit modem-specific idiosyncrasies such as the syntax for doing a ‘‘flash hook’’ to be handled transparently. Finally, dial

string rules can be used to provide dialing *aliases* such as mapping ‘‘a-zA-Z’’ to the corresponding numeric codes.

Consult dialrules(4F) for detailed information.

Tagline Support

HylaFAX includes support for *tag lines*; a line of text imaged at the top of each outgoing facsimile page. Tag lines imaging requires the specification of:

- a format string,
- the name of a file containing an uncompressed Portable Compiled Font (PCF) from the X11 Window System, and
- the host bit order.

When tag lines are enabled the fax server will image a line of text across the top of each outgoing facsimile page using the specified tag line format string and font. This procedure is done *on-the-fly*, just like a fax machine does it. However, because the fax server has lots of information about the outbound facsimile job, the imaged tag line may be setup to display a variety of information.

A 100 dpi 18point Lucida Typewriter font from the X Window System is included in the HylaFAX distribution for use in imaging tag lines. Experiments indicate this is a reasonable font to use. You can however use an alternate font of your liking.

On Silicon Graphics systems a tag line font can be installed by converting one of the compressed font files provided with the standard X server; e.g.

```
% zcat /usr/lib/X11/fonts/lutRS18.pcf.Z > /var/spool/fax/etc/lutRS18.pcf
```

To enable use of the font and imaging of the tag lines setup the `TagLineFont` configuration parameter:

```
TagLineFont: etc/lutRS18.pcf
```

If font files are stored on your system in an uncompressed format then you can reference the file directly using an absolute pathname. Relative pathnames must be specified with respect to the top of the spooling area. If the `TagLineFont` parameter references a non-existent file or a file that does not contain a valid PCF font then tag line support will be disabled and a message will be logged through syslog.

The default tag line format string is ‘‘`From %%n|%c|Page %%p of %%t`’’ which prints three fields containing the phone number of the server, the date and time of the outgoing job, and the page number and total pages for the job. Consult the `TagLineFormat` parameter description in config(4F) for information on configuring a different format string.

The tagtest(1M) program may be used to try out different tag line formats and fonts before they are configured for use by the server.

Finally, be certain the host bit order is configured to reflect the order of bits on your machine. The host

bit order should be properly setup at the time the configure script is run. If the server has the wrong host bit order then tag lines will be imaged incorrectly.

Adaptive Answer Support

Adaptive answer is the ability for a modem to determine whether an incoming phone call is for data, fax, or voice use. If a modem supports a good adaptive-answering facility then it should be enabled with the `ModemSetupAACmd` and HylaFAX will service fax, data, or voice calls *as identified by the modem* and specified in the modem configuration files. For example,

```
ModemSetupAACmd: AT+FAA=1
```

Beware however that for some modems adaptive-answer support works properly only when the modem is left in Class 2 or Class 2.0; this may be a problem if you want to configure the modem to ‘‘idle in Class 0’’ to avoid confusing naive programs that make use of the modem for outbound calls.

The adaptive-answer algorithm used by some modems can confuse some fax machines and/or data modems. If you do not intend to service both data and fax calls you may not want to configure adpative-answer for inbound calls.

If a modem does not support adaptive-answering, then there are several options. If the modem is a Class 1 modem, then HylaFAX provides a simple adaptive-answering strategy whereby incoming calls are first answered as if they are for a fax machine and, if that fails, then answered as if they are for a data modem. This facility is enabled by specifying something like:

```
AdaptiveAnswer:         yes             # enable adaptive answer
AnswerRotary:           "fax data"      # answer for fax, then data
ModemAnswerCmd:         AT+FCLASS=1;A   # default is to answer as fax
ModemAnswerDataCmd:     ATH+FCLASS=0;A  # hangup and answer as data
Class1RecvIdentTimer:   10000           # timeout fax answer in 10 secs
```

in the configuration file. The above lines cause the fax server to do the following in response to an incoming phone call:

1. Issue "`AT+FCLASS=1;A`" to answer the phone call in Class 1; i.e. as a fax machine (issuing CNG tones).
2. Send TSI and DIS frames as required by the fax protocol.
3. Wait for DCS from the caller (if it is a fax machine).
4. Timeout waiting for DCS in 10 seconds (or whatever is specified for `Class1RecvIdentTimer`).
5. Issue "`ATH+FCLASS=0;A`" to hangup and then re-answer the phone in Class 0; i.e. as a data modem.

This technique assumes many things about the capabilities of the modem and the local telephony service and may not work for all Class 1 modems or for all locales.

Note also that by reversing the order of the items specified in the `AnswerRotary` parameter you can get HylaFAX to answer first for data and then for fax. If calls are answered first as data, then you may need to constrain the timeout used to recognize a data call so that time still remains to setup a fax connection.

Consult your modem manual and the `ModemAnswerResponseTimeout` configuration parameter.

A second facility supported by the fax server in lieu of adaptive answering is a *rotary of answering techniques*. The general idea is that a list of alternative ways to answer the phone is supplied and the server will rotate through the list on *consecutive inbound calls* until it finds one that works. For example, one might specify something like:

```
AnswerRotary: "fax data"
```

which would instruct the server to answer incoming calls as if they were from a fax machine until a call was received from a data modem, in which case it would then answer subsequent calls as a data modem until a non-data call was received (in which case it would go back to fax). The rotary list can have up to three items, with items being selected from one of: *fax*, *data*, *voice*, *extern*, and *any* (answer a call of an unknown type). The *extern* answering request forces an external application (the ‘‘egetty’’ program) to be invoked to deduce the type of an inbound call (and possibly handle it). The *voice* answering request causes a ‘‘vgetty’’ program to be invoked to handle the call. Finally, in conjunction with the rotary answer facility there is an `AnswerBias` parameter that can be used to specify an index into the rotary list to use after *successfull* calls. In the above example, this parameter can be used, to force calls to always be answered first as data by specifying:

```
AnswerRotary:   "fax data"
AnswerBias:     1
```

Note that the adaptive-answer and rotary answer facilities differ only in whether the rotary of answering techniques is applied to a single call or to consecutive calls.

Caller-ID Support

Caller-ID is a feature provided by phone companies whereby information about a calling party is passed to the receiver *prior* to answering a call. This facility is not universally available and not all modems are capable of processing the provided signals. For modems that are capable of processing caller-ID information HylaFAX provides the following support.

- Any caller-ID information received by a faxgetty process is recorded in trace logs provided the `ServerTracing` configuration parameter is set to enable the collection of server messages.
- Inbound calls may be accepted or rejected according to an access control list specified by the `QualifyCID` configuration parameter.

Note that HylaFAX parses caller-ID information according to two configuration parameters, `CIDName` and `CIDNumber`, that specify how to identify a caller’s name and phone number (if provided by the modem). For example, for a ZyXEL U1496 modem, the appropriate configuration parameters to use for caller-ID are:

```
CIDNumber: "CALLER NUMBER: " # pattern string for phone number info
CIDName:   "CALLER NAME: "   # pattern string for identity info
```

Consult config(4F) for a full description of the caller-ID-related configuration parameters.

Distinctive Ring Support

Distinctive ring is a feature provided by phone companies whereby multiple phone numbers can be directed to a single phone line with different ring patterns for each phone number. This is a simple mechanism that can be used to distinguish between incoming fax, voice, and data calls (i.e. all fax calls are directed to one number, voice to a different number, and data to a different number).

HylaFAX supports distinctive ring capabilities through the `RingFax`, `RingData`, and `RingVoice` modem configuration parameters. Modems that support distinctive ring send a different `RING` status message to the host depending on the ring pattern presented by the phone company. By setting the configuration parameters faxgetty will match the appropriate `RING` status message and use the information to treat the inbound call as fax, data, or voice *before* answering the call. For example,

```
RingFax:        RING1           # treat ring pattern 1 as fax
RingData:       RING2           # treat ring pattern 2 as data
```

Copy Quality Checking

HylaFAX will automatically check the quality of received facsimile and regenerate rows of data that have errors in them. This facility is termed *copy quality checking* and is only done if the modem is incapable of doing this task itself, or if the software is configured so that modem copy quality checking is disabled.

Copy quality checking is automatically done in the server for Class 1 modems. To configure copy quality checking directly in a Class 2 or Class 2.0 modem--when the modem is capable--setup the `Class2CQCmd` parameter for the modem. For example,

```
Class2CQCmd: "AT+FCQ=1;+FBADMUL=5;+FBADLIN=10"
```

This enables copy quality checking for receiving and sets the copy quality parameters so that acceptable page quality requires that no more than 10 consecutive bad rows of data may be present and less than 5% of the rows in a page have errors in them. To disable copy quality checking in the modem use

```
Class2CQCmd: AT+FCQ=0
```

If a modem indicates that it supports copy quality checking but `Class2CQCmd` has not been specified then HylaFAX will note this when the modem is prepared for use.

Copy quality checking in the server (for any style of modem) is controlled by two configuration parameters: `PercentGoodLines` and `MaxConsecutiveBadLines`; see config(4F) for more information. These parameters define the criteria by which the server decides if a received page has acceptable copy quality. Pages that are deemed unacceptable are rejected and the sender is told to resend the page.

Beware that many fax machines are incapable of resending pages that have unacceptable copy quality because they do not buffer a full page image.

If you encounter problems with the copy quality checking support you can disable it by setting either the `PercentGoodLines` or `MaxConsecutiveBadLines` parameters to 0; e.g.

```
PercentGoodLines:       0       # disable copy quality checking
```

Note however that doing this means you may receive facsimile that are unreadable.

Continuation Cover Pages

A *continuation cover page* is a cover page automatically generated by HylaFAX when retransmitting an outbound job after a protocol error abnormally terminated a previous transmission. This cover page is only generated when a user-submitted cover page has already been transmitted; it is intended to identify the sending system and provide information about why the retransmit is being done. HylaFAX generates the cover page using the parameters supplied when the job was submitted. In addition it provides a *comments* section that identifies the transmission as a continuation of a previous job and describes why the previous attempt to transmit failed (this may be useful to the receiver).

By default HylaFAX does not enable continuation cover pages. To enable use of this feature the `ContCoverPage` configuration parameter must be set to the pathname of a cover sheet template file to use in generating continuation cover pages. This cover page template file is passed to the mkcover(1M) script, or to the program specified with the `ContCoverCmd` configuration parameter; consult the mkcover manual page for a description of this file.

Time-of-Day Usage Scheduling

HylaFAX can be configured so that outbound calls are placed only during certain hours of the day. This can optionally be configured on a per-destination basis, making it easy to reduce phone costs by processing toll calls only during the time when off-peak rates are available.

By default outbound calls are permitted at any time. To constrain calls to a particular time or range of times the `TimeOfDay` parameter can be setup according to the syntax specified in config(4F). This syntax is consistent with the syntax used by the UUCP software. For example, to permit outbound calls only between 9AM and 5PM local time the following might be used:

```
TimeOfDay:      "0900-1700"
```

Per-destination Controls

HylaFAX supports many controls on the operation of the scheduler and related software. For example, outbound jobs may be restricted so that processing is only done at certain times of the day. All of these parameters can be specified in the configuration file used by the central HylaFAX scheduler process. Parameters specified in this way apply to *all outbound jobs*. It is also possible to specify these

parameters on a *per-destination basis* using the `DestControls` configuration parameter. This parameter specifies the pathname of a file that holds per-destination control parameters; destctrls(4F).

Per-destination controls are especially useful for controlling which phone numbers can be called. For example, by specifying an entry of the form:

```
^911$   RejectNotice = "Calls to emergency numbers are not permitted"
```

any outbound job that would cause HylaFAX to phone 911 would automatically be rejected with the submitter notified by electronic mail using the specified message.

Another useful feature of this facility is the ability to delay toll calls to off-hours so that off-peak phone rates may be used. Entries of the form:

```
^[+]1415.*      TimeOfDay = "Any"
^[+]1510.*      TimeOfDay = "Any"
.*              TimeOfDay = "Wk1700-0830,Sat,Sun"
```

might be used to permit calls to US area codes 415 and 510 at any time of day, but force all other calls to be done only during off hours. Note that the parameters specified in the configuration file serve as the *default values* to use when scheduling an outbound job. That is, if a parameter value is not obtained from the destination controls file then it is taken from any parameters specified in the configuration file. For example, if the above rules had not included the last line that matched ‘‘`.*`’’, then the time-of-day to use for long-distance calls would have been that specified in the configuration file (by default `Any`).

Automatic Truncation of Whitespace

HylaFAX can be configured to automatically discard trailing whitespace in outbound facsimile. This truncation is termed *page chopping* and can be setup so that it is done for every page, only the last page of each document, or not all. Page chopping is performed during document preparation and is applied to all submitted documents, no matter what format they are submitted in. By default HylaFAX will only chop the last page of a document and at least 3 inches of trailing whitespace must be present before the page will be truncated. The `PageChop` and `PageChopThreshold` configuration parameters control the page chopping support. Clients can override the defaults setup on the server for each outbound job through command line options to sendfax and client-side configuration parameters.

Modem Classes

Groups of modems can be referenced by name using a facility termed *modem classes*. A modem class defines a logical name for a set of modem devices on a server. This name can be used by clients in the same way that a specific modem device is used. The default modem to use for outbound jobs, ‘‘any’’, is actually just a predefined modem class. Modem classes are useful for groups that want to logically partition a set of modems on a single server.

Modem classes are defined with the `ModemClass` configuration parameter. Each definition specifies a

name and a regular expression that identifies the set of modems that are to be included in the class. For example,

```
ModemClass:     "any:.*"
```

is the builtin definition for ‘‘any’’. Regular expressions are matched against device identifiers for modems that are signalled to faxq by faxmodem or faxgetty.

Modem Priorities

The HylaFAX scheduler process assigns modems to outbound jobs based on three criteria: the modem is considered ready for use, the modem is a member of the requested modem class (if any was specified), and the modem’s capabilities support the requirements of the job (e.g. transmission of high-resolution 2D-encoded data). When multiple modems are present on a server machine it is also possible to force an *order* to the set of modems that are considered for allocation by assigning each modem a *priority*. This can be used, for example, to reduce the likelihood of assigning modems that are heavily used for inbound calls; e.g. near the front of a hunt group.

Modem priorities are specified using the `ModemPriority` configuration parameter. Priority values are integer numbers in the range 0-255 with lower values meaning higher priority.

Modem priorities are passed to the HylaFAX scheduler by the faxgetty processes or with the faxmodem program. Priorities can be dynamically changed by using the faxconfig program to send a new priority to a faxgetty process (who in turn will notify the scheduler), e.g.

```
hyla# faxconfig -m ttyf2 ModemPriority 32
```

or with the `-u` option to the faxmodem program when operating in a send-only environment. The ability to dynamically alter the priority of modems means it is easy to reconfigure modem usage based on time-of-day and traffic patterns.

Transcoding of Received Facsimile

Transcoding refers to the on-the-fly conversion of encoded data. HylaFAX supports transcoding of received facsimile so that received documents can be stored in files using an optimal compression algorithm. If the `RecvDataFormat` configuration parameter is set it defines the format for writing received facsimile data, one of: ‘‘1-D MR’’, ‘‘2-D MR’’, ‘‘2-D MMR’’, or ‘‘adaptive’’. An ‘‘adaptive’’ format is the default; it causes the received data to be written using the data format negotiated by the sender and receiver. Using

```
RecvDataFormat: "2-D MMR"
```

gives the most space-efficient data format. Note that `RecvDataFormat` defines the encoding of the page data, not the file format: this is always TIFF.

Automatic Processing of Received Facsimile

It is desirable to deliver inbound facsimile directly to receipients. HylaFAX routes inbound facsimile through the faxrcvd(1M) shell script. This script may be tailored to use local knowledge and/or tools to route and deliver inbound facsimile. In particular, if facsimile are received from well-known senders and always directed to a specific person faxrcvd can be tailored to automatically dispatch these facsimile by electronic mail; consult the manual page for details. Services such as Direct Inward Dial (DID) are not supported because the hardware uses non-standard programming interfaces. In the future, when updated versions of the ITU facsimile protocols are supported in commercial fax modems HylaFAX will be able to use routing information provided in the protocol to do automatic delivery of inbound facsimile.

Another common request is to automatically print received facsimile as they arrive. This is easy to do by editing the faxrcvd script to use a program such as fax2ps or tiff2ps to convert the TIFF/F file and then submit the resulting PostScript; e.g.

```
$TIFFBIN/fax2ps $1 | lp
```

Note that `TIFFBIN` is known to be the pathname of the directory where the TIFF tools have been installed (see faxsetup and the file **etc/setup.cache** in the spooling area).

Rejecting Junk Facsimile

By default HylaFAX will accept inbound facsimile from any caller. If the `QualifyTSI` configuration parameter is setup then HylaFAX will receive inbound facsimile only if the caller sends an acceptable Transmitting Subscriber Identification (TSI) string; a string that uniquely identifies who the sender is. HylaFAX determines which TSI are acceptable by using the rules given in the file specified with `QualifyTSI`. This mechanism provides a way to automatically reject *junk facsimile*. Note however that because a TSI is not authenticated in any way it is a simple matter for a caller to masquerade as an acceptable sender and bypass the access control mechanism. Consult tsi(4F) for complete details of this facility.

UUCP Lock Files

HylaFAX implements shared inbound/outbound modem usage using UUCP lock files. In normal operation each modem has a faxgetty process listening for data from the modem. When an inbound call is received the modem emits a `RING` status message that wakes up the appropriate faxgetty process. faxgetty then checks the UUCP lock file for the port to make sure the port is not busy for outbound use. If the port is busy (i.e. a UUCP lock file is present), then faxgetty will stop listening on the port and wait for the outbound use to terminate. If the port is not busy, then faxgetty will answer the call according to the parameters specified in the configuration file.

For this scheme to work outbound callers must install a UUCP lock file before using a modem. This lock file must be created using the same conventions understood by HylaFAX. In particular the lock file

name must be consistent and the file contents must include the process ID of the lock file owner written either as a binary or ascii value according to the system-specific conventions.

HylaFAX supports several schemes found on different systems. The appropriate scheme is normally setup according to the target system at the time that HylaFAX is configured for building. It is possible, however, to override these parameters through the runtime configuration files; consult the config(4F) manual page for information about the `UUCPLockType` configuration parameter. In addition to the lock type there are configuration parameters that control the protection mode of the lock file (`UUCPLockMode`), the directory in which lock files are created (`UUCPLockDir`), and a timeout parameter used to purge *stale lock files*. Again, consult config(4F) for a full description of these parameters.

Modems that Lie about their Capabilities

Class 2 and 2.0 fax modems all too frequently have bugs in their firmware. Sometimes it is possible to workaround firmware problems by restricting a modems' capabilities. To do this the `Class2DCCQueryCmd` configuration parameter may be set in the per-modem configuration file. By default this parameter holds the command to send the modem to query its capabilities. However if the query string has a leading ''!'' then no command is sent to the modem and instead the remainder of the string is treated as the response returned by the modem. Thus to avoid using a particular modem capability all you need to do is setup a capabilities string that has a restricted set of modem capabilities.

Capabilities are returned using a syntax specified in the Class 2/2.0 specification:

```
(vr),(br),(wd),(ln),(df),(ec),(bf),(st)
```

where,

- `vr` indicates which vertical resolutions are supported,
- `br` indicates which signaling rates are supported,
- `wd` indicates which page widths are supported,
- `ln` indicates which page lengths are supported,
- `df` indicates which data formats are supported,
- `ec` indicates whether or not the optional Error Correction Mode (ECM) is supported,
- `bf` indicates whether or not the optional Binary File Transfer (BFT) protocol is supported, and
- `st` indicates which minimum scan times are supported.

Each parameter is specified as single value, range of values (e.g. `0-3`), or list of values (e.g. `0,1,2`). The values are given in the Class 2/2.0 specification and in the following table (note that this table does not list values defined in the latest ITU T.class2 specification):

Param	Value	Description	Param	Value	Description
vr	0	98 lines/inch	df	0	1-D Modified Huffman
	1	196 lines/inch		1	2-D Modified Huffman
br	0	2400 bits/sec		2	2-D Uncompressed Mode
	1	4800 bits/sec		3	2-D Modified Modified Read
	2	7200 bits/sec	ec	0	disable ECM
	3	9600 bits/sec		1	enable T.30 Annex A, ECM
	4	12200 bits/sec		2	enable T.30 Annex C, half duplex
	5	14400 bits/sec		3	enable T.30 Annex C, full duplex
wd	0	1728 pixels in 215 mm	bf	0	disable file transfer modes
	1	2048 pixels in 255 mm		1	select BFT, T.434
	2	2432 pixels in 303 mm	st	0	scan time/line: 0 ms/0 ms
	3	1216 pixels in 151 mm		1	scan time/line: 5 ms/5 ms
	4	864 pixels in 107 mm		2	scan time/line: 10 ms/5 ms
ln	0	A4, 297 mm		3	scan time/line: 10 ms/10 ms
	1	B4, 364 mm		4	scan time/line: 20 ms/10 ms
	2	unlimited length		5	scan time/line: 20 ms/20 ms
				6	scan time/line: 40 ms/20 ms
				7	scan time/line: 40 ms/40 ms

An example use of this mechanism is found in the prototype configuration file for the AT&T DataPort modem. The rev C01.66.10 firmware for the DataPort returns an invalid string for the `AT+FDCC=?` query command so the prototype configuration file supplies a valid one instead:

```
Class2DCCQueryCmd:      "!(0,1),(0-5),(0-4),(0-2),(0),(0),(0),(0-7)"
```

Beware of specifying that a modem has capabilities that it does not; HylaFAX is likely to try and make use of them!

Configuration Parameter Usage During Modem Setup

There are many configuration parameters that control the operation of HylaFAX in resetting and initializing a modem. This section presents these parameters and explains how they fit together in normal operation. By understanding how these parameters are used it is possible to control many aspects of the operation of HylaFAX.

In normal operation HylaFAX will first *reset* a modem and then prepare it for use with a setup procedure appropriate for the modem. Modem reset involves the following steps:

1. Lower the Data Terminal Ready (DTR) signal, pause `ModemResetDelay` milliseconds, and then raise DTR. DTR is used to force a modem reset rather than sending a command such as `ATZ` to the modem because many modems can get confused and enter a state where commands sent from the host are ignored. Instead HylaFAX uses a hardware control signal to force the modem to do a reset operation. This usage requires that a modem respond to a DTR transition by resetting itself. Since modems all take different amounts of time to do a reset operation the time that HylaFAX waits for the modem to complete the reset operation is programmable. A certain Hayes Technical bulletin claims one should wait 2.6 seconds for a modem to complete a reset operation. In practice this is much longer than most modems need and this parameter can be reduced with a significant speedup in the overall setup process.
2. Once the modem has been reset (assuming the modem monitors DTR and does the ‘‘right thing’’ in response to DTR being lowered), HylaFAX sets up the baud rate and flow control parameters for the tty device using the `ModemRate` and `ModemFlowControl` parameters.
3. HylaFAX then flushes any data received from the modem since the reset and sends the following sequence of commands:

`ModemResetCmds`	any additional reset commands
`ModemEchoOffCmd`	disable modem echo of commands
`ModemVerboseResultsCmd`	enable verbose result codes
`ModemResultCodesCmd`	enable transmission of result codes
`ModemNoAutoAnswerCmd`	disable modem from auto-answering calls
`ModemOnHookCmd`	place modem ‘‘on hook’’
`ModemPauseTimeCmd`	configure delay for "," in dial string
`ModemWaitTimeCmd`	configure time to wait for carrier
`modemflowcontrolcmd`	establish DTE-DCE flow control scheme
`ModemSetupDTRCmd`	force desired modem response to DTR transition
`ModemSetupDCDCmd`	force desired modem response to DCD transition

   Note that these commands are sent as two separate `AT` commands, broken between `ModemOnHookCmd` and `ModemPauseTimeCmd`; this is done to avoid problems with certain modems.

The reset sequence must be successful for HylaFAX to consider the modem in an acceptable state and to proceed with the subsequent steps:

1. Query the modem using `ModemClassQueryCmd` to verify it supports the functionality specified by the `ModemType` (this parameter can be used to control which class to use when a modem supports multiple classes).
2. Set the modem into the appropriate class using `Class`*`X`*`Cmd`, where *`X`* is one of 0, 1, or 2, depending on the application.
3. Identify the manufacturer, model, and firmware revision information using `ModemMfrQueryCmd`, `ModemModelQueryCmd`, and `ModemRevQueryCmd` parameters. This information is obtained early in the setup procedure in case special-case handling is required for the modem. Note also that for Class 1 modems it is typically not possible to query the modem for some of this information and it must instead be ‘‘hard-coded’’ in the prototype modem configuration file that is selected according to the product ID code returned by the `ATI0` command.
4. Identify the modem capabilities using the `ModemDCCQueryCmd` for Class 2/2.0 modems, or according to the capabilities of the Class 1 driver and the result of `AT+FTM=?` and `AT+FRM=?` commands for Class 1 modems.

5. For Class 2/2.0 modems: establish whether the modem supports copy-quality checking and polled retrieval of documents; both these capabilities are automatically provided by the driver for Class 1 modems. If the modem supports copy quality checking, according to the result of the `Class2CQQueryCmd` and copy-quality setup commands are provided with the `Class2CQCmd` parameter, then the modem is setup accordingly.

Modem initialization is then carried out by sending the following commands to the modem:

- *Class 2/2.0 modems*:

`Class2Cmd`	force the modem into Class 2/2.0
class2FlowControlCmd	establish DTE-DCE flow control scheme
`Class2TBCCmd`	select stream modem host-modem communication
`Class2BORCmd`	set the bit order for HDLC and page data
`Class2PHCTOCmd`	set the timeout during page transfer (Phase C)
`Class2CQCmd`	enable copy quality checking
`Class2NRCmd`	enable negotiation reporting (*Class 2.0 only*)
`Class2PIECmd`	disable program interrupt handling (*Class 2.0 only*)
`Class2BUGCmd`	enable HDLC frame reporting if requested
`Class2DCCCmd`	initialize modem capabilities
`Class2RELCmd`+	enable reception of byte-aligned EOL codes
`Class2CRCmd`+	enable facsimile reception
`Class2LIDCmd`+	set local station identifier
`ModemSetupAACmd`+	enable adaptive-answer support

- *Class 1 modems*:

Class1Cmd	force the modem into Class 1
class1FlowControlCmd	establish DTE-DCE flow control scheme
`ModemSetupAACmd`	enable adaptive-answer support

(*Commands marked with a ''+'' are optional and sent to the modem only if it is to be setup to receive calls*.)

Flow control is explicitly set separately for Class 0 (data) operation and Class 1, 2, 2.0 (fax) operation. To configure the use of a different flow control scheme in each class simply override the default definition of the `Class`*X*`Cmd` parameter; e.g.

```
Class2Cmd: "AT+FCLASS=2<xon>"
```

Note however that this will only work for outbound usage; doing something similar for inbound processing is more complicated because the modem, rather than the host, decides when to switch between Class 0 and Class 1, 2, 2.0. Consequently HylaFAX must *react to events rather than initiate them*. This work is also complicated by the fact that many modems do not accept commands (or are confused by commands) from the host during certain critical sections of inbound call processing.

Modem Parameter Usage for Outbound Calls

An outbound facsimile job is broken up into *phases* that correspond roughly to the phases defined in the

CCITT T.30 facsimile protocol:

- call placement
- selection of station session capabilities
- per-page capabilities negotiation
- page transfer
- page acknowledgement

The last three phases may be repeated multiple times depending on the number and characteristics of the facsimile being transmitted.

To send a facsimile a phone call must be placed. HylaFAX takes the following actions:

1. Lock the modem.
2. Initialize the modem for outbound facsimile use.
3. Instruct the modem to place the phone call.

The corresponding set of commands sent to the modem are:

- *Class 2/2.0 modems*:

`Class2Cmd`	force the modem into Class 2/2.0
class2FlowControlCmd	establish DTE-DCE flow control scheme
`Class2TBCCmd`	select stream modem host-modem communication
`Class2BORCmd`	set the bit order for HDLC and page data
`Class2PHCTOCmd`	set the timeout during page transfer (Phase C)
`Class2CQCmd`	enable copy quality checking
`Class2NRCmd`	enable negotiation reporting (*Class 2.0 only*)
`Class2PIECmd`	disable program interrupt handling (*Class 2.0 only*)
`Class2BUGCmd`	enable HDLC frame reporting if requested
`Class2DCCCmd`	initialize modem capabilities
`Class2LIDCmd`	set local station identifier
`Class2SPLCmd`	request polling status (if a polled receive is to be done)
`Class2DDISCmd`	setup initial session parameters (optional)
`ModemDialCmd`	place the call

- *Class 1 modems*:

`Class1Cmd`	force the modem into Class 1
class1FlowControlCmd	establish DTE-DCE flow control scheme
`ModemDialCmd`	place the call

Note that the modem is always switched into the appropriate class and flow control scheme before placing the call. For Class 2 modems that do not properly implement the `AT+FDIS` command the `Class2DDISCmd` parameter must be configured so that HylaFAX will setup initial session parameters *before* placing the phone call.

If the call completes and a facsimile transmit session is started then HylaFAX will send `ModemSendBeginCmd` to the modem. This parameter can be used to enable servicing that must be done only while Carrier Detect (CD) is asserted to the host. For example, on Sun systems it is not possible to

enable RTS/CTS flow control on serial ports implemented with the ZS8530 unless CD is asserted; to workaround this problem for outbound calls one can use:

```
ModemSendBeginCmd:      "<rts>"
```

to force HylaFAX to enable RTS/CTS flow control on the host tty device after carrier is established. Beware however that some modems raise and lower CD during Class 1 operation and this can cause problems for systems with restrictions of this sort.

The next phase in the processing of an outbound job is the selection of station capabilities. The receiver is required to transmit a series of messages that provide its identity and capabilities (e.g. whether it can accept 2D-encoded facsimile data). HylaFAX examines the received capabilities and compares them to the capabilities required for the job. If the receiver is capable of accepting the documents to be sent then HylaFAX proceeds with the transmission. Otherwise HylaFAX will disconnect and, either re-prepare the outbound documents according to the receiver's capabilities or reject the job because it cannot be sent.

Page transfer requires the selection of session capabilities that correspond to the page to be transferred and the actual transfer of the page data to the modem. HylaFAX handles documents with varying page characteristics. In particular this means that documents with a mixture of high res and low res data (e.g. a low res cover page followed by high res image or text) are handled transparently.

For Class 2 modems session parameters are set using the `Class2DISCmd` (note that this is not the same as `Class2DDISCmd`). If the negotiated session parameter needs to be changed then this command will be sent to the modem. The Class 2 specification requires that a modem accept this command at any time prior to an `AT+FDT` command. However because some modems do not properly implement this part of the specification the `Class2DDISCmd` is provided to workaround modem limitations. Beware however that modems that require `Class2DDISCmd` may not be capable of renegotiating session parameters at all; this can be an annoying problem.

For Class 2/2.0 modems, page data is transferred using the `AT+FDT` command. This command instructs the modem that a page of facsimile data will follow. The modem is released to negotiate session parameters (as needed) and HylaFAX then waits for a response that indicates page data may be transferred to the modem. Class 2 modems indicate they are ready for page data by sending `CONNECT` followed by XON (DC1) to the host. (The `ModemWaitForXON` configuration parameter controls whether or not HylaFAX should wait for XON because some Class 2 modems do not follow this aspect of the spec.) Class 2.0 modems indicate they are ready for page data by sending a `CONNECT` response to the host (but no XON). Class 2 modems return `OK` to the host when they have accepted all the page data. Class 2.0 modems do not return `OK`.

After a page of data has been transferred to the modem HylaFAX indicates whether this is the last page to be sent and/or whether more documents are to come. A *post-page message* is then sent and a post-page response is awaited. Class 2/2.0 modems handle this exchange entirely within the modem. For Class 1 modems HylaFAX implements this part of the protocol on the host: the post-page message may be retransmitted as many as three times. A post-page response indicates whether a page was successfully received and/or whether or not the receiver wants to resynchronize the line due to a perceived loss in signal quality. HylaFAX is capable of retransmitting pages that a receiver deems unacceptable. HylaFAX will automatically retransmit a page up to three times if it does not receive a positive acknowledgement from the receiver.

Modem Parameter Usage for Inbound Calls

Inbound call handling is the most complicated part of the HylaFAX software. This is because modems vary so widely in the way they process inbound calls for fax, data, and voice use. Also, inbound calls typically require that a modem be configured *before* a call is received and many modems do not accept commands from the host until after a call has been recognized, accepted, and processing initiated.

HylaFAX handles inbound calls in the faxgetty program. The modem is initialized as described above and faxgetty then waits for a ring status message from the modem. On receiving input from the modem the following processing takes place:

1. The modem is locked. If the modem cannot be locked because there is an outbound user in control of the modem then it is released and faxgetty will wait for the lock file to be removed.
2. Wait for `RingsBeforeAnswer` rings, waiting at most 5 seconds for each ring status message. If distinctive ring support is configured (`RingFax`, `RingData`, and `RingVoice`) then the ring status messages are checked and the type of call is potentially deduced. If Caller-ID support is configured (`CIDName` and `CIDNumber`), then any Caller-ID information returned by the modem is parsed.
3. If the appropriate number of rings was not received, then the line is hung up and the modem is reset and initialized.
4. Check any Caller-ID information. If `QualifyCID` is configured and the caller is not acceptable, then reset and initialize the modem.
5. If the call type is already known from distinctive ring information, then the call is processed immediately. If faxgetty was instructed to answer any type of call, then the call is answered as appropriate. However, if a particular type of call was to be processed (e.g. data) and the deduced type of call does not match, then the call is rejected and the modem is reset and initialized.
6. If the call type is unknown then the call is answered according to the the current type specified by `AnswerRotary` and `AnswerRotor`.
7. If the call is not successfully answered and if `AdaptiveAnswer` is enabled, the next item in the `AnswerRotary` list is used to immediately re-answer the call. This is repeated until a call is successfully answered or the list of choices in `AnswerRotary` is exhausted.

Call answering is done as follows:

1. Send the modem one of `ModemAnswerDataCmd`, `ModemAnswerFaxCmd`, and `ModemAnswerVoiceCmd` according to whether the call is to be answered as data, fax, or voice, respectively. If any of these are not set then `ModemAnswerCmd` is used instead.
2. Wait for a response from the modem that signifies carrier has been established and which identifies the type of call. The exact set of responses is given in config(4F).
3. If `ModemWaitForConnect` is enabled, then HylaFAX will read responses from the modem until a `CONNECT` response is sent to the host or an error is detected; e.g.

```
Host       Modem
ATA   -->
      <--  DATA
      <--  CONNECT
```

4. If a call is successfully answered, then one of `ModemAnswerDataBeginCmd`, `ModemAnswerFaxBeginCmd`, and `ModemAnswerVoiceBeginCmd`, is sent to the modem according to the call type deduced above. This mechanism is useful for modems that automatically change DTE-DCE communication to a fixed line speed or flow control setting for inbound facsimile calls.

At this point the call has been answered, carrier established, and the call type is known.

- Data calls are handled by invoking a system getty program; but only if the `GettyArgs` parameter is set.
- Voice calls are handled by invoking a system vgetty program; but only if the `VGettyArgs` parameter is set.
- Facsimile calls are handled directly in faxgetty.

For a data or voice call faxgetty forks and execs the appropriate program. The UUCP lock file is setup so that it is owned by the child (important for SLIP and PPP) and the parent handles cleanup of various resources when the child process terminates.

For fax operation faxgetty drops into the facsimile receive protocol.

More to be added.

Alpha-numeric Pager Support

This chapter describes how to configure the support for transmitting messages to alpha-numberic pager devices using the IXO/TAP protocol. This facility is implemented with the sendpage(1) client program, the hfaxd(1M) server program that implements the Simple Network Pager Protocol (SNPP), and the pagesend(1M) server program that does the actual delivery.

Alpha-numeric pages are transmitted by contacting a server machine that speaks SNPP and requesting that one or more messages be delivered to one or more paging terminals. HylaFAX implements SNPP within the framework of the hfaxd client-server protocol process (SNPP is one of several protocols it supports). Pager messages submitted through hfaxd result in jobs being submitted to the HylaFAX scheduler process and these jobs are in turn handed to the pagesend program for delivery by placing a phone call to a pager service provider and communicating the requests using the IXO/TAP protocol.

Note that by using SNPP to communicate between client and server, client applications other than sendpage can be used to submit pager messages. There supposedly are several client applications for Mac and Windows-based systems that use SNPP to submit pages and a growing number of pager service providers are providing on-line SNPP services via the Internet.

Setting up SNPP

sendpage normally tries to contact an hfaxd server process at port 444 on a HylaFAX server machine.

To enable this service hfaxd must be started with a `-s` option when run standalone or with a `-S` option when started by the inetd process. For example, as a standalone server hfaxd might be started by:

```
hyla# /usr/local/sbin/hfaxd -i hylafax -o 4557 -s 444
```

to enable service for the HylaFAX client-server protocol (at the port associated with the symbolic name ‘‘hylafax’’), the old client-server protocol at the default port (4557), and SNPP at the official port (444).

One of the tasks that hfaxd does in implementing SNPP is to map client-specified Pager Identifier Numbers (PINs) to a dialstring to use in contacting the service provider. This work is done using rules specified in a pagermap(4F) file. For example, the following pagermap file would accept only PINs with a leading ‘‘Sky’’ followed by a number and pass the number through as the PIN:

```
^Sky([0-9]+)$   1800SkyTel/\1
.*              reject
```

Note that this mapping functionality can also be used to define aliases; e.g.

```
^sam$           1800SkyTel/1234567
```

The default pagermap filename is **etc/pagermap** (relative to the root of the HylaFAX spooling area).

If hfaxd does not have a pagermap file then it will reject all requests to submit pager messages.

Each page request received by hfaxd causes a job to be submitted to the HylaFAX scheduler process. The scheduling parameters for these jobs are defined by the *service level* specified with the SNPP `LEVE` request (typically through the `-l` option to sendpage). hfaxd uses the service level to set a job’s scheduling priority, expiration time, and the time to delay in between retrying calls to the service provider based on three *maps* that can be specified with hfaxd configuration parameters. The parameters are: `PriorityMap`, `KillTimeMap`, and `RetryTimeMap`, respectively. Consult the hfaxd(1M) manual page for complete information on these parameters.

Setting up Pagesend

There are several configuration parameters associated with the IXO/TAP support. The most important parameter is `PagerSetupCmds` which is a set of commands to send to the modem to configure it for a call to the pager service provider. This command should usually be defined to constrain the modem to connect at 1200 baud using V.22 (and no error correction protocol). Other configuration parameters exist to control esoteric aspects of the IXO protocol implementation and should not need to be changed; consult config(4F) for full details.

Besides the per-modem configuration parameters described above, the only other pager-related information that may need to be setup is information in the info(4F) database to constrain the maximum length of an alpha-numeric pager message, any password string that must be sent to the service provider during the initial login sequence, and any non-standard parity setting to use in communication. For example,

```
&pagerMaxMsgLength:     112
```

```
&pagerPassword:         "foobar"
&pageTTYParity:         none
```

The `pageTTYParity` setting is not needed if the service provider correctly implements the IXO/TAP protocol. Some providers however do not use the standard 7-bits of data with even parity that is specified and it is necessary to specify either ‘‘`none`’’ for 8-bits of data and no parity or ‘‘`odd`’’ for 7-bits of data and odd parity..

Troubleshooting

Each call placed by *pagesend* generates a session log similar to the logs created when transmitting facsimile. Tracing bit `0x0002` enables logging of the IXO protocol operation and `0x1000` enables more low-level tracing messages that display the binary data associated with the protocol messages (the latter is needed only when debugging subtle problems). Both bits should be set in the `SessionTracing` parameter for a modem.

The most common problems are not setting up the `PagerSetupCmds` properly for the modem and service provider and service providers that do not implement the IXO protocol according to the specification; usually by using odd instead of even parity.

CLIENT SETUP

HylaFAX is based on a client-server architecture. A single server machine with one or more modems may service a network of client machines that do not have modems. Client and server machines typically communicate using a special HylaFAX client-server protocol that is transported on top of the TCP/IP protocols. Alternatively, clients can be setup to submit facsimile and pager requests using other transport mechanisms such as electronic mail or a network printing service. This chapter discusses only the setup of a UNIX system as a client machine that uses the standard HylaFAX client applications to submit and manage outbound jobs. Building a mail-to-fax gateway is discussed in a separate ‘‘Mail to Fax Gateway’’ chapter. You are on your own if you want to use a printer protocol to hook clients up; though it is not much more complicated than the mail route (you should look at how the mail support is done).

From the Source Distribution

From the source distribution configure and build the software as described in the chapter ‘‘Building HylaFAX From Source Code’’ (NB: this will build both client and server software). To install only the programs and files required on a client then do

```
hyla# make installClient
```

The set of files required for a HylaFAX client is also given in the hylafax(1) manual page (see the

FILES section).

Setting Up Defaults

With the applications and data files available on a client machine, the only other work one might do on the client machine is to configure default settings for the HylaFAX client applications. The most important of these settings is the identity of the machine to contact for service. This can be specified in many ways. HylaFAX client applications read configuration information from a system-wide file named **hyla.conf** followed by a per-user configuration file named **.hylarc** that is stored in the user's home directory. The `Host`, `Port`, `Protocol`, and `Modem` configuration parameters can be setup to completely specify a HylaFAX server and modem to use as a default. Otherwise the `FAXSERVER` and `SNPPSERVER` environment variables can be used to override this information (the latter is used only by the sendpage program). If no configuration values are defined on a client machine then HylaFAX uses values that are compiled into each program: service is requested from the `localhost` at the port associated with the `hylafax` service or at port `4559` if this service is not defined in the system services database.

Consult the manual page for each client application for a full list of the configuration parameters that are used and the set of configuration files that are searched.

Setting up Server Access

Once a client machine is setup for use the server machine may need to be configured to permit client access. Specifically, the file **etc/hosts** in the spooling area on the server machine must be setup to permit client access to HylaFAX services. This documentation has a section on ‘‘Setting up client access’’ and the hosts(4F) manual page has reference information how this is done.

Customizing Cover Pages

The fax submission program sendfax(1) includes support for the automatic generation of cover pages for each outbound job. Cover pages are generated by invoking a program, by default faxcover(1), to create a PostScript cover page for a transmission. Users can customize cover page generation in many ways. The default faxcover program supports customization through the user of *template files* that are parameterized with PostScript. Alternatively the faxcover program itself can be replaced by overriding its definition in a configuration file (either a personal file or a system-wide file). Finally clients may simply suppress the automatic cover page support provided by sendfax and supply pre-constructed cover pages of their own choice as normal documents that are to be transmitted. The only negative aspect of the last alternative is that cover pages submitted in this manner may confuse the continuation cover page support provided a the HylaFAX server.

Customizing Document Conversions

A HylaFAX server accepts PostScript or TIFF format files and converts them to the appropriate format required to transmit them as facsimile. Any other documents must be converted prior to submitting them to the server for transmission. In particular ASCII text must be converted before it can be submitted. The fax submission program sendfax(1) uses a set of *file type conversion rules* to decide how to prepare documents for transmission as facsimile and HylaFAX includes several document conversion programs that are used on client machines to prepare documents for transmission: textfmt(1) to formats text and convert it to PostScript and, for IRIX systems, sgi2fax(1) to convert SGI RGB-format images to TIFF. Users can customize the file type conversion rules as needed by overriding the default set of conversion rules. To do so just create a **.hylarc** file that specifies a private set of file type conversion rules:

```
TypeRules: /usr/people/sam/.typerules
```

then copy the default set of rules and edit it as needed. For example the default rules do not include support for preparing WordPerfect documents for transmission. However there is a user-contributed *wpr* script that can be used together with the following typerules:

```
0       byte    0xff    error   WordPerfect document
>1      string  WPC     ps      /usr/local/bin/wpr %i >%o
```

Consult the typerules(4F) manual page for details.

HylaFAX Server Operation

In normal operation HylaFAX server machines run four independent programs:

- faxq, the central scheduler process
- faxgetty, the ‘‘front door’’ program that handles incoming calls,
- hfaxd, the program that handles the client-server protocol for submitting jobs and retrieving status information, and
- faxqclean, the spooling area cleaner process.

The faxq scheduler process is normally started once when the system is booted. If faxgetty processes are to be used, they should be setup so that the init(1M). program will start them on each port where a fax modem resides. The hfaxd program can be run either as a standalone server process or started by the inetd(1M) process; the intent is that it run standalone. faxqclean handles the delayed purging and archiving of outbound jobs and is intended to be run from cron(1) regularly (how regularly is a decision for the system administrator).

Under a System V-based operating system the normal installation procedure sets up the system so the faxq and hfaxd processes are started up by init(1M). On other systems you will need to arrange for this yourself (typically by editing **/etc/rc.local** or similar). faxgetty processes must be manually setup by editing the appropriate control file for init; usually either **/etc/inittab** or **/etc/ttytab**. hfaxd can be setup to be started by inetd when the faxsetup script is run to configure a system for use with HylaFAX.

If you need to start the faxq server by hand, consult the faxq(1M) manual page and the shell script **etc/hylafax** that is used on System V-based systems.

Incoming facsimile are placed in the **recvq** subdirectory of the spooling area and probably will need to be cleaned up periodically. Likewise there is logging information in the **log** subdirectory and accounting information in the **etc** subdirectory of the spooling area that may need some attention.

The faxcron script is designed to handle the periodic maintenance of the spooling area. This script is designed to be run from cron(1) every day.

If you want to do accounting check out the xferstats and recvstats scripts for a basic attack on how to process the **etc/xferlog** accounting file that has records of all facsimile transmissions and receptions. Note that faxcron uses these scripts to deliver statistics about facsimile that were recently transmitted and received.

Otherwise the only matter to be concerned with is the support for data connections. If your modems are capable of differentiating data connections from facsimile connections the fax server can be configured to startup a getty program whenever a call from a data modem is received. Alternatively, if your modem does not support an adaptive-answer facility, but it is a Class 1 modem, the server may be able to do adaptive answering in software. In any event, beware that if you enable data connections you should take the normal precautions you would take when there are dialup ports on your machine. Specifically, make sure that you have passwords, appropriate file protections, and proper configuration of *uucp* or similar.

If you encounter problems sending or receiving facsimile you can enable copious tracing information by editing the HylaFAX configuration files. Consult the Troubleshooting section and the config(4F) manual page, in particular, for help. There is also information in the HylaFAQ on problems one might encounter during server operation.

Controlling Modem Usage

HylaFAX provides a great of flexibility in how modems on a server machine are used. Modems can be constrained for outbound-only use, inbound-only use, or both. Modem priorities (see the section in the ‘‘Advanced Server Setup’’ chapter) can be used to control the order in which modems are assigned to outbound jobs. Modems can also be dynamically *taken off-line* for either inbound or outbound use by using the faxstate or faxconfig commands. For example, to mark a modem as ‘‘busy’’ or ‘‘down’’ so that it is not assigned to an outbound job the faxstate command would be used:

```
hyla# faxstate -s busy ttyf2
```

To temporarily disable the answering of inbound calls on a modem one could use faxconfig:

```
hyla# faxconfig -m ttyf2 RingsBeforeAnswer 0
```

Similarly, it is easy to constrain a modem so that only inbound data calls are accepted during part of the day with either data or facsimile calls the rest by using simple shell scripts that are invoked by cron. The most flexibility is available when using faxgetty to control modems.

Mail to Fax Gateway

It is easy to setup a simple mail to fax gateway facility with the tools included in this distribution and some simple additions to your mail delivery agent configuration.

1. Setup the HylaFAX software as usual.
2. If your system uses sendmail to deliver mail, then follow the instructions in **faxmail/mailfax.sh-sendmail**. (Thanks to Eric Allman for the sendmail configuration hack.)
3. If your system uses smail (e.g. Linux users), then follow the instructions in **faxmail/mailfax.sh-smail**.
4. Restart your mail software, refreeze your configuration or whatever is necessary to cause the configuration changes to be seen by the system.

Voila! Now mail to *user@dest.fax* will get formatted and submitted as a facsimile job to *user* at the specified *dest*.

Mail submitted through this gateway can include header lines in the envelope that control the various aspects of the transmission. For example, to have the facsimile sent at 196 lines/inch the header line:

```
X-FAX-Resolution: 196
```

would be used. Headers exist not only to control aspects of the transmission but also the formatting work that is done in preparing the mail text for transmission; consult the faxmail(1) manual page for complete information.

The normal access control mechansims on submitting a facsimile for transmission are enforced; you may need to use them if you setup a fax gateway on your system!

Dirk Husemann has contributed a more elaborate system; the README file is accessible in the user contributed software area on the master FTP server.

Troubleshooting HylaFAX problems

This section contains tips for dealing with the most common problems encountered when setting up and running the software. If you did not follow the instructions in the chapter on ‘‘Server Setup and Basic Configuration’’ then do not bother reading this chapter; read the setup information first!

There are several components to the complete HylaFAX software package:

- client applications (sendfax, (sendpage, faxstat, faxrm, etc.),
- the job submission and server status process that runs on the server machine and communicates

with client applictions (hfaxd),
- the programs that handle delivery of outbound facsimile and pager requests (faxsend and pagesend),
- the processes that service each modem (faxgetty),
- the central scheduler for outbound jobs (faxq), and
- the spooling area cleaner process (faxqclean).

If you are having trouble first try to identify which part of the system is failing. Work forward from the client application to the server machine. On the server machine work from the hfaxd process to the scheduler to the delivery programs. Usually it is pretty obvious which piece of the system has got a problem but if you are unfamiliar with the software you can easily be fooled by error messages that may be passed back to client programs from a process deep within a server machine. The following sections cover specific areas:

- Client Basics
- Client Access Control Problems
- Server Basics
- Scheduler Operation
- Session Tracing
- PostScript Document Preparation
- TIFF Document Preparation
- Communication Problems
- Getty Problems
- Problems with UUCP, cu, tip, etc.

You can also consult the HylaFAQ for answers to common questions.

TROUBLESHOOTING: CLIENT BASICS

All client applications support a `-v` option to enable various levels of debugging. It is possible with one or more `-v` options to trace the protocol between the application and the hfaxd process on the server machine. hfaxd has a `ServerTracing` configuration parameter that controls various tracing support, including the protocol messages it receives. If you are in doubt whether a problem is on the client machine or the server, try the following:

Run faxstat to request server status. You should see something like:

```
HylaFAX scheduler on hyla.chez.sgi.com: Running
Modem ttyf2 (+1 510 528-9999): Running and idle
```

or possibly,

```
HylaFAX scheduler on hyla.chez.sgi.com: Not running.
```

If you do not see something like this, then you are having problems communicating with the hfaxd program on the server machine or there is a configuration problem on the server machine. If you cannot establish a connection to the hfaxd process on the server machine, then verify that you have your `FAXSERVER` environment variable setup correctly (if the server is not on the same machine where the

faxstat program is run) and that both client and server programs are communicating on the same TCP port.

Beware of settings that might be present in personal or system-wide configuration files. Some applications such as sendfax search an additional file as well. Consult manual pages for complete information about configuration file handling.

On the server machine make sure that the hfaxd program is setup to run standalone or properly configured to be invoked by the inetd program. If run standalone then hfaxd should be running and have been started with a `-i` option (and possibly other options). hfaxd should also send messages to the system logging facility whenever it is started up and these messages should identify the client-server protocols it is servicing; e.g.

```
May 28 20:11:20 5D:hyla HylaFAX[8447]: HylaFAX INET Protocol Server: restarted.
May 28 20:11:20 5D:hyla HylaFAX[8447]: HylaFAX Old Protocol Server: restarted.
May 28 20:11:20 5D:hyla HylaFAX[8447]: HylaFAX SNPP Protocol Server: restarted.
```

Otherwise check the contents of **/etc/inetd.conf**, or similar, for a line of the form:

```
hylafax stream  tcp     nowait  fax     /usr/local/sbin/hfaxd   hfaxd -I
```

There may also be other entries if support for the old client-server protocol and/or SNPP is enabled.

Be certain that inetd has ‘‘reread’’ its configuration file; either send it a SIGHUP or restart it. This should automatically happen when the faxsetup program is run on the server machine.

*Note also that the `fax` service must be defined on the server machine in order for inetd to startup the hfaxd program--check for this entry in the **/etc/services** file and/or the YP/NIS database.*

Note that hfaxd uses the chroot system call to confine clients to the HylaFAX spooling area on the server machine. On most systems only the super-user is permitted to do a chroot call so if hfaxd is not started by the super-user or the executable program is not setup to be setuid-root then it will not function properly. If this happens clients will usually be denied access with a message of the form ‘‘`Cannot set privileges.`’’.

You can also use an existing network program such as telnet or ftp to communicate with the hfaxd process;

```
hyla% telnet localhost 4559
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 hyla.chez.sgi.com server (HylaFAX (tm) Version 4.0beta012) ready.
help
214-The following commands are recognized (* =>'s unimplemented).
   ABOR    CHMOD   IDLE    JSUBM   JGRP*   NLST    REIN    STAT    USER
   ACCT*   CHOWN   JDELE   JSUSP   JGSUBM* NOOP    REST    STOR    VRFY
   ADMIN   DELE    JKILL   JWAIT   JGSUSP* PASS    RETR    STOT
   ALLO*   DISABLE JNEW    JGDELE* JGWAIT* PASV    RNFR*   STOU
   ANSWER  ENABLE  JOB     JGKILL* LIST    PORT    RNTO*   STRU
   APPE    HELP    JOBFMT  JGNEW   MDTM    PWD     SHUT    SYST
   CWD     FILEFMT JPARM   JGPARM* MODE    QUIT    SITE    TZONE
   CDUP    FORM    JREST   JGREST* MDMFMT  RCVFMT  SIZE    TYPE
214 Direct comments to FaxMaster@hyla.chez.sgi.com.
quit
221 Goodbye.
```

```
Connection closed by foreign host.
```

If the network-related configuration is setup properly but faxstat still does not return an expected answer then use the `-v` option to trace the client-server protocol:

```
hyla% faxstat -v
Trying localhost (127.0.0.1) at port 4559...
Connected to localhost.
220 hyla.chez.sgi.com server (HylaFAX (tm) Version 4.0beta012) ready.
-> USER sam
230 User sam logged in.
-> TZONE LOCAL
200 Using time values in PST.
-> PORT 127,0,0,1,7,198
200 PORT command successful.
-> LIST status
150 Opening new data connection for "status".
HylaFAX scheduler on hyla.chez.sgi.com: Running
Modem ttyf2 (+1 510 528-9999): Running and idle
226 Transfer complete.
```

A protocol or configuration problem should be evident from the trace information. You can also configure hfaxd to log its operation through syslog by setting the `ServerTracing` configuration parameter in the **hfaxd.conf** file:

```
ServerTracing: 0x3              # enable protocol tracing
```

hfaxd will reread this file for each new network connection so there is no need to restart the server if running standalone.

Once again, remember that inetd will not see a change to the **inetd.conf** file until it is restarted or sent a SIGHUP; and that hfaxd logs its debugging information through syslog (using the syslog facility setup in the **hfaxd.conf** file, or the setting compiled into the program when the software was built).

TROUBLESHOOTING: CLIENT ACCESS CONTROL PROBLEMS

If you are able to establish a network connection to the hfaxd server process then access control problems are either due to incorrect installation of the server software or misconfigured permissions on the server machine. Client access is defined by the contents of the **etc/hosts** file located in the HylaFAX spooling area on the server machine. This file must exist and *must not be publicly readable* or access will be denied to all clients.

*Beware of the protection on the **etc/hosts** file when upgrading from HylaFAX 3.0. Previous versions of HylaFAX did not care if the file was publicly readable so it probably is.*

The hfaxd program confines clients to the spooling area on the server machine using the chroot(2) system call. This means that clients should not have access to any information on a HylaFAX server machine outside the spooling area (and hfaxd restricts access to only part of this area).

HylaFAX clients are assigned a ‘‘fax user ID’’ when they login to a HylaFAX server. This identifier is similar to the normal UNIX user ID but is maintained separately by HylaFAX and is independent of the normal system operation. Beware that hfaxd stores this ID as the group ID of files that are created on a HylaFAX server on behalf of a client. The fax user ID assigned to a client is defined by the information

in the **etc/hosts** file; consult the manual page for more details. You can also find the user ID for a particular user by logging in with telnet and issuing a `STAT` request:

```
melange% telnet flake 4559
Trying 192.111.25.39...
Connected to flake.esd.sgi.com.
Escape character is '^]'.
220 flake.esd.sgi.com server (HylaFAX (tm) Version 4.0beta020) ready.
user foo
230 User foo logged in.
stat
211-flake.esd.sgi.com HylaFAX server status:
    HylaFAX (tm) Version 4.0beta020
    Connected to melange.esd.sgi.com (192.111.25.40)
-->  Logged in as user foo (uid 2)
    "/" is the current directory
    Current job: (default)
    Time values are handled in GMT
    Idle timeout set to 900 seconds
    Using long replies
    No server down time currently scheduled
    HylaFAX scheduler reached at /FIFO (not connected)
    Server FIFO is /client/25133 (open)
    File cache: 15 lookups, 0 hits (0.0%), 1.1 avg probes
        15 entries (2.3 KB), 0 entries displaced, 0 entries flushed
    TYPE: ASCII; STRU: File; MODE: Stream; FORM: PS
    No client data connection
211 End of status
quit
221 Goodbye.
Connection closed by foreign host.
```

Client access may be controlled with passwords that are transmitted in the clear by a client across the network connection. This is obviously insecure and plans exist for improved security measures. The existing password facilities use the same crypt(3) function the system login facilities use and encrypted passwords are stored in the **etc/hosts** file in a format that is compatible with most systems' password files. This means that client passwords can be copied from a password file if desired (though this is obviously discouraged). The HylaFAX client-server protocol `ADDUSER` request includes a variety of checks for easy-to-guess passwords.

HylaFAX uses the standard system crypt function to implement client passwords. Systems without this routine may substitute the publicly available GNU version but then it may not be possible to copy encrypted passwords from the system password file.

If a client is prompted for a password when contacting the server then it means the client is setup with a password in the **etc/hosts** file. Read the hosts(4F) manual page carefully to understand the format for this file.

Logging of client login and network connections can be controlled separately by bits in the hfaxd `ServerTracing` configuration parameter; consult the hfaxd manual page for details.

TROUBLESHOOTING: SERVER BASICS

One faxq process does the job scheduling and initiation of outbound calls; it handles many tasks:

- manage the queue of jobs and schedule their processing

- prepare PostScript, TIFF, and PCL documents for transmission
- start the processes that do the low-level work to deliver facsimile or pager messages

One faxgetty server process is usually run for each fax modem on a machine. These processes are responsible for handling inbound calls, carrying out the following tasks:

- answer incoming call and decide whether the call is fax, data, or voice
- do the low-level work to receive facsimile
- possibly invoke the system getty program to handle data calls
- possibly invoke the system vgetty program to handle voice calls
- possibly invoke the system egetty program to deduce the type of a call

Proper setup of HylaFAX involves setting up the configuration files and the ancillary programs and command scripts that are invoked by faxq and by faxgetty. The setup of the ancillary programs should automatically be done when the software is configured and installed. The modem-related setup and much of the system-related configuration work should be done by the faxsetup and faxaddmodem commands. Problems that arise in the normal operation of a server typically fall into two categories:

- incorrect configuration and/or setup of the server software,
- incorrect configuration of the modem.

In either case HylaFAX provides extensive *tracing* or *logging* facilities that should supply the information needed to locate and correct a problem.

Tracing information is broken up into two separate areas: *server tracing* and *session tracing*. Server tracing is the logging done by server processes when not in active conversation with another device such as a facsimile machine or paging service provider. This tracing covers modem initialization, scheduler operation, and general bookkeeping and maintenance work. Session tracing is the logging done while a server process is actively communicating with a remote device; e.g. the communcation work done to send or receive a facsimile. This split permits better control over the amount of information logged in a production environment. Typically once a system is setup and working the amount of server tracing information captured can be quite small while the session tracing logged needs to be more extensive to help debug any communication problems that might arise.

HylaFAX servers processes are controlled by a collection of *configuration files*. There is a configuration file for the faxq scheduler process, **etc/config**, and one file for each process that uses a modem, **etc/config.*device***. (The hfaxd process that implements the client-server protocols also has its own configuration file.)

faxq tracing information is controlled by the `ServerTracing` and `LogFacility` configuration parameters. `ServerTracing` controls which work should be traced while `LogFacility` specifies the syslog(3) facility where the tracing messages should be directed. By default server tracing information is directed to the ‘‘`daemon`’’ facility.

Modem-related tracing information is controlled by `ServerTracing`, `SessionTracing`, and `LogFacility` parameters that are put in the per-modem configuration files. `ServerTracing` controls the logging done when one of these programs is not in active conversation with another device while `SessionTracing` controls the logging during other times. As before, `LogFacility` defines where server

tracing messages get sent. Session tracing messages are not sent using syslog; they are written to *session log files* that are described separately below.

To capture server tracing messages you must enable the appropriate bits in a `ServerTracing` parameter **and** configure the syslogd process to capture messages sent to the `daemon.debug` facilitiy (or substitute for `daemon` to reflect the value of the `LogFacility` parameter).

*By setting `LogFacility` to something like `local5` it is easy to capture HylaFAX syslog messages in a separate file; just setup the **syslog.conf** file appropriately, e.g.*

```
local5.debug        /var/spool/fax/etc/syslog
```

A sample server trace log is shown below. The lines marked ‘‘`FaxQueuer`’’ are generated by the faxq process. The lines marked ‘‘`FaxGetty`’’ are generated by a faxgetty process. The process ID of each process is shown enclosed in ‘‘[]’’. Each line is marked with the date and time that it was generated.

```
Apr  1 12:25:31 6V:oxford FaxQueuer[307]: HylaFAX (tm) Version 3.0beta112
Apr  1 12:25:31 6V:oxford FaxQueuer[307]: Copyright (c) 1990-1996 Sam Leffler
Apr  1 12:25:31 6V:oxford FaxQueuer[307]: Copyright (c) 1991-1996 Silicon Graphics, Inc.
Apr  1 12:25:58 6V:oxford FaxQueuer[307]: MODEM ttym2 DOWN
Apr  1 12:25:58 6V:oxford FaxGetty[477]: OPEN /dev/ttym2
Apr  1 12:26:03 6V:oxford FaxGetty[477]: MODEM TELEBIT T3000SA - Version LA7.20F/T3000SA - Version
Apr  1 12:26:04 6V:oxford FaxQueuer[307]: MODEM ttym2 READY
```

The tracing facilities supported by the HylaFAX server processes should provide enough information to debug any problem in the software. Consult the config(4F) manual page for complete information on the various parameters.

TROUBLESHOOTING: SCHEDULER OPERATION

HylaFAX requires a faxq process to be running for outbound transmissions to be done. The faxstat program should show this process running in its output:

```
HylaFAX scheduler on hyla.chez.sgi.com: Running
Modem ttyf2 (+1 510 528-9999): Running and idle
```

If the scheduler is marked ‘‘Not running’’ but faxq *is* running on the server machine then the first thing to check is the server tracing log for faxq to see if there is a problem. faxstat gets its information from hfaxd on the server machine and hfaxd decides if faxq is running based on whether it can open the FIFO special file **FIFO** in the spooling area on the server machine. Check that this file is the correct type and that it has the appropriate permission. faxq will create this file when it starts up if it does not exist; if there is a problem try stopping faxq, remove the file, possibly fiddle with faxq’s server tracing (to get more information logged), and then restart faxq:

```
hyla# faxquit
  ...verify that faxq has terminated...
hyla# /usr/local/sbin/faxq
```

Note that if the faxquit command does not cause faxq to terminate then you may need to forcibly kill the process (but beware of doing this when outbound jobs are in progress as this can leave the system in an inconsistent state). When sending a signal to faxq use SIGTERM or SIGINT; faxq catches these signals and tries to cleanup its state as much as possible.

FIFO-related problems usually happen when the FIFO special file is removed without first stopping the HylaFAX server processes. This can cause one or more processes to be left with an open file descriptor to a file that is no longer present in the filesystem, and hence unreachable. When doing maintenance work on a HylaFAX server it is a good idea to first shut down the server processes. This is simple to do with the **hylafax** shell script used on System V-style systems; simply do

```
hyla# /etc/init.d/hylafax stop
```

If you believe that you are having problems with the messages exchanged by the various HylaFAX server processes `ServerTracing` bit 0x04000 can be set to force logging of all the messages sent and received through FIFO special files; usually you need to do this only for the faxq process.

Other than FIFO communication problems, the most common scheduler-related problem encountered is that no outbound jobs get scheduled for processing. faxq will only schedule an outbound job when it believes it has a modem available that is capable of handling a job's needs. A modem's existence and capabilities are signalled to faxq through messages received on its FIFO file. The two programs that send modem information are faxgetty and faxmodem. Processes that want to send faxq information about new modems must be able to access the file; this means the protection on the file must be such that clients can open it for writing (the default setup should make this happen). The order in which faxq and faxgetty are started does not matter; a handshaking protocol between faxq and faxgetty insures that modem status information will be exchanged no matter what order the processes startup in.

When in doubt about what is happening, or not happening, to jobs enable the job queue management tracing bit in the `ServerTracing` parameter for the faxq process; e.g.

```
ServerTracing:  0x201
```

and check the log to understand what is going on. There is also a separate bit for tracing low-level job queue operations; this should only be needed on rare occasions.

Otherwise there is very little that can go wrong with the scheduler with respect to managing the queue of outbound jobs. Setting the system time backwards on the server machine can cause problems as timers managed by faxq are calculated relative to the current time-of-day. Jobs may be rejected without a phone call if a `rejectNotice` entry is present for a destination phone number in the destination controls file destctrls(4F). Beware that multiple jobs to the same destination are usually serialized to reduce phone calls. Jobs that are blocked in this way have a "*Blocked by concurrent...*" status. You can change the maximum number of concurrent jobs that will be scheduled to a destination with the `MaxConcurrentJobs` configuration parameter.

TROUBLESHOOTING: SESSION TRACING

The `SessionTracing` parameter controls tracing information *during the time HylaFAX is engaged in conversation with another device* (fax machine, pager service provider, etc.) Tracing of this sort is done by faxgetty processes (when receiving facsimile) and by processes started up by faxq to process outbound jobs (faxsend, pagesend, etc.). Session tracing is controlled by configuration parameters specified in the per-modem configuration files. It is also possible to enable session tracing on a per-destination basis for outbound jobs through the per-destination `DestControls` facility provided by

faxq.

Communication-related problems will be found in the information logged under session tracing. Session tracing information is stored in files in the **log** subdirectory in the spooling tree and is returned to users via electronic mail when notification is requested, or when an unrecoverable error is encountered. The log(4F) manual page has information on the meaning of many messages that might appear in these files.

A sample snippet from a session log is shown below. The trace was collected from a transmission through a Class 1 modem. The `SessionTracing` parameter was set to `0x4f`. Notice that the format is very similar to the server tracing information collected through syslog (shown above). Beware that, unlike previous versions of HylaFAX, session logs are collected in individual files that are uniquely named according to a unique *communication identifier*. This can make locating the session log file a bit tricky; to find the appropriate log file you can look for a record in the **etc/xferlog** file or a server tracing message. Either should contain a communication identifier which can then be used to select the appropriate file from the **log** directory in the spooling area. (If only one call is being handled on a server you can also just look for the most recently changed file in the **log** directory.)

Session logs are straightforward to understand. Messages sent *to the modem* are identified by lines with a ‘‘`<--`’’ mark while data received *from the modem* are identified by lines with a ‘‘`-->`’’. Timestamps show the date and time, with time to the right of the decimal point displayed to 10 millisecond precision (the typical granularity of the realtime clock on a UNIX system). The number of characters in each message prefix the message itself. Unimportant binary data, usually the facsimile page data, is sometimes shown generically as ‘‘`data`’’.

```
Apr 01 09:01:09.78: [18302]: SESSION BEGIN
Apr 01 09:01:09.80: [18302]: MODEM set DTR OFF
Apr 01 09:01:09.80: [18302]: MODEM set baud rate: 0 baud (flow control unchanged)
Apr 01 09:01:09.80: [18302]: DELAY 2600 ms
Apr 01 09:01:12.40: [18302]: MODEM set DTR ON
Apr 01 09:01:12.40: [18302]: MODEM set baud rate: 38400 baud, input flow RTS/CTS, output flow RTS/C
Apr 01 09:01:12.40: [18302]: MODEM flush i/o
Apr 01 09:01:12.40: [18302]: <-- [14:ATE0V1Q0S0=0H0]
Apr 01 09:01:12.53: [18302]: --> [2:OK]
Apr 01 09:01:12.53: [18302]: <-- [20:ATS8=2S7=60&K4&D2&C1]
Apr 01 09:01:12.66: [18302]: --> [2:OK]
Apr 01 09:01:12.66: [18302]: <-- [11:AT+FCLASS=1]
Apr 01 09:01:12.78: [18302]: --> [2:OK]
Apr 01 09:01:12.78: [18302]: <-- [4:ATM0]
Apr 01 09:01:12.90: [18302]: --> [2:OK]
Apr 01 09:01:12.90: [18302]: <-- [11:AT+FCLASS=1]
Apr 01 09:01:13.02: [18302]: --> [2:OK]
Apr 01 09:01:13.06: [18302]: DIAL 14159657824
Apr 01 09:01:13.06: [18302]: <-- [15:ATDT14159657824]
Apr 01 09:01:23.85: [18302]: --> [7:CONNECT]
Apr 01 09:01:23.92: [18302]: MODEM input buffering disabled
Apr 01 09:01:25.45: [18302]: --> HDLC<26:FF C0 02 2C 4C 1C EC AC 6C 9C AC 8C 2C 8C D4 04 04 04 04 0
Apr 01 09:01:25.45: [18302]: --> [2:OK]
Apr 01 09:01:25.45: [18302]: REMOTE CSI "+14159657824"
Apr 01 09:01:25.45: [18302]: <-- [8:AT+FRH=3]
Apr 01 09:01:25.45: [18302]: --> [7:CONNECT]
Apr 01 09:01:25.79: [18302]: --> HDLC<10:FF C8 01 00 76 5F 00 C6 4A 7E>
Apr 01 09:01:25.79: [18302]: --> [2:OK]
Apr 01 09:01:25.82: [18302]: REMOTE best rate 14400 bit/s
Apr 01 09:01:25.82: [18302]: REMOTE max page width 2432 pixels in 303 mm
Apr 01 09:01:25.83: [18302]: REMOTE max unlimited page length
Apr 01 09:01:25.83: [18302]: REMOTE best vres 7.7 line/mm
Apr 01 09:01:25.83: [18302]: REMOTE best format 1-D MR
Apr 01 09:01:25.83: [18302]: REMOTE best 0 ms/scanline
Apr 01 09:01:25.83: [18302]: REMOTE does not support PostScript transfer
Apr 01 09:01:25.83: [18302]: USE 14400 bit/s
Apr 01 09:01:25.83: [18302]: USE 0 ms/scanline
Apr 01 09:01:25.83: [18302]: SEND file "docq/doc19.cover;30"
```

. . . .

Normal installation of HylaFAX will enable enough session tracing to debug most communication problems. The default configuration files come with `SessionTracing` set to 11 which is a good setting for Class 2 modems (i.e. a lot of information is provided, but the load on the server should not keep it from operating properly). For Class 1 modems a setting of 0x4f will also cause HDLC frames to be collected. Beware of tracing timer operations and modem I/O; these trace flags are only useful if you are trying to debug a problem specifically related to a timer not going off, or a problem where data appears to be corrupted.

When debugging modem-related problems, only enable the tracing that you really need. Enabling all tracing can affect the operation of the server processes by altering the timing of operations.

Note that when capturing a trace for the purpose of submitting a bug report, the less extraneous information that you include, the easier it is for people to help understand the problem. Most of the time HylaFAX will return the relevant session log for a communication failure in the notification message sent to a user when an outbound job fails. Note however that the contents of this log is controlled by the value of the `SessionTracing` parameter specified in the per-modem configuration files. If this parameter is set too low then session logs may be returned that do not show sufficient information to diagnose a problem.

TROUBLESHOOTING: POSTSCRIPT DOCUMENT PREPARATION

PostScript documents submitted for transmission as facsimile are converted to a binary format by the ps2fax(1M) script that is invoked by the scheduler. If this preparation fails it is either due to the submission of invalid PostScript or a problem in the setup of the PostScript RIP that does the conversion from PostScript to TIFF/F. faxq tries to return any error messages returned by the PostScript RIP to the user that submits a job but some programs make this difficult. In this case it may be easiest to see what is happening by invoking the ps2fax script directly using the same command arguments used by faxq; this information can be found in the faxq trace log. Beware however, that if faxq is started up by the init(1M) program that it may inherit a different shell environment. In particular, beware of problems with search paths when the PostScript RIP is linked with Dynamic Shared Objects (DSOs); e.g. when Ghostscript is linked with the the X11 driver and a DSO version of the X library.

*On machines with dynamic shared libraries (e.g. SunOS), if you link Ghostscript with the X11 device driver and use shared X11 libraries that are not in a standard location, then you may need to augment the HylaFAX **util/ps2fax.gs.sh** script with something of the form:*

```
LD_LIBRARY_PATH=/usr/local/R5/lib:/usr/openwin/lib
export LD_LIBRARY_PATH
```

The faxsetup script should verify that the PostScript RIP is installed in the correct location and properly configured for use with HylaFAX. It may not however be able to verify DSO-related problems of the nature described above.

PostScript imaging problems may also result in the faxq process not being able to reopen the imaged document after the ps2fax script is run. After faxq invokes ps2fax to image a document it validates the

resulting TIFF/F file to make sure the work was done correctly. This is necessary because some programs terminate with exit status 0 indicating a successful run even if an error was encountered.

Some other potential problems to be aware of. If Ghostscript is configured with a `tiffg32d` device driver to generated 2D-encoded data beware that versions of Ghostscript prior to 3.12 (inclusive) had a bug in this driver that caused invalid data to be generated. If you are in doubt you can disable the use of 2D-encoded facsimile data with the `User2D` configuration parameter to faxq:

```
Use2D: no
```

(remember this goes in faxq's **etc/config** file, not the per-modem configuration file).

Versions of Ghostscript prior to about 3.63 permitted PostScript documents to set the output page dimensions to arbitrary values. Some applications such as Frame 4.0 and various PostScript drivers found in Microsoft Windows used this facility to force the output page to be 1734 pixels wide (8.5 inches at 204 pixels/inch). This causes problems when HylaFAX requests that documents be imaged with pages that are 1728 pixels wide. The result is that documents will be submitted, imaged, and then rejected during transmission because they have an incorrect page width. The solution is to get a current version of Ghostscript or to edit the PostScript documents to remove the `setpagedevice` requests that (incorrectly) force the output page dimensions.

TROUBLESHOOTING: TIFF DOCUMENT PREPARATION

Like PostScript documents TIFF documents may need to be prepared before they can be transmitted as facsimile. This work is done by the tiff2fax(1M) script that is invoked by faxq. The tiff2fax script depends on programs that are part of the separate TIFF software distribution and, in some instances, the ps2fax script. If you encounter a problem preparing a TIFF document for transmission capture the command arguments to the tiff2fax script from the faxq log and try running it by hand.

TROUBLESHOOTING: COMMUNICATION PROBLEMS

Communication problems refer to errors that occur during an outbound call handled by faxsend or pagesend, or an inbound call handled by faxgetty. Almost all communication problems can be diagnosed from the information in a session log. The only server tracing information that might be needed is for modem setup work done by faxgetty which does not appear in a session log.

If a problem occurs during modem setup by faxgetty, set `ServerTracing` to 11, or similar, in the modem configuration file and check the server trace log. Modem initialization for an outbound job is included in the session log and controlled by the `SessionTracing` parameter. Problems during modem setup are typically caused by:

- misconfiguration of the flow control scheme,
- not disabling the getty program on the modem port (so that multiple processes are simultaneously trying to read/write to the modem), and

- invalid initialization sequences.

Once a call is placed, facsimile communication happens in several *phases*. First the sender and receiver negotiate a set of *session parameters* to use during communication. These parameters define the format of data that is to be exchanged, the physical dimensions of the pages, and certain other parameters related to the communication work (speed at which to transfer data, modulation scheme, time to delay between raster scanlines to permit a receiver's printer to run properly, etc.). An example of this negotiation for a Class 2.0 modem is:

```
Aug 16 09:54:03.26: [28224]: DIAL xxxxxxxxx
Aug 16 09:54:03.26: [28224]: <-- [14:ATDTxxxxxxxxx\r]
Aug 16 09:54:25.33: [28224]: --> [4:+FCO]
Aug 16 09:54:25.33: [28224]: --> [26:+FCI:Gupta Corporation    ]
Aug 16 09:54:25.33: [28224]: REMOTE CSI "Gupta Corporation"
Aug 16 09:54:25.33: [28224]: --> [20:+FIS:1,3,0,2,0,1,0,4]
Aug 16 09:54:25.33: [28224]: --> [2:OK]
Aug 16 09:54:25.33: [28224]: REMOTE best rate 9600 bit/s
Aug 16 09:54:25.33: [28224]: REMOTE max page width 1728 pixels in 215 mm
Aug 16 09:54:25.33: [28224]: REMOTE max unlimited page length
Aug 16 09:54:25.34: [28224]: REMOTE best vres 7.7 line/mm
Aug 16 09:54:25.34: [28224]: REMOTE best format 1-D MR
Aug 16 09:54:25.34: [28224]: REMOTE supports T.30 Annex A, ECM
Aug 16 09:54:25.34: [28224]: REMOTE best 20 ms, 10 ms/scanline
Aug 16 09:54:25.34: [28224]: USE 9600 bit/s
Aug 16 09:54:25.34: [28224]: USE 20 ms, 10 ms/scanline
Aug 16 09:54:25.34: [28224]: SEND file "docq/doc1617.ps;30"
Aug 16 09:54:25.34: [28224]: USE page width 1728 pixels in 215 mm
Aug 16 09:54:25.35: [28224]: USE unlimited page length
Aug 16 09:54:25.35: [28224]: USE 3.85 line/mm
Aug 16 09:54:25.35: [28224]: USE 1-D MR
Aug 16 09:54:25.35: [28224]: <-- [23:AT+FIS=0,3,0,2,0,0,0,4\r]
Aug 16 09:54:25.45: [28224]: --> [2:OK]
```

The receiver announces its *capabilities* and the sender then selects which of these capabilities it wants to use in doing the transmission.

Beware that a common problem in Class 2 modems is for the modem to reject or ignore an `AT+FDIS` *command to set the session parameters between the time a call is placed and a page is transmitted. When this problem occurs pages may appear ''squished'' or the session may be aborted abnormally. If you suspect this problem use the* `Class2DDISCmd` *configuration parameter to enable workaround support; e.g.*

```
Class2DDISCmd:      AT+FDIS
```

Following the negotiation of the session parameters pages of facsimile data are transmitted followed by a *post page exchange* of messages. For example:

```
Aug 16 09:54:25.45: [28224]: <-- [7:AT+FDT\r]
Aug 16 09:54:46.53: [28224]: --> [20:+FCS:0,1,0,2,0,0,0,5]
Aug 16 09:54:46.53: [28224]: --> [7:CONNECT]
Aug 16 09:54:46.53: [28224]: SEND begin page
Aug 16 09:54:46.56: [28224]: <-- data [1029]
Aug 16 09:54:46.56: [28224]: <-- data [1024]
Aug 16 09:54:46.56: [28224]: <-- data [1026]
Aug 16 09:54:46.56: [28224]: <-- data [1026]
Aug 16 09:54:46.83: [28224]: <-- data [1025]
Aug 16 09:54:47.10: [28224]: <-- data [1027]
Aug 16 09:54:47.36: [28224]: <-- data [1024]
Aug 16 09:54:48.20: [28224]: <-- data [1024]
Aug 16 09:54:48.65: [28224]: <-- data [845]
Aug 16 09:54:49.11: [28224]: SENT 9037 bytes of data
Aug 16 09:54:49.11: [28224]: SEND 1D RTC
Aug 16 09:54:49.11: [28224]: <-- data [9]
Aug 16 09:54:49.56: [28224]: SEND end page
Aug 16 09:54:49.56: [28224]: SEND send MPS (more pages, same document)
```

```
Aug 16 09:54:49.56: [28224]: <-- data [2]
Aug 16 09:55:18.11: [28224]: --> [2:OK]
Aug 16 09:55:18.11: [28224]: SEND recv MCF (message confirmation)
```

In this case after the page an ‘‘MPS’’ message was sent to indicate that this page was to be followed by more pages of the same document. The receiver responded with an ‘‘MCF’’ message indicating the received page of facsimile data was OK (had an acceptable number of errors, if any) and that the sender should commence sending the next page of data. This procedure continues until the sender is done in which case it signs off by sending an ‘‘EOP’’ message at the end of the last page to be transmitted.

The basic work described above occurs for all facsimile communication no matter whether it is done with a Class 1, 2, or 2.0 modem. Two things to understand from this abbreviated description:

- Facsimile pages *are not considered to be delivered until an acknowledgement is received from the receiver*. A receiver may explicitly reject or ‘‘NAK’’ a page for many reasons; HylaFAX will retransmit information as required to deliver a facsimile.
- The format of facsimile data transmitted by HylaFAX is based on the negotiated set of session parameters. If a facsimile document is not properly formatted or prepared according to the session parameters then the receiver may reject it. HylaFAX optimally *preformats* facsimile data according to any user-specified requests and any cached capabilities for the receiver. If this information is wrong then it will disconnect and reformat the documents according to the correct parameters.

A more detailed description of the underlying facsimile protocol can be found at http://www.grayfax.com/ and, of course, in the CCITT/ITU T.30 recommendation.

When debugging a communication problem try to identify if the problem is transient or repeatable and if the problem always happens when communicating with the same sender/receiver. Also check the negotiated session parameters to see if there is any correlation, for example, to the negotiated data format (i.e. 1D-encoded data versus 2D-encoded data). Communication problems can be caused by many things including:

- misconfiguration of the communication equipment used by HylaFAX.
- poor or faulty implementations of the fax protocols in the sender and/or receiver, or
- line noise caused by a poor quality phone connection or other environmental problem (e.g. an answering machine on the same phone line),

If you can communicate with some facsimile devices but not others than you can usually rule out the first cause. Transient or unrepeatable problems, jobs that require one or more retransmissions before working, or other similar problems are frequently due to line noise or poor phone connections but sometimes incorrect protocol implementations or host-modem flow control problems that depend on host load or page content. If you believe your have a noise problem it is a good idea to isolate the modem on the phone line (i.e. remove any other equipment such as an answering machine or handset) and to try an alternate line if possible, though this may not matter if the problem is *close to the receiver*. Last but not least, when using a Class 2 or 2.0 modem check with the vendor to make sure the firmware revision for the modem is up to date. If you can reliably reproduce a problem then knowing the make and model of the device at the other end can help a modem vendor understand and/or fix a protocol implementation problem.

If communication problems persist to a specific receiver when using 2D-encoded data, you can

If communication problems persist to a specific receiver when using 2D encoded data, you can disable its use through the info(4F) database. Note also that sendfax has a command line option that lets you select 1D-encoded data in any single transmission.

The HylaFAQ contains information on some common communication problems that might be encountered. It is also important to monitor the operation of a server to detect trends that might indicate modem or telecommunication problems; the faxcron(1M) script is useful for doing this since it extracts the transcripts of failed calls.

TROUBLESHOOTING: GETTY PROBLEMS

HylaFAX will invoke the getty program when a data connection is established and the `GettyArgs` parameter is set to a non-null string. When HylaFAX starts up a getty program it sets the standard input, output, and error descriptors to the modem device (closing all other descriptors), creates a new process group, and turns off the `CLOCAL` bit on the tty device so that if carrier is dropped the process group will receive a SIGHUP signal.

- be sure that the system is not also configured to start getty on the modem's tty port
- beware of misconfigured getty arguments that cause getty to reopen and/or initialize the modem device
- also be careful that on some systems the **/etc/ttytab** entry for the modem port must be setup as a *dialup* line as opposed to a *local* line

The section on ''System-specific Guidance'' in the chapter on setting up a server and the HylaFAQ have information about other problems that you may encounter.

TROUBLESHOOTING: PROBLEMS WITH UUCP, CU, TIP, ETC.

If you have a problem running the fax software together with other communication programs such as uucp, cu, tip, slip, ppp, etc. first verify that your data communication software is configured to use the correct modem initialization strings and that both HylaFAX and the program(s) in question are using *the same device name*. Many of the prototype modem configuration files for Class 2 and Class 2.0 modems will leave the modem *idling* in Class 2 or 2.0. This means that in order to place a data call the modem must first be reset to Class 0; e.g.

```
AT+FCLASS=0DT<phone number>
```

Other common problems involve the ownership and protection of the modem device file. When a HylaFAX server process is running it forces the tty device to be owned by the ''fax'' user (typically the same UID as the ''uucp'' user) and to have the mode specified by the `DeviceMode` configuration parameter. Finally, beware that there are several different styles of UUCP lock files; verify that your UUCP and related programs use the same style that HylaFAX is configured to use. The UUCP lock file scheme used by HylaFAX may be specified with the `UUCPLockType` configuration parameter. If you specify this parameter be certain to put it in both the faxq configuration file and each modem configuration file; otherwise one HylaFAX server process may do the right thing while another may not.

Contributed Software

You can find contributions from HylaFAX users in the **contrib** subdirectory of the public FTP area where you found this software (or look in ftp://ftp.sgi.com/sgi/fax/contrib). New contributions are welcome, so long as the author's identity is clearly marked so that questions go directly to the author and not to me. The HylaFAQ also has information about some of the more popular contributed software programs.

HylaFAX Mailing Lists and Other Final Words

There are two mailing lists for users of the HylaFAX software:

- flexfax@sgi.com, a public list for all users of the HylaFAX software (actually located at celestial.com);
- flexfax-announce@sgi.com, a low volume list where announcements of new versions of HylaFAX are posted.

`flexfax@sgi.com` tends to have a fair amount of traffic about the server-side software; if you are not running a fax server it may not be worth joining. Beware also that this mailing list has many people on it. Please take this into consideration when posting notes to it; i.e. *avoid posting large trace logs and the such*. Also, when corresponding about this software please always specify:

- what version you have (check here if you are unsure of your software version),
- what system you're running on, and,
- if the problem is modem-related, identify the modem and the firmware revision.

For example: "HylaFAX v2.3.0alpha99 under Solaris 2.3 with gcc 2.5.7; ZyXEL 1496E with 6.11a firmware."

There is no need to subscribe to both the `flexfax` and `flexfax-announce` mailing lists; postings to the announcement list are automatically fed to the normal `flexfax` list.

If you want to subscribe to either of these lists you need to send electronic mail that includes **only** a *subscribe* request in the body of the mail; for example, to subscribe to the `flexfax` mailing list you might send:

```
To: flexfax-request@sgi.com

subscribe
```

To specify a mailing address other than the one used to post your note, specify it in the mail; e.g. to subscribe the user `hopeless@lost.com` to the `flexfax-announce` mailing list use:

```
To: flexfax-announce-request@engr.sgi.com

subscribe hopeless@lost.com
```

When you have been added to the mailing list you will receive confirmation by electronic mail at the subscribed address.

*To unsubscribe from a list send an unsubscribe request like the above to the same mail address; **do not send administrative requests to the mailing list**.*

To subscribe to the flexfax mailing list send your request to flexfax-request@sgi.com. To subscribe to the `flexfax-request` mailing list send your request to flexfax-announce-request@engr.sgi.com.

If you have not previously done so, please fill out and submit the survey form. The information provided in this form is purely for use in assisting users of HylaFAX.

ACKNOWLEDGEMENTS

This software is more than 6 years old and is the product of many folks' work. Robin Schaufler did the original scheme for delayed submission and worked on the original client-server protocol. More recently, the following people have helped either by testing or by contributing fixes and/or improvements:

Matthias Apitz	Chris Beekhuis	Peter Bentley
Marc Boucher	Piete Brooks	Bill Campbell
Brent Chapman	Camden Clarke	Tom Corson
Alan Crosswell	Mark Diekhans	Greg Ferguson
Steve Fine	Andrew Ford	Nico Garcia
Wolfgang Henke	Viet Hoang	Bert Hooyman
Ken Hornstein	Dirk Husemann	Brian Katzung
Masao Kitano	Carsten Koch	John T Kohl
Scott Kramer	Rickard Linck	Rick Lyons
Tom Lislegaard	Rob MacKinnon	Kevin McManamon
Les Mikesell	Bill Morrow	Andy Moskoff
Chris Munonye	Rob Newberry	Dag Nygren
Jonas Olsson	Dave Packer	Damon Permezel
David Pike	Amir Plivatsky	Andy Rabagliati
Eric Rescorla	Marshall Rose	Daniel Rosenblatt
Joel Rosi-Schwartz	Tim Rylance	Joseph E. Sacco
Thomas A. Szybist	Brent Townshend	Peter White
Steve Williams	Scott Woodard	David Vrona
Paul Vixie	Christian Zahl	

(and surely others). Also, a special thanks to Ed McCreight for helping me understand the stuff *between the lines* that's necessary to make a working Class 1 driver.

The home page artwork was sketched by Rob Myers. I then butchered it with my handy use

The home page artwork was sketched by Rob Myers. I then butchered it with my handy use of the freely available gimp program. If you like the art send Rob a compliment. If you don't like it blame me, or better yet, send me a better version to replace it (please). Please note that the artwork is copyright Silicon Graphics Inc. and is not to be used without written permission.

Regular Expression Support

The regular expression support is based on Henry Spencer's POSIX 1003.2 compliant regex package that has

```
Copyright 1992, 1993, 1994 Henry Spencer.
All rights reserved.
```

Consult **regex/COPYRIGHT** for the full copyright notice associated with this software.

Compression Support

The zlib package used in the client-server protocol has the following copyright notice:

```
(C) 1995-1996 Jean-loup Gailly and Mark Adler

 This software is provided 'as-is', without any express or implied
 warranty.  In no event will the authors be held liable for any damages
 arising from the use of this software.

 Permission is granted to anyone to use this software for any purpose,
 including commercial applications, and to alter it and redistribute it
 freely, subject to the following restrictions:

 1. The origin of this software must not be misrepresented; you must not
    claim that you wrote the original software. If you use this software
    in a product, an acknowledgment in the product documentation would be
    appreciated but is not required.
 2. Altered source versions must be plainly marked as such, and must not be
    misrepresented as being the original software.
 3. This notice may not be removed or altered from any source distribution.

 Jean-loup Gailly        Mark Adler
 gzip@prep.ai.mit.edu    madler@alumni.caltech.edu
```

Consult the file **zlib/README** for more information on the package.

PCF Font Support

The code to read PCF fonts is distantly related to the X11R5 code that is

```
Copyright 1990 Massachusetts Institute of Technology
```

Consult **faxd/PCFFont.c++** for the full copyright notice.

Text to PostScript Conversion

The `textfmt` program is *very distantly* related to the `lptops` program written by Nelson Beebe; there was no copyright notice on the version of the code that `textfmt` grew out of.

Configuration Support

The **config.guess** and **config.sub** scripts are part of the GNU autoconf package and covered by the GNU Public License (GPL). Several ideas in the configure script are directly "borrowed" from autoconf (and I have tried to maintain as much compatibility as possible).

Sample PCF Font

The PCF font **etc/lutRS18.pcf** included for use with tag lines is a compiled version of a LucidaTypewriter font that was contributed to X11 by Bigelow & Holmes. Redistribution of this font requires inclusion of this copyright notice:

```
NOTICE TO USER: The source code, including the glyphs or icons
forming a par of the OPEN LOOK TM Graphic User Interface, on this
tape and in these files is copyrighted under U.S. and international
laws. Sun Microsystems, Inc. of Mountain View, California owns
the copyright and has design patents pending on many of the icons.
AT&T is the owner of the OPEN LOOK trademark associated with the
materials on this tape. Users and possessors of this source code
are hereby granted a nonexclusive, royalty-free copyright and
design patent license to use this code in individual and
commercial software. A royalty-free, nonexclusive trademark
license to refer to the code and output as "OPEN LOOK" compatible
is available from AT&T if, and only if, the appearance of the
icons or glyphs is not changed in any manner except as absolutely
necessary to accommodate the standard resolution of the screen or
other output device, the code and output is not changed except as
authorized herein, and the code and output is validated by AT&T.
Bigelow & Holmes is the owner of the Lucida (R) trademark for the
fonts and bit-mapped images associated with the materials on this
tape. Users are granted a royalty-free, nonexclusive license to use
the trademark only to identify the fonts and bit-mapped images if,
and only if, the fonts and bit-mapped images are not modified in any
way by the user.


Any use of this source code must include, in the user documentation
and internal comments to the code, notices to the end user as
follows:


(c) Copyright 1989 Sun Microsystems, Inc. Sun design patents
pending in the U.S. and foreign countries. OPEN LOOK is a
trademark of AT&T. Used by written permission of the owners.


(c) Copyright Bigelow & Holmes 1986, 1985. Lucida is a registered
trademark of Bigelow & Holmes. Permission to use the Lucida
trademark is hereby granted only in association with the images
and fonts described in this file.



SUN MICROSYSTEMS, INC., AT&T, AND BIGELOW & HOLMES
MAKE NO REPRESENTATIONS ABOUT THE SUITABILITY OF
```

```
THIS SOURCE CODE FOR ANY PURPOSE. IT IS PROVIDED "AS IS"
WITHOUT EXPRESS OR IMPLIED WARRANTY OF ANY KIND.
SUN  MICROSYSTEMS, INC., AT&T AND BIGELOW  & HOLMES,
SEVERALLY AND INDIVIDUALLY, DISCLAIM ALL WARRANTIES
WITH REGARD TO THIS SOURCE CODE, INCLUDING ALL IMPLIED
WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
PARTICULAR PURPOSE. IN NO EVENT SHALL SUN MICROSYSTEMS,
INC., AT&T OR BIGELOW & HOLMES BE LIABLE FOR ANY
SPECIAL, INDIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES,
OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA
OR PROFITS, WHETHER IN AN ACTION OF  CONTRACT, NEGLIGENCE
OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION
WITH THE USE OR PERFORMANCE OF THIS SOURCE CODE.
```

HylaFAX Use and Copyright

Silicon Graphics has seen fit to allow me to give this work away. It is free. There is no support or guarantee of any sort as to its operations, correctness, or whatever. If you do anything useful with all or parts of it you need to honor the copyright notice enclosed below which basically says ‘‘*you can’t use my name or Silicon Graphics’ name without our written permission*’’. I would however be interested in knowing about any interesting work with HylaFAX and, hopefully, be acknowledged. Finally, note that HylaFAX is a trademark of Silicon Graphics, Inc.


```
Copyright (c) 1988-1996 Sam Leffler
Copyright (c) 1991-1996 Silicon Graphics, Inc.
HylaFAX is a trademark of Silicon Graphics

Permission to use, copy, modify, distribute, and sell this software and
its documentation for any purpose is hereby granted without fee, provided
that (i) the above copyright notices and this permission notice appear in
all copies of the software and related documentation, and (ii) the names of
Sam Leffler and Silicon Graphics may not be used in any advertising or
publicity relating to the software without the specific, prior written
permission of Sam Leffler and Silicon Graphics.

THE SOFTWARE IS PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND,
EXPRESS, IMPLIED OR OTHERWISE, INCLUDING WITHOUT LIMITATION, ANY
WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

IN NO EVENT SHALL SAM LEFFLER OR SILICON GRAPHICS BE LIABLE FOR
ANY SPECIAL, INCIDENTAL, INDIRECT OR CONSEQUENTIAL DAMAGES OF ANY KIND,
OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS,
WHETHER OR NOT ADVISED OF THE POSSIBILITY OF DAMAGE, AND ON ANY THEORY OF
LIABILITY, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE
OF THIS SOFTWARE.
```