NAME

HylaFAX − introduction to *HylaFAX* server operation and file formats

DESCRIPTION

HylaFAX is a system for sending and receiving facsimile. It supports queued transmission and asynchronous reception of facsimile. Ancillary programs are invoked by the system for flexibility and configurability. *HylaFAX* includes client and server programs to support remote submission of jobs for transmission, remote removal of queued jobs, and to remotely query the status of jobs queued for transmission. This document describes the organization of the filesystem spooling area in which *HylaFAX* server and server-related processes operate, and introduces the various files that reside in the spooling area.

OVERVIEW

The spooling area is typically located under the directory */var/spool/hylafax*. Ancillary command scripts used by the server programs *faxq*(8C), *faxsend*(8C), *pagesend*(8C), and *faxgetty*(8C) are located in the **bin** subdirectory. Configuration, access control, and accounting information are maintained in the **etc** and **config** subdirectories. Outgoing jobs are described by files in the **sendq** subdirectory, while received facsimile are deposited in the **recvq** subdirectory. The **docq** and **temp** subdirectories are also used in the preparation of outbound jobs; the latter holds files that may be freely purged while the former holds client files that may reside on the server independent of an associated job. The **doneq** subdirectory holds jobs that have completed but have not yet been purged or archived. On systems with job archival support, completed jobs that have been archived are placed in the **archive** subdirectory. The **pollq** subdirectory holds documents that are available for polled retrieval from the server. The **info** subdirectory contains files that describe the capabilities of facsimile machines that *HylaFAX* has called−this information is used in preparing documents for transmission. The **status** subdirectory contains files that server processes write their current status to. The **log** subdirectory contains logging information about send and receive sessions. The **client** subdirectory contains FIFO special files used for communication with *faxq*.

HylaFAX supports multiple modems on a host. A single process acts as central queueing agent for all outbound jobs. Typically each modem also has a server process that monitors the modem status and handles inbound phone calls. Per-modem server processes communicate with the central queueing agent using FIFO special files; see *mknod*(2) or *mkfifo*(2). Any other synchronization between server processes is done using file-level locking. The *faxq* process listens for commands written to the file named **FIFO**, while each *faxgetty* process listens for commands written to a per-device file named **FIFO**.*devid* (where *devid* is an identifier derived from the name of the device special file to which the modem is connected; e.g. *ttym2* for */dev/ttym2*, *term_10* for */dev/term/10*.) To send a command to the queueing agent, one writes to **FIFO**. This is useful, for example, for submitting a job for transmission. To send a command to a specific *faxgetty* process, the **FIFO**.*devid* file is used.

Client applications interact with a *HylaFAX* server machine using a communications protocol implemented by the *hfaxd*(8C) program. The *hfaxd* program is typically started at system startup; it listens for client requests for service and creates a process for each client. *hfaxd* supports the submission of outbound jobs, querying the status of the send and receive queues, and altering parameters of previously submitted jobs. The *hfaxd* processes communicate with the *faxq* process through FIFO special files. Commands sent to *faxq* are sent to **FIFO** and responses are received on FIFO files that each *hfaxd* creates in the **client** subdirectory.

SETUP

Each *HylaFAX* server machine must run the *faxsetup*(8C) command prior to starting up *HylaFAX* server processes. *faxsetup* verifies that the *HylaFAX* software has been installed correctly and that any parameters that were specified at the time the software was built are appropriate for the system.

SENDING

Each outgoing facsimile job has a job description file that is located in the **sendq** subdirectory. This file contains all the information necessary to manage the transmission; c.f. *sendq*(5F). The actual documents that are to be sent are usually located in the **docq** subdirectory (though it is also possible to reference documents from the **recvq** directory). *HylaFAX* accepts POSTSCRIPT, PDF, PCL, and TIFF documents for transmission (support for PCL documents requires an external application). Documents are automatically converted to TIFF/F documents prior to transmission according to the capabilities of the remote facsimile

machine: maximum page width and length, ability to handle 2D-encoded data, and ability to handle high resolution (7 line/mm) data. This remote machine capability information is stored in files in the **info** subdirectory. If a machine has not been called before, *HylaFAX* assumes the remote machine has the requested capabilities. If a capabilities mismatch is detected while sending a facsimile *HylaFAX* will disconnect and re-convert the submitted documents according to the newly discovered capabilities. Users may also restrict the session parameters used to format documents on a per-job basis.

The actual transmission is handled by a *faxsend*(8C) process that is initiated by the scheduler. This program may be substituted for by specifying the **FaxSendCmd** configuration parameter in the *faxq* configuration file.

While a job is being processed by a server process, its job description file is locked for exclusive use with *flock*(2). The *hfaxd*(8C) program uses this information to tell if a job is actively being processed.

If the **SessionTracing** parameter in a server's configuration file is non-zero, then tracing information for an outgoing job will be logged in a file in the **log** subdirectory. Each destination machine has a separate log file named by its canonical phone number.

The remote job submission facility includes host and user access control. The file **etc/hosts.hfaxd** must be present and list those hosts and users that are permitted to queue jobs for transmission or do other operations that alter the status of a job. Note that it is necessary to include the ''local host'' definition (usually 127.0.0.1) if local submission is to be permitted. For more information consult *hosts.hfaxd*(5F).

If an error is encountered during transmission and a subsequent retransmission would not include the original cover page, then *HylaFAX* can be configured to generate a *continuation cover page* that is prepended to the retransmitted pages. Such cover pages are usually generated by the **bin/mkcover** command; though the exact command to use can be specified in the configuration file read by *faxq*.

HylaFAX can be configured to generate a line of status information across the top of each page of an outbound facsimile. This information, termed a ''tagline'', typically includes the sender's identity (i.e. phone number), the time and date of the transmission, and the page number. The exact format of the tagline is configurable and applications can override the default configuration parameters on a per-job basis. Note that in the United States the law requires that a tagline that identifies the sender's phone number must appear on each transmitted page of facsimile.

Facsimile transmitted to receivers that accept variable-length pages may have short pages ''*chopped*''. That is, if a page has a significant amount of trailing whitespace and the receiver accepts variable-length pages then only the top part of the page will be transmitted. *faxq* can be configured so that only the last page of each document is potentially chopped, all pages are potentially chopped, or chopping is disabled. The minimum whitespace threshold is also configurable. Applications can override the default configuration parameters on a per-job basis.

RECEIVING

faxgetty server processes can be configured to answer incoming phone calls and automatically receive facsimile. Received documents are placed in the **recvq** subdirectory as TIFF Class F files. The *faxgetty* processes can be configured to make these files publicly accessible, or they can be made private in which case an administrator must manage their delivery and/or the assignment of ownership to particular users. When a facsimile is received, the *faxgetty* process usually invokes the **bin/faxrcvd** command; though the exact command to invoke can be specified in the per-modem configuration file. The default *notify* command is a shell script that sends a mail message to a well known user, the *FaxMaster*, but one might also, for example, automatically spool the document for printing.

HylaFAX supports a simple form of access control for receiving facsimile. Each *faxgetty* process may be configured to check the Transmission Subscriber Identifiers (TSI) of the remote fax machine against an access control list, typically **etc/tsi**. Only if the TSI is matched by a regular expression pattern in the file, is the remote machine permitted to transmit a document. This mechanism can be used, for example, to guard against *junk fax*.

HylaFAX can be configured to do *copy quality checking* on received facsimile data. When this feature is enabled *faxgetty* decodes and analyzes the received facsimile data as it is received. If data is received with

too many errors, according to the setting of the **MaxConsecutiveBadLines** and **PercentGoodLines** configuration parameters, then the sender will be told to retransmit the page. When copy quality checking is enabled it is also possible to force received facsimile data to be saved with a different compression scheme than was used for transmission. This function is known as *transcoding* and can significantly reduce the space needed to store received facsimile.

POLLING

HylaFAX supports the polled retrieval of facsimile documents. Documents that are received because of a poll request are stored in the **recvq** subdirectory and also delivered directly to the requester using the **bin/pollrcvd** command; though the exact command to invoke can be specified with the **PollRcvdCmd** configuration parameter. The **pollrcvd** script typically encodes the binary facsimile data and returns it to the user via electronic mail.

INBOUND CALL HANDLING

In environments where Caller-ID information is available, *HylaFAX* also supports a call screening facility similar to the TSI access control facility. *faxgetty* can be configured to check the phone number of each caller against an access control list, typically **etc/cid**. Only if the number is matched by a regular expression pattern in the file is the call answered. All Caller ID information is logged, irregardless of whether or not it is used to screen incoming calls.

faxgetty is also capable of using *distinctive ring* information to identify whether an inbound call is voice, data, or fax. Consult the **RingData**, **RingFax**, and **RingVoice** parameters in *hylafax-config*(5F) for a description of this facility.

DATA CALLS

Most fax modems also support non-facsimile communication. *HylaFAX* uses the locking mechanism employed by *uucp*(1C), *cu*(1C), *slip*(8C), and *ppp*(8C). Any *faxgetty* processes will transparently ''get out of the way'' when an application wants to use a modem for an outgoing call. In addition, *HylaFAX* can be configured to deduce whether an incoming call is for facsimile or data use. If a call from a data modem is recognized and the **GettyArgs** parameter is specified in the configuration file, *faxgetty* will invoke the *getty*(8C) program so that caller may login to the system. Similar functionality is also available for invoking a ''voice getty'' process, though auto-detection of inbound voice calls is less extensive.

STATUS

HylaFAX maintains status information in several forms. General status information for each server process is maintained in the **status** subdirectory and returned to users by the *faxstat*(1) program. The *syslog*(3) facility is used by all server processed for logging status and error diagnostics. The server processes may also be configured to log various kinds of debugging and tracing information; c.f. the **ServerTracing** parameter description in *hylafax-config*(5F).

Any problems encountered when transmitting a facsimile are described in messages returned to the user by electronic mail. A user may also request notification by mail when a job is requeued; for example, because a call failed. Notification by electronic mail is implemented by the **bin/notify** command script; though the name of the script may be overridden with the **NotifyCmd** configuration parameter.

The *faxstat* utility provides (remote) status of jobs queued for transmission, jobs received, and the general status of server processes.

The file **etc/xferfaxlog** contains status information about all facsimile sent and received on a machine. This file is in a simple ASCII format that is easy to manipulate with programs such as *awk*(1), to generate accounting information. See *xferfaxlog*(5F) for information about the format. See *xferfaxstats*(8C) and *recvstats*(8C) for example scripts that print summarized accounting information.

Finally, the *hfaxd* process supports a event monitoring facility that can be access via the *faxwatch*(8C) program. This facility permits clients to register interest in various events and receive ''realtime notification'' when such events occur on the server. Using this facility it is/should-be simple to construct applications that do things like monitor modem status and use.

MODEM STATE CHANGES

In normal operation each modem is managed by a *HylaFAX* server process such as *faxgetty*. These processes communicate with the central scheduler process to notify it when a modem is ready for use, busy for outbound use, or possibly in an unusable state (either purposely marked unavailable or potentially found to be wedged). Modem usage can be explicitly controlled with the *faxstate*(8C) program. The *faxconfig*(8C) program can also be used to dynamically make changes to configuration parameters that may cause a modem to be treated differently (e.g. setting **RingsBeforeAnswer** to zero will cause *faxgetty* to not answer incoming calls).

When *HylaFAX* is used in a send-only configuration there are no *faxgetty* processes and communication must be done directly with the *faxq* process. The *faxstate* program can still be used to manipulate modem use for outbound jobs but the *faxconfig* program is not frequently needed.

JOB SCHEDULING

Outbound jobs are scheduled by a single process. Jobs have a ''scheduling priority'' that is assigned at the time the job is submitted. This priority can be changed at any time the job is not actively being processed using the *faxalter*(8C) program. A job's scheduling priority may also be altered by *faxq* in response to certain scheduling events (e.g. after a failed attempt to send).

Modems are assigned to outbound jobs if they are deemed ready for use. Modem readiness is usually communicated to *faxq* by per-modem *faxgetty* processes. In a send-only environment however this is not possible; instead modems configured for use with *faxmodem* are considered always ready for use unless they are presently assigned to an outbound job or their state is explicitly changed through the *faxstate*(8C) program (*faxstate* can also be used in a send-recv environment).

Each modem has a ''modem priority'' in the range [0..255]. Modems with a lower priority number are assigned to outbound jobs first. Modem priority is statically configured through configuration files, the *faxmodem* program, and the *faxconfig* program. If multiple modems share the same priority value, then *faxq*(8C) will allocate jobs to them in a round-robin fashion.

JOB MANAGEMENT

Outbound jobs are considered to be in one of several states that reflect their treatment by the central scheduling process. Jobs are initially created in a *suspended* state, and may be returned to this state at any time that they are not actively being processed (e.g. a *faxsend* program is running to process the job). Jobs that are suspended are not processed by the scheduler; and their internal state may be safely altered by the owner or a system administrator. Suspending and then releasing a job has the effect of requeueing the job, meaning that it will end up at the bottom of queue for that job's priority. Jobs that are ready for processing by the scheduler are ''submitted'' and their state is changed to be either *pending* (delayed waiting for a future time to send), *sleeping* (delayed waiting for a scheduled timeout), *blocked* (delayed by concurrent activity to the same destination), or *ready* (ready for transmission, waiting only for available resources). When a job is actively processed by the *faxsend* program its state is marked *active*. Jobs that have completed, either successfully or unsuccessfully are placed in a *done* state and their job description files are moved to the **doneq** subdirectory. Clients may still access the state of jobs that are done; until a ''cleaner process'' either purges them from the system or archives their state. This delayed removal of a completed job's state permits clients to resubmit failed jobs using previously transmitted documents and other job state information. The exact mechanics of how and when done jobs are processed is system-dependent; for example, how long a job is left in the done queue before being purged, and whether job archival support is present.

CONFIGURATION

HylaFAX server programs read configuration information from a configuration file. Multiple files are used, one for the *faxq* program and one for each modem. Long-running server programs all automatically re-read their configuration file if it is modified. Typically this re-reading is done frequently enough that administrators do not need to be aware of exactly when it takes place. However in some esoteric cases the file may not be read when expected (the one important case is that the *faxgetty* process reads its configuration file only when answering a call or when resetting a modem; this means that it will not recognize changes when the modem is idle).

In addition to the static configuration files, *HylaFAX* server programs accept commands on their FIFO

special files to alter configuration parameters in the running executable (the *faxconfig*(8C) program can be used to dynamically change configuration parameters). Values set in this way however are lost when the process exits or if the configuration file is re-read.

NOTES

Automatic routing of incoming facsimile is desirable.

FILES

FIFO	fifo for job submission
FIFO.<devid>	fifo for communicating with a faxgetty process
/usr/local/sbin/faxinfo	command that prints information about received facsimile
/usr/local/sbin/faxquit	command to force server to quit
bin/faxrcvd	faxd command for handling newly received facsimile
bin/mkcover	faxd command for generating continuation cover pages
bin/notify	faxd command for doing user notification
bin/pollrcvd	faxd command for delivering facsimile received by poll
bin/ps2fax	faxd command for converting POSTSCRIPT to TIFF
docq/doc*	documents available for transmission
etc/setup.cache	server setup file created by *faxsetup*
etc/cid	caller id access control list
etc/config.<devid>	configuration data for <devid>
etc/hosts.hfaxd	hosts that may submit jobs for transmission
etc/tsi	fax machine receive access control list
etc/xferfaxlog	log of facsimile sent and received
info/*	data base of remote fax machine capabilities
client/*	FIFO special files created by client processes
config/*	prototype configuration files used by *faxaddmodem*
log/*	session logging records
recvq/fax*	received facsimile
sendq/q*	descriptions of jobs queued for transmission
doneq/q*	descriptions of jobs that are done
status/*	server status information
tmp/*	temporary files created when submitting a job
archive/*	database of archived jobs

SEE ALSO

faxsetup(8C), *faxq*(8C), *faxgetty*(8C), *hfaxd*(8C), *faxsend*(8C), *faxrcvd*(8C), *faxconfig*(8C), *faxmodem*(8C), *faxstate*(8C), *notify*(8C), *pollrcvd*(8C), *recvstats*(8C), *xferfaxstats*(8C), *archive*(5F), *hylafax-config*(5F), *dialrules*(5F), *doneq*(5F), *hosts.hfaxd*(5F), *hylafax-info*(5F), *hylafax-log*(5F), *tsi*(5F), *recvq*(5F), *sendq*(5F), *status*(5F), *xferfaxlog*(5F),

Extensive documentation is available in online at http://www.hylafax.org/. Many of these materials are also included in the software distribution.