INTERNET-DRAFT
Expires in Six Months

K. Miller, StarBurst
K. Robertson, StarBurst
A. Tweedly, Cisco
M. White, StarBurst
January, 1997

# StarBurst Multicast File Transfer Protocol (MFTP) Specification

<draft-miller-mftp-spec-02.txt>

## Status of this Memo



## Abstract

The Multicast File Transfer Protocol (MFTP) is a protocol that operates above UDP in the application layer to provide a reliable means for transferring files from a sender to up to thousands (potentially millions with network "aggregators" or relays) of multiple receivers simultaneously over a multicast group in a multicast IP enabled network. The protocol consists of two parts; an administrative protocol to set up and tear down groups and sessions, and a data transfer protocol used to send the actual file reliably and simultaneously to the multiple recipients residing in the group .

## Intellectual Property Rights

StarBurst Communications owns U.S Patent 5,553,083 covering the method for data transfer used in the Multicast File Transfer Protocol (MFTP) described herein. StarBurst hereby records a grant by StarBurst Communications Corporation to permit the conditional free use of this patent to help facilitate adoption of MFTP by the Internet community

StarBurst hereby covenants that it will not assert its U.S. Patent 5,553,083 or its non-U.S. counterparts against any party that

implements MFTP or any derivative that includes MFTP, and operates in compliance with MFTP or such derivative, when MFTP or such derivative is employed in connection with any Internet Protocol.

This grant of rights will terminate with respect to any party that asserts a patent it owns, directly or indirectly, against StarBurst for StarBurstÆs implementation of and operation in compliance with MFTP or any derivative, as defined above. The termination will occur as of the date the patent is asserted against StarBurst.

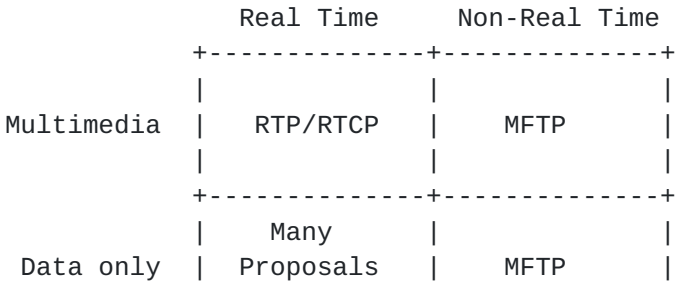
A written confirmation of this grant, and/or a license under any other StarBurst patent under StarBurstÆs then current terms, conditions, and royalty rates must be obtained by sending a request to:

Edward M. Durkin
StarBurst Communications Corp.
150 Baker Avenue
Concord, MA 01742

Table of Contents

## 1. Introduction

This document provides a description of the StarBurst Multicast File Transfer Protocol (MFTP). The MFTP protocol is designed to provide efficient, reliable transfer of data organized as files from one sender to multiple receivers. This protocol has been optimized for file delivery, rather than attempting to be a generalized reliable multicast transport layer that can handle both file delivery and streaming data.



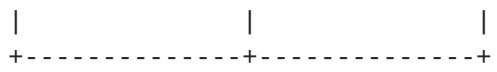


Figure 1-1 Multicast Applications and Protocols

Figure 1-1 shows a spectrum of multicast applications and the protocols serving them. StarBurst MFTP is designed to address the non-real time applications depicted in the right half of the diagram. Other reliable multicast protocols attempt to solve the non-real time and data only real time applications with one protocol.

This MFTP specialization brings a number of benefits, including:

1. Simplicity, which generally means increased reliability
2. Virtually no performance change over high delay networks such as satellite
3. Ability to be scaleable to thousands of recipients over one hop networks such as satellite with no intermediate relaying entities
4. Ability for the sender (Server) to have complete knowledge of the state of receivers (Clients)

Additionally, MFTP has flexibility so that optional features can be invoked to increase scalability beyond thousands, including:

1. **Aggregation and relaying of responses by the network infrastructure or other intermediate entity**
2. **Response suppression to reduce responses from members of a subnet similar to that used by IGMP as specified in RFC 1112**

Both of these techniques can be used to increase scalability by reducing Client message traffic to the Server.

Typical applications for MFTP include electronic software distribution, transmission of critical information to field offices, distribution of multimedia information to local servers, and replication of web servers to the edges of networks for improved performance. A significant new application is the ability to provide subscription based "push" information delivery to receivers who have signed up for a particular information service.

MFTP is designed to operate with networks that support multicast and broadcast data transport at the link layer such as LANs and SMDS, and with multicast IP router networks. Multicast user groups can be set up for either type network. When the network is router-based, data is delivered only to hosts in the multicast group based on multicast IP addresses. Each host supports RFC1112, Host Extensions for IP Multicasting.

When the network does not support multicast, data may be broadcast with multicast groups defined at the application layer at each MFTP host. This means that data is broadcast within the network but filtered at the MFTP host so that only members of the defined group receive the data. Data may also be sent in unicast mode to a single

receiver.

The MFTP Server (data sender) manages the multicast groups, initiates the file transfer, and controls the transfer operation. The MFTP

Client (data receiver) joins the multicast group, receives the data sent by the Server, and provides reception status to the Server when requested.

MFTP transmission is efficient because data is sent in a streaming mode. This means that the Server continuously sends data without waiting for responses from Clients. Acknowledgment traffic generated by the Clients is minimized because the Clients send responses only for Data Transmission Units (DTUs) that are not received. In addition, the status for a range of DTUs is aggregated in each response, further reducing the response traffic. At the end of a file transmission, the DTUs reported by the Clients as lost or received in error are retransmitted by the Server.

MFTP supports file checkpoint/restart so that an interrupted file transfer may be resumed without retransmitting data that has already been received by the Clients. Unlike other protocols that use all available bandwidth to transmit data, MFTP allows a maximum Server transmit rate to be specified. This allows files to be transmitted without stealing bandwidth needed by other applications.

MFTP consists of two protocol components - the Multicast Control Protocol (MCP) and the Multicast Data Protocol (MDP). MCP allows the Server to dynamically control the joining and leaving of Clients to multicast groups. MDP then handles the reliable transmission of data products to the Clients that have joined the multicast group.

The major features and characteristics of MFTP are summarized below.

Reliable data transfer via selective NAK mechanism
Uses multicast transmission, as defined in RFC 1112
Scaleable to thousands of recipients without intermediate entities
Scaleable to millions with intermediate entities to aggregate responses
Includes multicast group management and control
First level congestion control mechanism defined
Minimal performance change on high delay networks such as satellite
Additionally supports broadcast and unicast transmission
The maximum data rate of each transmission is specified
File checkpoint/restart is supported

## 2. Definitions

This section provides definitions for terms used in this document.

Aggregation: The process of collecting responses from Clients and combining them into one response to be relayed back to the Server. This reduces the back traffic to the Server and thus increases the number of Clients that can be served.

Announcement: The process of informing Clients that a data transfer is about to start. This involves sending a message to the "well known" multicast group address where Client hosts have joined and are listening for such messages. The message identifies the data product,

its size, name, etc. This message is retransmitted at regular intervals for a configured time period (e.g. 3 minutes). After the product announcement phase is completed, the Server begins the transmission of the product (see Product Delivery below).

ARS: Application Reference String. This is a variable length, case-sensitive string that the application provides to identify itself. It is used by MFTP to ensure that data is sent and received by like applications.

Authorized Client: A term used primarily for Closed Groups where the IP address of each Client that is allowed to receive the data is included in the Announce message. That is, the Client is "authorized" by the applications to receive the data if the Client finds its IP address in the Announce message. All other Clients are prohibited from receiving the data.

Block: A group of MFTP data messages. Clients report the receive status of data messages a block at a time.

Client: The receive function of the MFTP protocol. A single MFTP implementation may contain only a Client function, or it may contain both a Client and Server function (see Server below). Therefore, Client does not necessarily refer to the host itself.

Closed Group: A form of group management where the Server specifies the Clients that are allowed to participate in the data transfer. All other Clients are prohibited from receiving the data. Also refer to Open Limited Group and Open Unlimited Group.

Data Product: A file that is sent by the Server to Clients.

Done Message: A message sent from the Clients to the Server to indicate that the Client has received a complete data product.

DTU: Data Transfer Unit. A protocol message which contains the data that is being sent by the Server to the Clients.

Message Implosion: With a large number of Client hosts, all sending messages to the Server, it is possible for the Server to be overrun and messages to be lost. It is important for the protocol to minimize the number of messages that are being directed to any single host. An example of this is the negative-acknowledgment scheme that is used to obtain a Client's reception status and the way that the status for a range of DTUs is grouped together into a single response message.

MFTP: Multicast File Transfer Protocol. The protocol described in this document.

MTU: Maximum Transmission Unit. The largest unit of data, in bytes, that can be transmitted over the user's physical network.

Open Limited Group  A form of group management where the Server does

not specify which Clients may participate in the data transfer. Any Client that wishes to receive the data is required to register with the Server. The Server may limit the number of participants based on its own resources or some other criteria. Also refer to Closed Group and Open Unlimited Group.

Open Unlimited Group: A form of group management where the Server does not specify which Clients may participate in the data transfer. Clients that wish to receive the data are not required to register with the Server. Also refer to Closed Group and Open Limited Group.

Pass: For a file transmission, MFTP makes one "pass" through the file to transmit the file data. It then makes another "pass" through the file to retransmit data that was not received during the first pass. Additional passes may also be made to deliver the product successfully to all receiving hosts.

Private Address: The network address to which a Server transmits the data product. The Server specifies the Private Address in the Announce message for each product transfer. The private address is most relevant in multicast transmission because separate multicast addresses are used for product announcement and delivery. Only Clients that are authorized to receive a product actually join the multicast group defined by the private address.

Product Delivery: The process of transmitting the data product from a Server to Clients. The data is transmitted in messages called Data Transfer Units (DTUs). A group of DTUs is called a block. The transmission of the entire data product once is called a pass. No DTU retransmissions occur during a pass. Rather, additional passes are made when retransmissions are required. However, only the missed DTUs are retransmitted on each subsequent pass. This phase is always preceded by the Product Announcement phase (see Announcement above).

Public Address: The network address to which a Server transmits a product announcement. The public address is most relevant in multicast transmission because separate multicast addresses are used for product announcement and delivery. Any number of Servers may announce products to a single public address and any number of Clients may join the multicast group defined by the public address.

Registration: The process of a Client informing a Server that the Client intends to participate in the product transmission that is currently being announced. The Client does this by sending a Registration response message to the Server. This response is sent to the address specified by the "Response Address" parameter in the Announce message.

Relay: An entity that forwards messages to another destination after

receiving them. For example, an aggregator also acts as a relay by combining Client responses and then forwarding them to the Server or another aggregator. Relays can also be Clients that in turn act as Servers for another group that cannot be reached directly by the Server or for efficiency.

Response Address: An IP address (unicast or multicast) to which Clients send Registration, Status, and Done messages. The Response Address is specified in the Announce message. The ability to specify an address other than the Server's address allows an intermediate entity to perform aggregation/relaying of the responses.

Server: The transmit function of the MFTP protocol. A single MFTP implementation may contain only a Server function, or it may contain both a Server and Client function (see Client above). Therefore, Server does not necessarily refer to the host itself.

Transmission ID: A number that uniquely identifies an individual data product transmission that originates from a single Server. The concatenation of the Server's IP address and the Transmission ID uniquely identifies any MFTP transmission in the network. The Transmission ID is included in every MDP message sent by the Server and Client.

## 3. MFTP Architecture

MFTP operates above the User Datagram Protocol (UDP) layer of the TCP/IP protocol suite. As such, it uses the UDP Sockets interface.

MFTP defines both a send function (the Server) and a receive function (the Client). The Server only transmits data products and the Client only receives data products. An implementation may optionally include both functions, as shown in Figure 3-1 or may only have one. The checksum, defined in UDP, is the usual mechanism to determine if a datagram is corrupted. Some implementations may not include a UDP checksum; in these cases, an optional checksum is provided by MFTP.

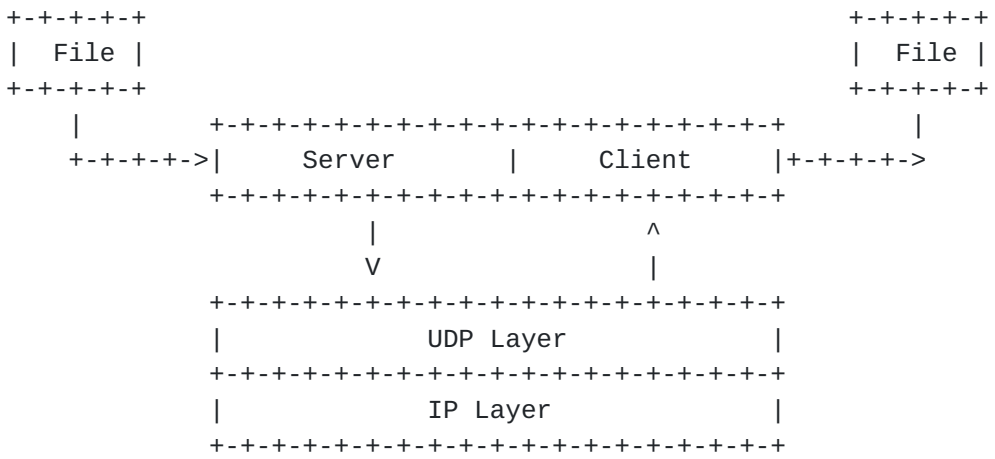



Figure 3.1 MFTP Data Architecture

File products can be transmitted directly from the file system by the Server and written directly to the file system by the Client. All implementations are required to support the MFTP Echo message and Echo Response message, which provide functions similar to "ping".

## Transmission Modes

Messages sent by the Server may be sent in either unicast, broadcast, or multicast mode. The Client must be able to receive Server messages in any of these modes. This allows flexibility in designing the Server to achieve network optimization goals. For example, unicast may be used to send an Abort message to a single Client when that is the only Client that is being aborted. All other Clients are spared from having to receive the message on the multicast group. An entire product transfer can occur in unicast mode when there is only a single Client receiver.

Broadcast mode may be used when the network does not support IP multicast. All Clients receive the messages, but messages that are not directed to the Client are filtered and discarded. This mode should normally be avoided because of its impact on the network.

The remainder of this document generally assumes multicast transmission, unless otherwise stated.

## Well-Known UDP Port

IANA-assignment of a "well-known" UDP port will be requested for MFTP. Certain MFTP messages must be sent to this port because it will be the only port number known both to the sender (Server) and the receivers (Clients). This includes the MCP messages, the Announce message and the Data message.

An MFTP station may include both the Server and Client function, both using the well-known port. To reduce the traffic on the well-known port, a Server may dynamically allocate a source "session" port to send a data product and to receive responses from the Clients. The Announce message is sent to the Client's well-known port. Client responses are returned to the source session port number. The Client may also allocate a session port to send messages, rather than using the well-known port.

The Server sends all messages to the Client's well-known port. The specified session port is in effect only for the current product delivery. This use of session ports is illustrated in Figure 3-2.

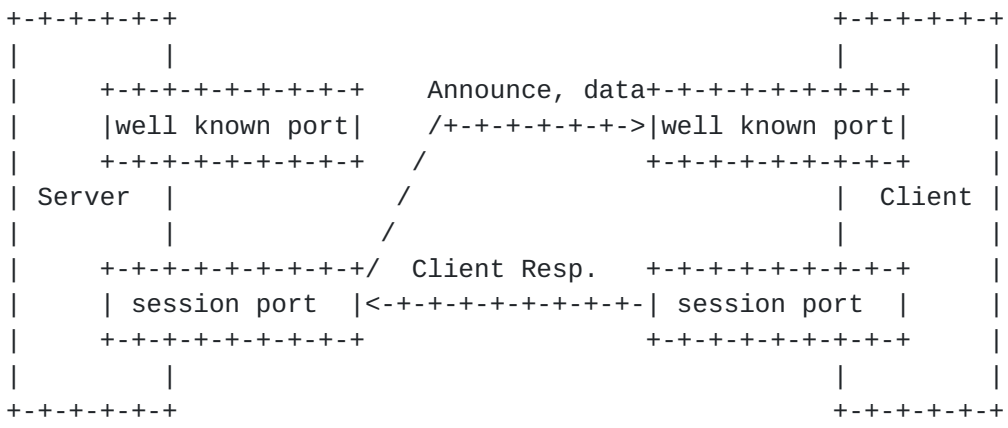



Figure 3-2 Session Ports

## 4. Scaling

By focusing on file based delivery rather than attempting to be a generalized reliable multicast transport layer, improvements in scaling without requiring intermediate aggregation points are achieved. It is known that the limitation for scaling in reliable multicast protocols is the NAK Implosion problem, where the administrative (NAK) back traffic from receiver (Client) to sender (Server) eventually overwhelms the sender as the number of receivers grows. The use of blocks in MFTP aggregates NAKs from each recipient so that one NAK with a bit map of the DTUs in the block can represent thousands of bad or dropped DTUs, depending on the actual number of bad or dropped DTUs in the block.

Although not required, MFTP recommends that the block size be tied to the MTU size so that the bit map in one NAK packet exactly fits the number of DTUs in a block. For example, if the MTU is 1500 bytes at the MAC layer (the maximum for Ethernet), the block size consists of over 11 thousand DTUs. Thus, a burst of thousands of dropped DTUs within a block results in only one NAK rather than thousands (see Section 11.2.1).

Even when the MTU at the IP layer is 576 bytes, the maximum guaranteed in TCP/IP networks with no fragmentation, the block size is over 3800 DTUs which still provides a high level of minimization of NAK traffic.

Experiments have been performed by StarBurst to emulate up to 10,000 Client receivers in one Closed group transmission and no significant performance degradation was observed. Obviously, scaling behavior will be affected by percent error rate and/or packet loss rate in the network, with the better the network characteristic the better the scaling. With MFTP, the Open Unlimited group model has scaling performance totally dependent on NAK traffic; however, for Closed and

Open Limited group models, all Client receivers Register and all Client receivers send Done messages, the latter acting as one ACK for the entire file. Thus, in these models, all receivers send one message at the beginning and at the ending of the transmission. This

traffic in fact becomes the scaling limitation in the Closed and Open Limited group models.

Products using MFTP in essentially the same form as described in this document have been operating in private networks since mid 1995. At the time of this writing, several installations are operating in a production environment with over 1000 simultaneous receivers and one installation expects to be operating with about 9000 receivers in a group later in 1997.

### 4.1 Scaling Extensions

In addition to the minimization of administrative back traffic described above, provision is made in MFTP to extend scaling to hundreds of thousands and perhaps millions of simultaneous receivers depending on the network characteristics by using network aggregation entities to further consolidate administrative back traffic. Additionally, provision is made for NAK suppression on subnets, similar to the method used by IGMP as specified in RFC 1112.

## 5. Performance

MFTP is a rate based protocol with a transmission rate defined by the sender (Server). By design, it does not attempt to provide error correction in real time but takes advantage of the non-real time nature of file transmission. Thus, transmission remains continuous at the set transmission rate through the total transmission of the file in the first pass, with corrections made in subsequent passes. This makes MFTP very insensitive to delay such as occurs over satellite networks and also makes it tolerant of asymmetric data channels.

Performance is affected as the number of simultaneous receivers grows, as this increases the number of bad or dropped packets in aggregate for the group resulting in more retransmissions. However, in practice there is often high correlation in bad/dropped packets among receivers. For example, all receivers downstream from a congested node will experience the same packet drops caused by that node. Since the retransmissions are sent to the same multicast group, they will all get the same correction simultaneously on a subsequent pass.

Protocol overhead is modest, both in header size and in the amount of processing needed. Thus, MFTP can be expected to be able to exceed the performance of FTP even in a point to point application.

## 6. Group Management

MFTP uses the push method for distributing data products. That is, at a time determined by the Server user application, a data product transmission is "announced" to the Client population. This

announcement consists of transmitting Announce messages for a period of time, followed by the data product itself. The Announce message identifies the data product that will be transmitted, its size, and other parameters documented herein.

MFTP uses two separate multicast groups, called the public group and the private group. The Server announces products on its public group address and Clients join the address to receive announcements. The product data is transmitted on the private group address. That is, the public group address is where any Client may join and listen for announcements made by any Server that is sending to the public address. However, only Clients that are authorized by the application to receive a particular transmission actually join the private address.

The public/private group architecture is shown in Figure 6-1. Clients 1 - 3 are joined to and listening on the public group. Only Client 3 wants to receive the product that has been announced and has joined the private group to receive it.

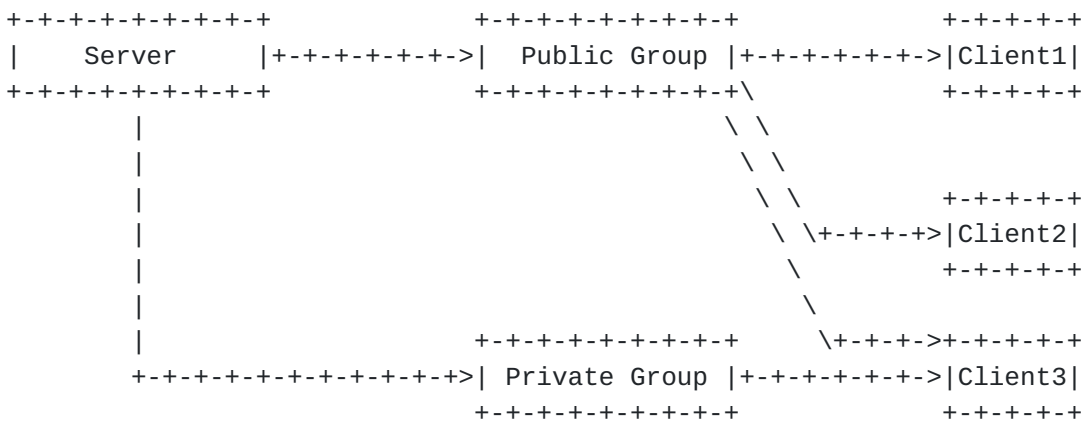



Figure 6-1 Group Architecture

Group management deals with how Clients learn about and are joined to the public and private groups. The attributes of the public group are discussed first, with the assumption that Clients are already joined to it. Once a Client is joined to the public group, it learns about the private group in the Announce message. Finally, the method used to join Clients to the public group is discussed.

## 6.1 Joining Clients to the Private Group

Use of the public group is completely flexible in that there may be one for each Server or multiple Servers may transmit product announcements to a single group address. It is recommended that the number of public groups be kept to a minimum, even one. This allows the use of a single public group for Clients to listen for announcements from many Servers without requiring a separate multicast group for each one.

In order to direct the product data to only those Clients that

actually want to receive it, the product announcement message includes a field that specifies a separate private multicast address. The Clients then dynamically join the private address and the Server transmits the product data on that address. Note that while joined to

the private group the Client continues to be a member of and listens for product announcements on the public address. This allows the Client to receive multiple data products simultaneously, if it so desires.

Normally, the Clients leave the private address at the end of the product transfer and continue to listen for product announcements on the public address. However, if the Server is going to transmit multiple data products, it may set an option in the product announcement that directs the Clients to not leave the private address. This reduces the overhead associated with joining and leaving multicast groups. By also announcing the next product transmission on the public group, Clients that did not hear or did not participate in the previous transfer may be added to the private group.

If the Server is supplied with more than one private address, it may arbitrarily use any one of the addresses to transmit a data product. Multiple addresses would typically be used to send multiple products to different groups at the same time by a single Server.

MFTP includes a unique transmission identifier in each message so that the Client can always distinguish messages that are associated with an individual data transmission.

### 6.2 Joining Clients to the Public Group

To discuss how Clients are joined to the public address, there are two models that are considered, Closed and Open.

In the Closed model, the Server knows in advance which Clients are authorized to receive a data transmission, and the number of Clients is relatively small (e.g. thousands of Clients). This allows the Server to tightly control group membership.

The public address may be known and configured in advance at the Clients or the Server may use the MFTP Multicast Control Protocol to direct Clients to join the public group. This type of group management is called Closed Groups since the Server restricts the membership of the group. This control occurs at two levels - (1) either by configuration or by use of MCP, the members of the group are restricted and (2) the product announcement itself includes a list of Clients that are authorized to participate.

Upon receiving the Announce message, the Clients designated to join the group register with the Server by sending a Registration message. All other Clients are administratively told not to join the group and thus do not receive the transmission. This is a Server-centric mode of operation that allows the Server to tightly control which Clients receive the data and also minimizes the number of Registration

messages arriving at the Server. It also allows the Server to keep detailed statistics on each Client. This mode is most useful when the data product has a value associated with it that prevents it from being freely distributed.

In the Open models, the receiving Clients are not known to the Server. This means that the Server is not able to configure the multicast groups via MCP. A Client must be able to obtain the public group address of Servers that it is interested in, perhaps by using the MCP Group Query message or by using one of the external protocols defined in the mmusic working group.

Since the MFTP server is not determining those Clients that join its session, this type of group management is called "Open Group". There are two variations of Open Groups, called "Limited" and "Unlimited", as discussed below.

The Limited mode allows the Server to announce a transmission without specifying the Clients that are authorized to receive the data. This may be because the Server does not know in advance which Clients may want to receive the data, or the number of potential Clients may be very large.

This type of group allows any Client to request participation in the data delivery, but the Server limits the number of actual participants based on some criteria determined by the application. As Clients register to receive the data, the Server learns the identity of each Client. Therefore, like the Closed Group, the Server is able to keep detailed statistics on each Client. In fact, after the Announce/Registration phase, protocol operation is identical to the Closed Group. This mode is most useful when it is permissible to freely distribute the product, but the sender wants to know who is receiving it.

The Unlimited mode allows any Client to participate in the data delivery and the Server does not limit the number of participants. To achieve this, certain types of Client responses are waived to prevent message implosion at the Server. This includes the Registration and Done messages. The only Client response that is retained is the Status Response message. Since Clients send this message only for DTUs that are not received, there will not necessarily be a response from every Client. The Server is unable to identify each receiving Client when this mode is used, so it is most useful when the data product can be freely distributed and the sender does not care to know who is receiving the product.

## 7. Response Address

The Server specifies the response address parameters in the Announce message. Each Client then sends all response messages to the specified response address (regardless of whether the Server message is received in unicast or multicast mode).

A different response address than the Server may be used by an aggregating entity in the network to collect responses from clients and reduce implosion traffic to the Server. This increases the scalability, i.e., the number of clients supported. The use of a multicast response address with a suitable TTL may be used for

aggregation and is required for response suppression (see Section 6). The response parameters include the following:

Response IP Address (may be different than Server's IP address)
Response port number (may be different than Server's port number)
Server IP address
Server port number

The Response IP Address may be either a unicast or multicast address. The normal mode is that the Response IP Address and port number be that of the Server if optional aggregation and/or suppression is not used.

The Server IP address and port number are echoed in the Client's response. This is the address and port where the response must eventually be sent by an intermediate relay point, if present.

## 8. Response Suppression

Response message suppression is an optional Client function that eliminates the sending of redundant status messages from a single subnet, such as a LAN. That is, congestion that occurs upstream from the subnet will produce an identical set of dropped messages at each Client on the subnet. The result is all Clients send identical status responses to a Status Request message. This can be avoided by having the Clients listen to each other's response and suppress duplicate responses.

Response message suppression is used only with the sending of the NAK Response message. Other Client responses are sent in the normal manner described in this document.

The NAK Response message is sent to the multicast Response Address as specified in the Announce message. Each Client must be joined to this address in order to hear the response of other Clients on the subnet.

When a Status Request message is received by the Client, and the request causes a NAK response message to be built, the Client does not immediately send the message. Rather, a delay timer is started. Each Client's delay timer is a different, randomly-chosen value between 0 and n seconds, where n is configurable with a default value of 1 second. The Client uses a pseudo-random number generator to compute its delay timer value, using the Client's own host IP address as part of the seed to reduce the chance of multiple Clients generating the same delay.

While waiting for its own delay timer to expire, each Client receives and examines the status responses sent by other Clients. If the pass

number, block number, and event hash fields all match the Client's unsent message, the Client's unsent message is discarded and the delay timer is canceled. Otherwise, the Client continues to listen to and examine additional responses. If no matching message is received

before the Client's delay timer expires, the Client transmits its own response message. Any response messages received after the Client's message has been transmitted are ignored. If a new Status Request message is received from the Server while the delay timer is running, the Client immediately sends its current response message, cancels the timer, and processes the new Status Request as described above.

The response suppression function is applied at the subnet level. Since all Clients participating in a product transfer will join to the multicast Response Address, messages that leave a subnet will also arrive at all other Clients joined to the group and will be processed as described above. When message aggregation is performed by the network infrastructure (see Message Aggregation above), the network can prevent responses from being propagated anywhere except back to the Server.

## 9. Configuration

The following configuration parameters affect the operation of the protocol:

- Announce Time Limit
- Fast Announce Option
- Announce Interval
- Announce Persistence
- Status Interval
- Status Retry Timer
- Status Retry Count
- Delivery Time Limit
- Completion Interval
- Transmit Rate
- Register Retry Timer
- Register Retry Count
- Response Back-off Timer
- Response TTL

Some parameters may apply only to product transmission, only to product reception, or to both. The application of each parameter is noted in the following descriptions. Also refer to the description of the protocol and message formats below for additional information.

### 9.1 Announce Time Limit

This parameter sets the maximum time, in seconds, that MFTP will use for the announce phase of a product transmission. The announce phase will last for this duration unless the Fast Announce Option (see below) is used.

### 9.2 Fast Announce Option

This parameter only applies to Closed Groups. The Server may shorten the duration of the Announcement immediately after all members of the group register. This is the normal mode of operation in the Closed Group model.

### 9.3 Announce Intervals

This parameter sets the interval, in seconds, between successive transmissions of the Announce message during the product announce phase.

### 9.4 Announce Persistence

This parameter specifies whether the Announce message should continue to be sent during the product delivery phase. This could be desirable in the Open Unlimited group model where the Announce can act as an advertisement" for what is being transmitted.

### 9.5 Status Interval

This parameter specifies the interval, in seconds, between Status Request messages sent by the Server to the Clients. It is used to establish the transmit block size. A default block size will be computed, based on the MTU. This block size will maximize the amount of NAK information that can be sent in a single status response message by the Client. The Server sends the status request message at the end of each transmitted block. However, for a large MTU (and therefore a corresponding large block size), a relatively long time may elapse before there is any status information that can be returned to the application about the NAK status of individual Clients. The application may set the Status Interval parameter to the time interval that it wants the Server to request NAK status from the Clients. The block size is adjusted for the requested interval so that a block of DTUs is sent in the interval and then NAK status is requested by the Server.

### 9.6 Status Retry Timer

This parameter specifies the time duration, in seconds, that the Server will wait for Client responses after a Status Request message is sent.

This timer is not used for Status Request messages sent during a pass because the Server does not stop and wait for responses. However, at the end of the pass, the Server cannot begin the next pass if no responses (status response or Done message) have been received for the current pass. In this case, the Server sends a Status Request

message, starts the Status Retry timer, and waits for any responses. When the timer expires, the Server begins the next pass if any status response messages were received. If there is no response, the Server sends another Status Request message and restarts the retry timer.

When there are no responses, this procedure continues up to the retry limit set by the Status Retry Count (see below) or until the Delivery Time Limit is reached, whichever occurs first.

### 9.7 Status Retry Count

This parameter specifies the maximum number of times that the Server will retransmit a Status Request message when no response is received from any Client. Refer to Status Retry Timer above to see how it is used.

### 9.8 Delivery Time Limit

This parameter sets the maximum time for the product delivery phase of a file transmission. The parameter is an integer that represents a percentage, with a minimum value of 100% (i.e. value =100). MFTP multiplies the integer by the optimal time that it takes to make one pass through the transmit file. The result is the maximum time devoted to the product delivery phase, in seconds.

### 9.9 Completion Interval

This parameter sets the interval, in seconds, between successive transmissions of the Completion message during the product transfer/completion phase.

### 9.10 Transmit Rate

This parameter specifies the bit rate at which a data product will be transmitted by the Server. The transmit rate may fall below this value due to other processing demands on the Server, but the rate shall never exceed the configured value. The transmit rate applies to all bits sent by the Server in a DTU, including all protocol headers. The algorithm to be used to control the transmit rate is unspecified.

### 9.11 Register Retry Timer

This parameter specifies the frequency, in seconds, that the Client will resend a Registration message once data transfer phase begins. During the Announce phase, the Registration message is resent every time that an Announce message is received that does not confirm the previous Registration transmission. Hence, the stimulus for Registration retransmission is the received Announce message and the rate of retransmission is identical to that of the Announce message (see Announce Interval parameter above).

### 9.12 Register Retry Count

This parameter specifies the maximum number of times that the

Client will retransmit a Registration message during the data delivery phase in order to solicit a confirmation from the Server. Refer to Register Retry Timer above to see how it is used.

### 9.13 Response Back-off Timer

This parameter specifies the maximum length of the back-off timer when suppression is invoked.

### 9.14 Response TTL

This parameter specifies the TTL of the response message from the client. It should normally be set to the maximum for the network.

## 10. Multicast Control Protocol

This section describes the Multicast Control Protocol (MCP). Configuration parameters and message formats are described in separate sections.

The purpose of MCP is to allow a Server to control the dynamic joining and leaving of remote hosts to multicast groups. MCP defines the following message types:

| | |
|---|---|
| Join Group | transmitted by the Server to dynamically join Clients to multicast groups. This message is normally unicast to an individual Client, but may be multicast to an existing multicast group. |
| Group Query | transmitted by the Client to query a Server for its current public multicast group address. This message is unicast to the Server. |
| Leave Group | transmitted by the Server to dynamically cause Clients to leave a multicast group. This message may be unicast to an individual Client or multicast to an existing multicast group. |
| Echo | transmitted by any MFTP station to determine if a remote host is reachable and running MFTP. This message may be unicast to an individual host, or multicast to an existing multicast group. |
| Echo Response | transmitted as a response to a received Echo message. This message is unicast to the originating host. |

MCP provides for creating multicast groups, dissolving multicast groups, and determining group membership. Each of these is discussed below. Also refer to the Message Format section for additional information.

### 10.1 Creating Multicast Groups (Join Group and Group Query)

Multicast group formation may be initiated by both the Server and the Client. If a Client knows the IP address of a Server, but not it's public address where it is announcing product transmissions, it may send the Group Query message to the Server. The Server then sends

a Join Group message to the Client to get it to join the Server's public multicast address.

The Join Group message may be sent by a Server at any time to dynamically direct one or more Clients to join a specified multicast group. If the Client(s) are not already a member of a multicast group, the message is sent in unicast or broadcast mode. Otherwise, the message may be sent to an existing multicast group to direct Clients joined to that group to join an additional group.

The Join Group message identifies the application type that is sending the message and the type of data that will be sent by the application. If the Client has an application layer user of the same type that wants to receive the indicated data products from this Server, it joins the multicast group.

There is no response to the Join Group message. However, the Echo message (see below) may be used to determine which Clients have successfully joined the group. Since the Echo message can be sent to the new group address, only those Clients that have actually joined the new group will respond to the Echo message.

### 10.2 Dissolving Multicast Groups (Leave Group)

The Leave Group message is used to direct one or more Clients to leave a specified multicast group. The message may be sent to the existing multicast group to direct its members to leave the group, or the message may be sent in unicast or broadcast modes.

There is no response to the Leave Group message. However, the Echo message (see below) may be used to determine which Clients have successfully left the group. If the Echo message is sent to the group address, there should be no response from those Clients that have left the group.

### 10.3 Determining Group Membership (Echo and Echo Response)

The Echo message may be sent at any time to determine which Clients have joined a multicast group. The receiving host responds with the Echo Response message. The response is sent if MFTP is running on the target host, without regard to whether there are any application layer users of MFTP. When users do exist, the receipt of the Echo message and the transmission of the Echo Response in no way affects those users.

The Echo message may be multicast, broadcast, or unicast. If the message is unicast, then the receiving host always responds to the message. If the message is multicast or broadcast, it may contain a list of the hosts that should respond to the message. In this case,

the receiving host responds only if it is specified in the host list. If there is no host list, the receiving host always responds.

The Echo message may be sent by either a Server or Client. It is also

useful for network testing, for example, determine if hosts are reachable, or to calculate response times.

## 11. Multicast Data Protocol

This section describes the Multicast Data Protocol (MDP). Configuration parameters and message formats are described in separate sections.

The purpose of MDP is to transfer the product data from the Server to the Clients in a reliable and efficient manner. MDP defines the following message types:

| | |
|---|---|
| Announce | transmitted by the Server to announce an impending product transfer. This message is sent to the Public multicast address. |
| Registration | transmitted by the Client as a response to the Announce message. This message is sent to the "Response IP Address" that is specified in the Announce message |
| Data Transfer Unit(DTU) | transmitted by the Server to deliver a data product. There is no explicit response to this message. This message is sent to the Private multicast address. |
| Status Request | transmitted by the Server to solicit status from Clients. This message is sent to the Private multicast address. |
| Status Response/ACK | transmitted by the Client in response to a Status Request message to acknowledge correct reception of a block of data messages. Normally, the Server requests only the NAK form of the status response to minimize responses from Clients. This message is sent to the Response IP Address. |
| Status Response/NAK | transmitted by the Client in response to a Status Request message to request the retransmission of data messages within a block. This is the normal response type requested by the Server and is the message that the generic term "status response" refers to in the following protocol discussion, unless otherwise noted. This message is sent to the Response IP Address (see Definition, Section 2). |

| | |
|---|---|
| Done | transmitted by the Client when it has received all of a product transfer successfully. This message is sent to the Response IP Address. |
| Completion | transmitted by the Server to confirm receipt of a Done message from one or more Clients. This message is sent to the Private multicast address. |
| Abort | transmitted by the Server to indicate that one or more Clients should quit participating in the current product transfer, or to indicate the premature termination of the current product transfer to all Clients. There is no response. This message is sent to the Private multicast address. |
| Quit | sent by the Client to indicate it is no longer participating in the current product transfer. There is no response. This message is sent to the Response IP Address. |

MDP consists of three general phases - Product Announcement, Product Delivery, and Product Completion. Each of these is discussed below. Also refer to the Configuration and Message Format sections for additional information.

### 11.1 Product Announcement

Product Announcement is the process of letting the Client population know that a product is about to be transmitted. The Server transmits the Announce message for this purpose. If multicast is being used, the message is sent on the public "well known" address. Otherwise, the message is sent in unicast or broadcast mode. The message identifies the product name, type, and other information about the transfer, including whether it is Closed, Open Limited or Open Unlimited (see Group Management above).

The Announce message is sent periodically during the Announce phase. The transmission interval is specified by the Announce Interval parameter, and the duration of the Announce phase is specified by the Announce Time Limit parameter.

The Announce phase serves several purposes. First, it helps to ensure that the message is actually received by the Clients since it is transmitted more than once. Second, subsequent transmissions of the message serve to confirm receipt of Registration messages received

from Clients by the setting of a flag in the message. Third, the announce duration allows Clients that are experiencing problems (e.g. insufficient disk space) to resolve those problems and still register for the product transfer.

When the first Announce message is sent, the Server starts the Announce Time Limit timer and the Announce Interval timer. When the interval timer expires, the Server resends the message unless one of the following conditions is true:

1. **The Group type is Closed and all Clients in the Host List (see below) have registered. Actually, this behavior can be modified** with the Fast Announce Option (see Configuration section). Note that after the last Client registers, there must be at least one additional transmission of the Announce message to confirm the registration. Also, the Server may introduce a delay to ensure that the last Client has actually joined the Private address before the data transfer phase is started. In any case, data missed by such Clients is recovered by subsequent data transmissions of the Server. No delay is currently specified by this protocol.

2. The Announce type is Open Limited and the maximum number of Clients that can be handled have registered.

3. The Announce Time Limit timer has expired. If no Clients have registered when this timer expires, then the product transmission is canceled.

The Announce message may contain a Host List. It is included for a Closed Group in order to identify those Clients that are directed to participate in the product transfer. As Client Registration messages are received, a flag is set in subsequent transmissions of the Announce message to confirm receipt of the Registration for each Client. The Host List is not initially included in the Announce message for an Open Limited group. However, it is included in subsequent messages to confirm those Clients that have registered. The Host List is never included in the Announce message for an Open Unlimited group.

Clients specified in a Closed group announcement respond with a Registration message whether they are going to participate in the transfer or not. If the Client is not going to participate, a reason is provided in the message. This allows the Server to track problems in the group.

A Server may receive more than one response from a Client during the same product announcement. The last response received should be considered the Client's "official" response. For example, the Client may initially decline the transfer due to insufficient disk space. The announce duration may allow the Client operator time to resolve the problem. If the Client operator frees up disk space, the Client may then respond that it will be participating in the transfer.

The Server may receive a Quit message from the Client after the Client has registered. After receipt of the sent message, the Server should not consider the Client a participant in the product transfer.

The logic flow diagram below illustrates the Server's product

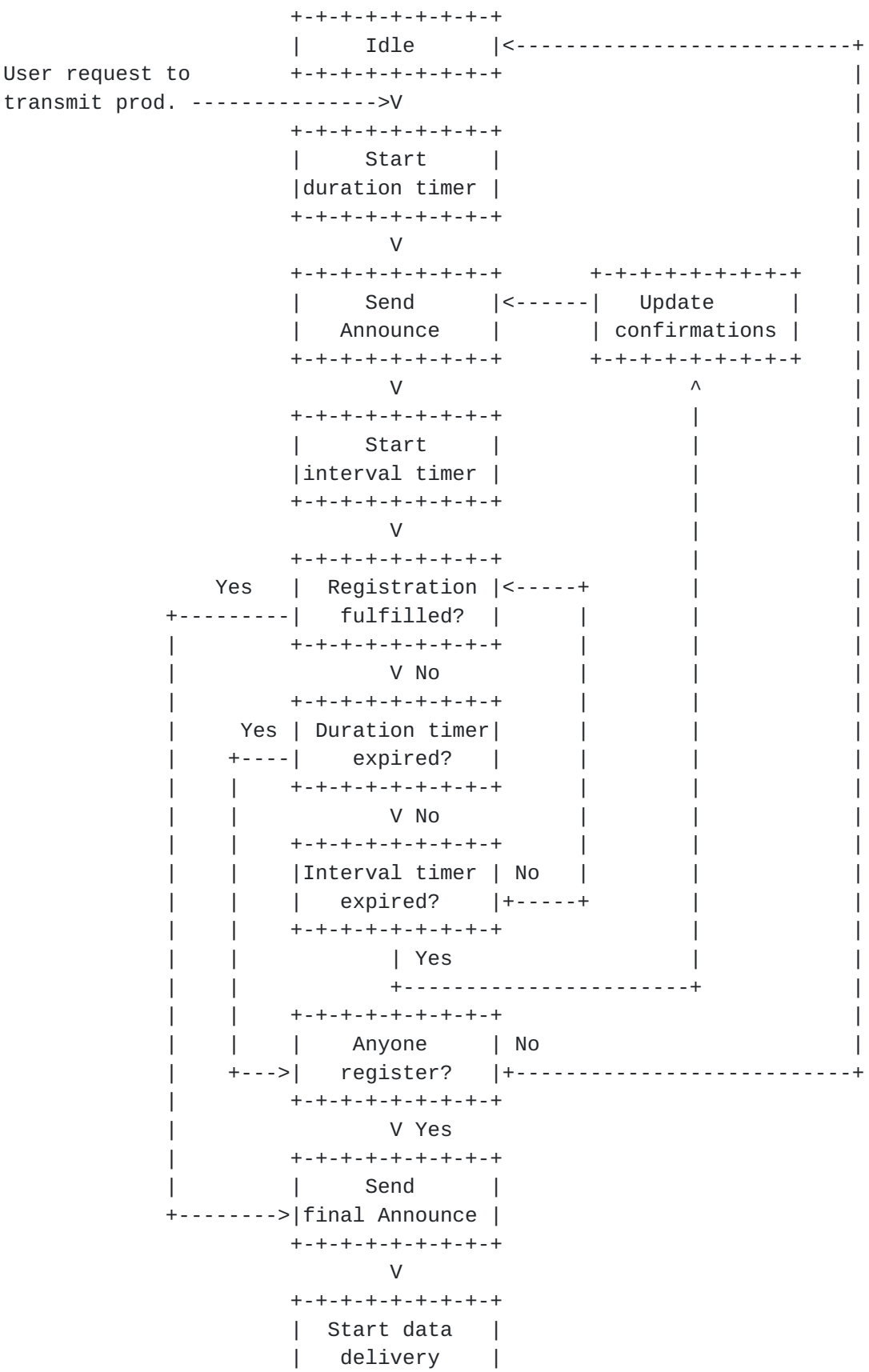

Idle
User request to transmit prod.
Start duration timer
Send Announce
Update confirmations
Start interval timer
Yes
Registration fulfilled?
No
Yes
Duration timer expired?
No
Interval timer expired?
No
Yes
Anyone register?
No
Yes
Send final Announce
Start data delivery

Figure 11-1
Server Product Announcement Phase

announcement phase. "Registration fulfilled?" refers to whether all Clients in a Closed group have registered, or whether the maximum number of Clients in an Open Limited group have registered (it does not apply at all to an Open Unlimited group). "Update confirmations" refers to the setting of the registration-confirmed flag in the Announce message to confirm Client registrations. This step also would not apply to Open Unlimited groups.

When a Client receives an Announce message, it examines the product information contained in the message and compares it with the requirements set by its application layer user(s). That is, it is expected that the application layer will designate the Servers that it is interested in receiving from and the types of data products that it wants to receive. Based on this information, the Client decides whether or not to receive the announced product.

If the Announcement is for Open Limited or Open Unlimited Groups and the Client does not want to receive the product, then the Client sends no response and discards the message. If the Announcement is for Closed Groups and the Client does not want to, or is not able to receive the product, then it still must send a Registration message stating the reason it is declining the transfer.

If a Client wants to participate in a transfer, its action is described separately for each group management type below. Note that joining and leaving the multicast group in the discussion below refers to IGMP operations and not MCP.

### 11.1.1 Closed Group

If the Client's IP address is not contained in the Host List of the Announce message, the Client sends no message and is administratively prohibited from participating in the transfer. Otherwise, it joins the data delivery multicast address (i.e. Private address) contained in the Announce message and sends a Registration message to the Response Address.

The Client's registration must be confirmed by the Server before it is allowed to receive the data product. The confirmation is indicated by a flag set in a subsequent Announce message, as discussed above. If the confirmation is received, the Client waits for and receives the product data. If confirmation is not received in the next Announce, the Client resends its Registration message.

If the announce duration (specified in Announce message, see Messages below), expires or the Client begins to receive product data before confirmation is received, it implies that the Server did not receive the Registration message before the start of the data delivery phase.

The Client discards the received data and resends its Registration message at the Register Retry Timer interval. The number of times that the Client resends its Registration message is limited by the Register Retry Count. If no confirmation is received before the retry limit is

reached, the Client returns to idle state and makes no further attempts to register for the current product transfer. Note that the Client also returns to idle state if a Completion or Abort message is received while the Client is still trying to register with the Server.

### 11.1.2 Open Limited Group

The Client joins the data delivery address contained in the Announce message and sends a Registration message to the Response Address. Subsequent operation is identical to Closed Groups in that the Client must receive confirmation of its registration before it is allowed to receive the product data.

### 11.1.3 Open Unlimited Group

The Client joins the data delivery address contained in the Announce message but sends no Registration message to the Server. The Client receives the data as sent by the Server on the data delivery address.

A Client may receive an Abort message from the Server at any time during the product registration. The Client returns to idle state and is not allowed to participate in the current product transfer. When returning to idle state, the Client normally leaves the data delivery group.

However, the Client would remain a member of the group if:

1. The Server specified in the Announce message that Clients should not leave the data delivery group (see Announce message below).

**2. Other transmissions being received by this Client are using the same data delivery group address.**

3. The private address is the same as the public address.

## 11.2 Product Delivery

The Product Delivery phase occurs after the Product Announcement phase. The file product is transmitted by the Server to the Clients.

### 11.2.1 File Delivery - Server

DTUs are transmitted to the private multicast address. The Server logically divides the file into one or more parts, called blocks. Each block is further divided into the data size that can be transmitted in a single DTU. All blocks and DTUs are a fixed size, except the last DTU which may be shorter. The manner in which the block size is determined is discussed further below. This data organization is

illustrated in Figure 11-2.

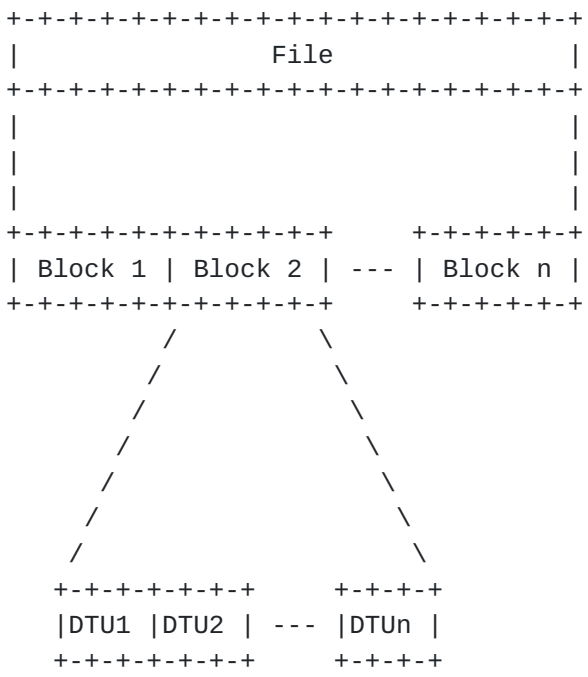



Figure 11-2
Organization of Transmitted Data

The Server transmits the file data in "passes". That is, the entire file is sent initially (i.e. pass 1). If any DTU retransmissions are required, the Server then makes another pass (i.e. pass 2) through the file, but sends only those DTUs that were reported as missed by the Clients. Additional passes may be required to successfully deliver all DTUs to all Clients.

The pass, block, and DTU number of each file fragment is indicated in the DTU header (see Message Formats below). The block and DTU number assignment is static for a given MTU size such that the retransmission of any file fragment will indicate the same block and DTU number. This allows the receiver to properly insert each fragment into the file regardless of when the fragment is received.

The Server uses the Status Request message to query the Clients for their receive status. The Clients send a Status Response message if they need to request the retransmission of any DTUs. Otherwise, a Client sends no Status Response to the request. Although Client status may be requested by the Server at any time, it is normally requested at the end of each block transmission. The response message contains a bit map where each bit represents the receive status of an individual DTU within the block. Hence, the status of an entire block is contained within a single response message.

The Server does not stop at the end of each block and wait for a response to the Status Request message nor does it immediately resend requested DTUs. Rather, the Server simply receives and processes the responses in order to schedule which DTUs need to be resent on the next pass. This makes the protocol insensitive (unless the data

product sent is very short) to network delay that can be excessive in some networks, e.g. satellite. Note that the Status Request message may be sent at any time rather than at block boundaries, if information is desired about transmissions before the end of the block.

After the last block of a pass has been transmitted, the Server sends a Status Request message for that block and immediately starts the next pass if there are DTUs that need to be resent. However, if no responses have been received for any previous block in the file, the Server stops and waits for responses. The length of time that the Server waits for a response is specified by the Status Delay parameter (see Configuration above). Note that the Status Request message may specify that status is requested for a single block or for a range of blocks. When the Server sends the Status Request message because no responses have been received, it specifies all blocks in the file. If a Client has not responded because it missed a previous individual block message, this ensures that such responses are solicited.

Each Status Request message specifies the current transmit pass number and the block for which status is being requested. These values are echoed by the Client in its response. The Server uses the following rules for processing Client responses.

1. If the response pass number is equal to the current transmit pass number, accept the response and schedule the requested DTUs for retransmission.

2. If the response pass number is not equal to the current transmit pass number, examine the block number and proceed as follows:

   a. If the response block number is greater than the current transmit block number, accept the response and schedule the requested DTUs for retransmission.

   b. Else, schedule for retransmission only those DTUs that have not already been retransmitted on this pass.

The block size is a function of the Status Interval and Transmit Rate parameters. That is, the application layer user specifies the frequency at which receive status should be obtained from the Clients. The Server computes how many DTUs it can send at the Transmit Rate in the Status Interval and establishes that as the block size. The block size is computed with the following formula:

block size = Status Interval x (Transmit Rate/(8 x (MTU - protocol headers)))

There is an upper limit to the block size that the application may set per Transmit Rate, as determined by the maximum DTU bit map size that the MTU will allow.

The default block size (used when application does not specify a Status Interval) maximizes the amount of status information that can be returned in a single message according to the MTU size.

Retransmissions may continue until all Clients have received the entire file or until the Delivery Time Limit duration has expired or until the Status Retry Limit has been exceeded. If the timer expires, the Server sends an Abort message and terminates the product delivery unless it is an Open Unlimited group, at which time a Completion message is sent. It is required to send several Abort messages to ensure that the Client receives at least one message since there is no confirmation message from the Client.

The logic flow diagram in Figure 11-3 illustrates the Server's Product Delivery phase. The completion processing is explained in a following section.

If a file is being restarted by the Server, the Server must first determine what part of the file has not yet been received by the Clients. There are many possible ways this can be done and no specific way is specified by the protocol. For example, the Server may locally store the state at the end of an aborted transfer and restore it for a file restart. Alternatively, the Server may want to query one or more Clients to determine what parts of the file still need to be transmitted. Once this initial restart state is determined, subsequent Server operation is identical to any retransmission pass (i.e. the Server sends only those DTUs necessary to complete the file transfer). File restart is allowed for Closed and Open Limited groups, but not for Open Unlimited groups.

### 11.2.2 File Delivery - Client

As DTUs are received by the Client, the Client must examine the block and DTU number in order to store the data in its proper place in the file. As discussed above, a bit map of each block is maintained that indicates whether a DTU has yet been received. If a duplicate DTU is received, it is discarded.

When a Status Request message is received from the Server, the Client immediately responds with a Status Response message for the indicated block. File delivery continues until all DTUs in the file have been received or until the receive timer expires. The receive timer value is specified by the Server in the Announce message. If the timer expires, the Client sends a Quit message and returns to idle state. It is required to send several Quit messages to ensure that the Server receives at least one message since there is no confirmation message from the Server.

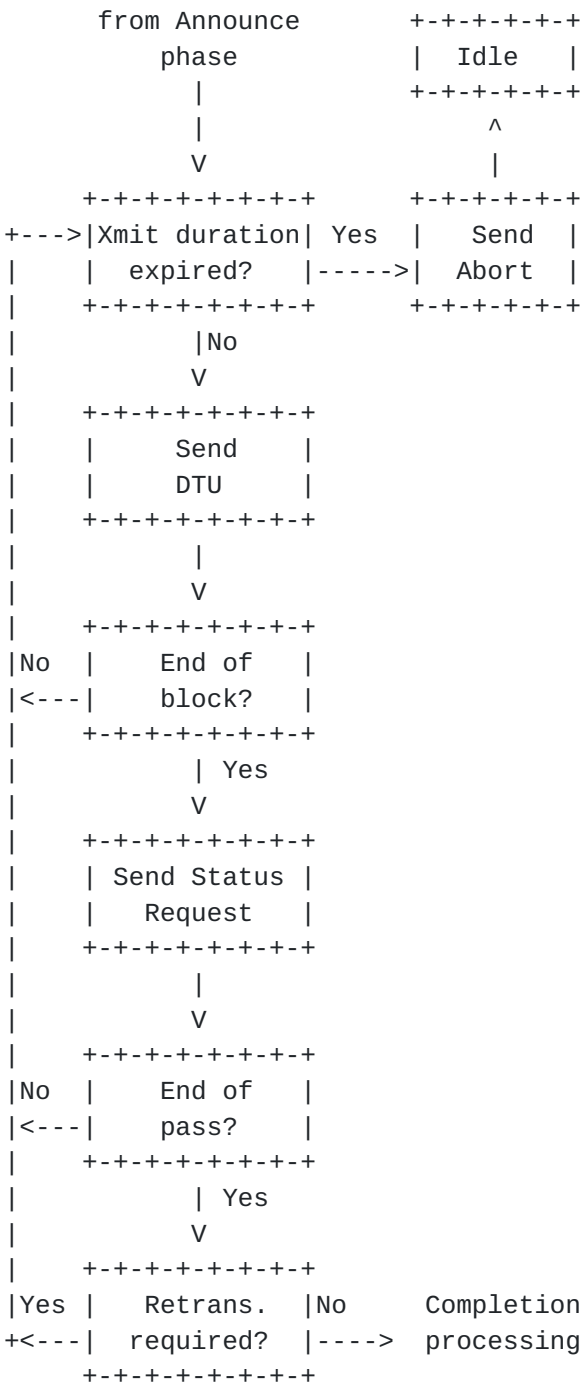



Figure 11-3 Server Product Delivery

The logic flow diagram in Figure 11-4 illustrates the Client's receive data processing. The completion processing is explained in a following section.

## 11.3 Product Completion

Product Completion is the process of terminating a product transfer. File product completion for the Client is discussed first, since it is the Client that normally initiates the completion procedure.

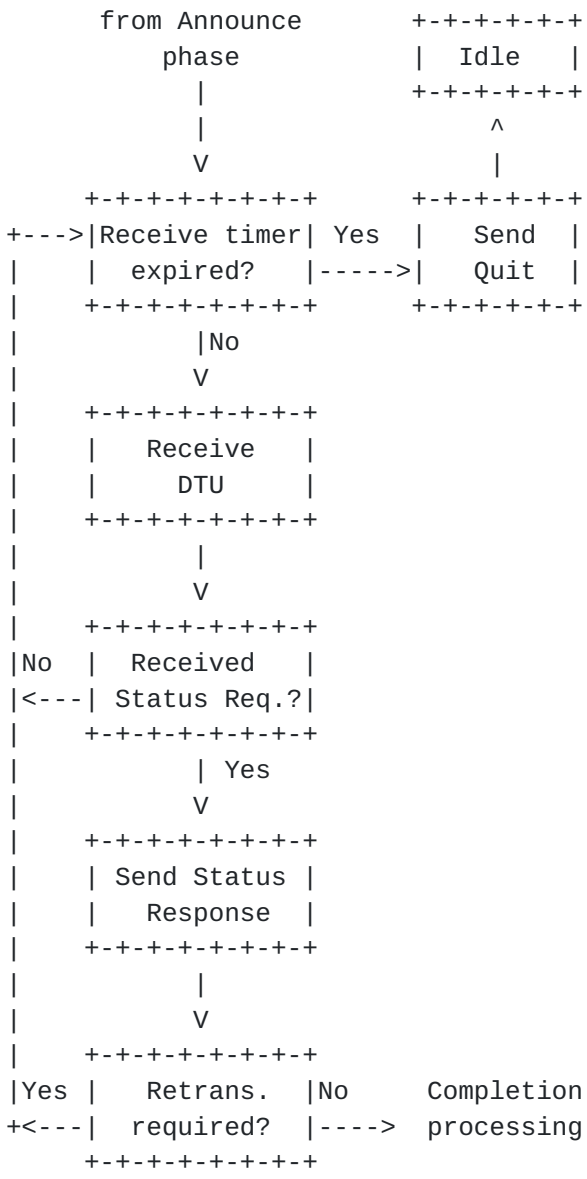



Figure 11-4. Client Product Reception

### 11.3.1 File Product Completion - Client

As each DTU is received, the Client determines if it has now received the entire file. If not, data reception continues. Otherwise, for Closed and Open Limited groups, the Client sends a Done message to the Server. For Open groups, the Client simply returns to idle state without sending any message to the Server.

After sending a Done message, the Client waits for receipt of a Completion message from the Server. Like the Announce message, the Completion message contains a Host List that specifies each Client

for which a Done message has been received (i.e. it confirms receipt of the Done message).

If the Client receives a Completion message and it's address is in the Host List, then the transfer is complete, the Client leaves the data delivery group (subject to conditions, as discussed above) and returns to idle state. Otherwise, the Done message is resent. The Done message is also resent for each Status Request message received from the Server (a Status Response is not sent). This procedure continues until the Client's Done message is confirmed or until the receive timer expires.

The logic flow diagram below illustrates the Client's completion processing as it applies to a Closed or Open Limited group.

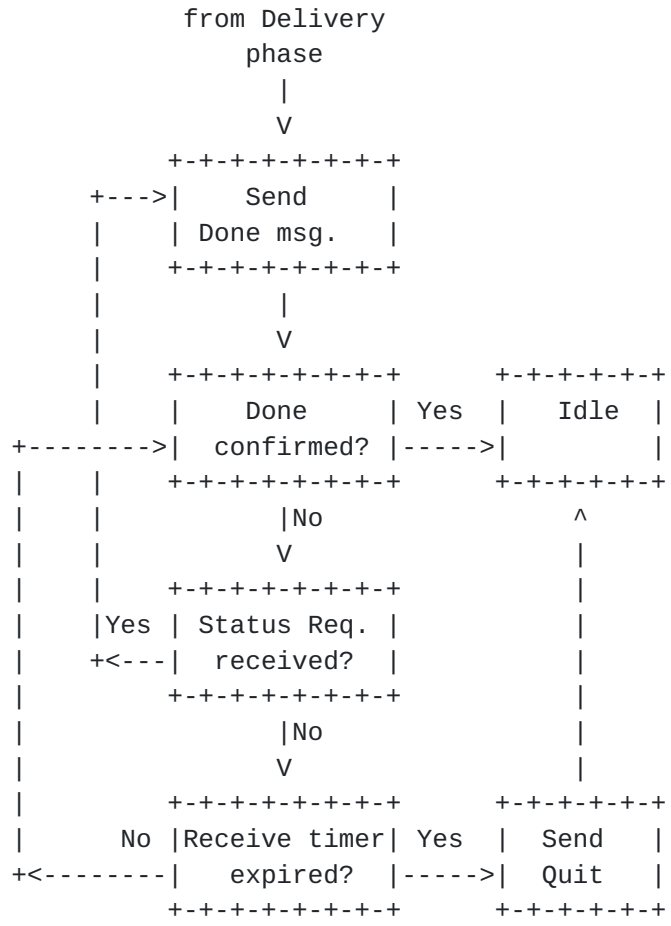



Figure 11-5 Client Product Completion

### 11.3.2 File Product Completion - Server

For Closed and Open Limited groups, the receipt of a Done message from a Client means that the Client has received all of the file

successfully. The Server confirms the receipt of Done messages with the Completion message. However, a separate message is not sent for each Done received. Rather, the Host List is used to "batch-confirm" all Done messages that have been received up to the time that the Completion message is sent.

The first Completion message is sent as soon as the first Done message is received, and periodically thereafter at intervals specified by the Completion Interval parameter. When a Completion message is sent, the interval timer is started. When the timer expires, the Server updates the Host List of the message with all Clients that have sent Done messages and sends the message again. If all Clients have not yet completed, the timer is restarted. There must be at least one additional transmission of the Completion message after the last Client has completed.

Since Clients will experience different network error rates, some Clients may complete the transfer while other Clients are still receiving retransmissions. This means that the Completion message may be sent while the file transfer is still in progress.

Data delivery continues until there are no longer any participating Clients (all Clients have sent Done or Quit messages), there is no response from any Client, or until the Delivery Time Limit duration has expired, whichever occurs first.

When the last Client completion has been confirmed with the Completion message, the Server considers the transfer complete and sends no further messages.

If the Server receives no response to a Status Request sent at the end of a pass, there may no longer be any Clients listening to the Server. The Status Delay and Status Retry Count parameters limit the length of time that the Server will attempt to receive a response. When the Status Request message is sent, the Server starts the Status Delay timer. If any status responses have been received by the time the delay timer expires, the Server begins the next pass where it retransmits the DTUs specified in the responses. If no responses are received, the Server sends another Status Request message and restarts the delay timer. This procedure repeats up to the limit set by the Status Retry Count parameter, or the delivery time limit expires, whichever occurs first. At that time, the Server sends an Abort message, terminates the product delivery, and sends four End messages over the Public Address. End messages are read by aggregation devices, if present, to determine that the session is over.

The Server may also use the Abort message to terminate transfer to one or more Clients at any time due to conditions at the Server or at the Client. Several Abort messages are sent to insure delivery as no response to the Abort message is required from the Clients.

For Open Unlimited groups, the Clients do not send Done messages. The Server simply continues to send retransmissions as long as there are

responses to the Status Request message. When there are no responses, the Server continues to send Status Request messages up to the Status Retry Limit (as discussed above) or the Delivery Time Limit expires. At that time, the Server sends a Completion message (with no Host List) and considers the transfer complete. There is no response from

the Clients.

For all group types, when the session is over the Server sends an End message on the same multicast address used for Announce. This message is sent four times and is used to conveniently notify infrastructure that may be aggregating that the session is ended. The End message structure is described in Section 12.

### 11.4 Flow Control

A first level of flow control may optionally be invoked to make sure particular parts of the network multicast tree do not become overloaded and bad receivers do not hold up transmission to the other members of the group.

The Announce message may optionally contain an error rate threshold (Note that this should probably be mandatory when MFTP is used in large heterogeneous networks such as the Internet). This threshold is used to inform clients that if they measure a DTU loss over the threshold they abort themselves from the group. By leaving the group, that leaf of the multicast tree is dissolved, removing congestion in that part of the network. The Server may then be directed by the application to set up a new group of aborted clients and create a new session at a later time at a lower transmission rate to service those clients.

## 12. Message Formats

This section defines the format of all MFTP protocol messages. All messages begin with a common protocol header that is 12 bytes in length and includes the message type. Additional information, depending on message type, is included in the message payload area. The format of the common header is shown in Figure 12-1 below.

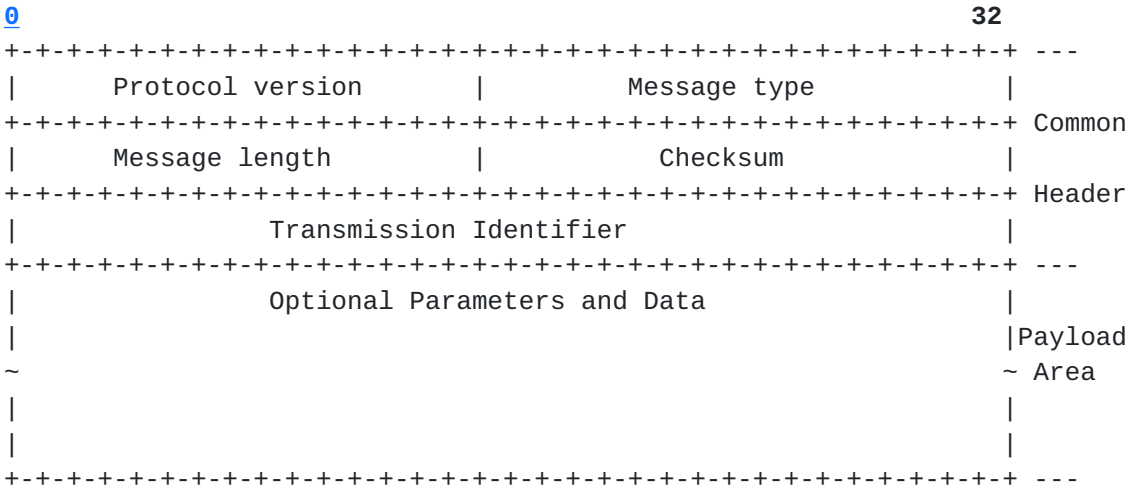


Figure 12-1 Protocol Header Format

Each header field is an unsigned integer. The fields are described below.

Protocol Version Number

This field contains an integer that identifies the MFTP protocol version that is implemented by the sender of this message. This document describes version 1.

Message Type

This field identifies the MFTP message type, as defined in the following table.

| | |
|---|---|
| QUERY GROUP | 0x0001 |
| JOIN GROUP | 0x0002 |
| LEAVE GROUP | 0x0003 |
| ANNOUNCE | 0x0004 |
| DATA TRANSFER | 0x0005 |
| STATUS REQUEST | 0x0006 |
| NAK RESPONSE | 0x0007 |
| COMPLETION | 0x0008 |
| ABORT | 0x0009 |
| REGISTRATION | 0x000A |
| DONE | 0x000B |
| QUIT | 0x000C |
| ECHO | 0x000D |
| ECHO RESPONSE | 0x000E |
| ACK RESPONSE | 0x000F |
| END | 0x0011 |

Message Length

This field contains the length, in bytes, of the entire MFTP message, beginning with the Protocol Version field.

Checksum

This field contains a checksum that verifies the integrity of the entire message including the MFTP header. A checksum is computed by the sender of each message and placed into this field. The receiver also computes the checksum and compares it with the received value. If the values do not match, the received message is discarded and treated as if it were never received.

The checksum is the 16 bit one's complement of the one's complement sum of all 16 bit words in the header and payload area. While computing the checksum, the checksum field itself is replaced with zeros.

The use of this field is optional as the UDP checksum provided within UDP should be used for this function. If the UDP checksum is used, this field should be set to zero.

Transmission Identifier

This field contains a number that uniquely identifies this transmission. A receiver uniquely identifies messages associated with this transmission by examining the source IP address and this transmission identifier. The transmission identifier is not applicable in MCP and is set to zero in MCP messages.

Payload

The information contained in the payload section of each message is parameterized. This is done for several reasons. First, it facilitates the addition of new parameter types as the protocol evolves. Second, it allows for the omission of parameters that do not pertain to a particular product type. This is particularly important in the Announce message.

All parameter data fields are a multiple of four bytes (i.e. 32 bits). When such a field is used to contain an integer, its format is as shown below. The integer is represented in two's complement notation. The most and least significant bytes are 0 and 3, respectively.

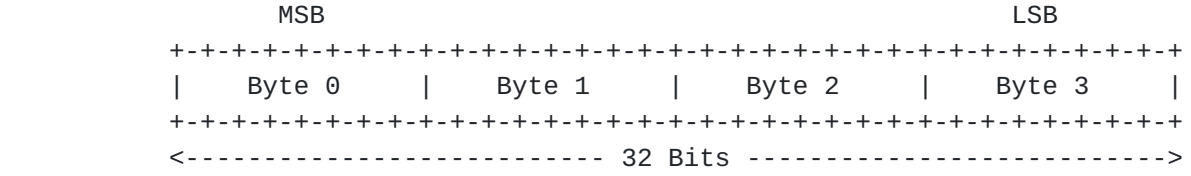



Figure 12-2 Basic Field Format

The type-length-value structure is shown below. The minimum parameter unit length is 64 bits. The Name and Length fields both consist of two 16 bit fields. The value field is variable length, but is always a multiple of 32 bits. When the value field contains a character or octet string that is not a multiple of 32 bits, the field is extended so that the 32 bit multiple requirement is always met. This means that 1, 2, or 3 pad bytes are added to the end of the string, and the value of those bytes is set to zero. However, the length field defines the true length of the value.

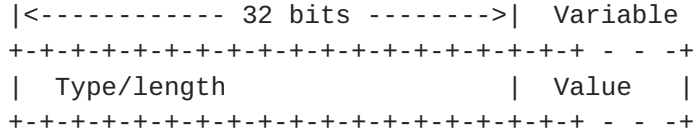



Figure 12-3 Type-Length-Value Format

Each item of data that is conveyed in the payload area of the message is identified as a separate message parameter and structured with the

type-length-value format. Each parameter may be used in one or more message types.

The following table defines the complete list of parameters and

includes the parameter description, type, and length.

| Parameter | Type | Data Length |
|---|---|---|
| Announce Duration | 0x0100 | 4 |
| Announce Options | 0x0101 | 4 |
| Application Reference | 0x0102 | variable(1-64) |
| Block Number | 0x0103 | 4 |
| Block Range | 0x0104 | 8 |
| Block Size | 0x0105 | 4 |
| Cancel Reason | 0x0106 | 4 |
| Client Address | 0x0107 | 4 |
| Data Rate | 0x0108 | 4 |
| Data Transfer Duration | 0x0109 | 4 |
| DTU Number | 0x010A | 4 |
| DTU Range | 0x010B | 8 |
| DTU Size | 0x010C | 4 |
| Echo Sequence | 0x010D | 4 |
| Echo Data | 0x010E | variable(1-4096) |
| Error Threshold | 0x010F | 4 |
| Event Count | 0x0110 | 4 |
| Event Hash | 0x0111 | 16 |
| Event List | 0x0112 | variable |
| Host List(complex) | 0x0113 | variable |
| Host List(simple) | 0x0114 | variable |
| Group Options | 0x0115 | 4 |
| Multicast Address | 0x0116 | 4 |
| Product Data | 0x0117 | variable |
| Product Name | 0x0118 | variable (1-256) |
| Product Size | 0x0119 | 4 |
| Private Address | 0x011A | 4 |
| Public Address | 0x011B | 4 |
| Registration Response | 0x011C | 4 |
| Response Address | 0x011D | 4 |
| Response Port | 0x011E | 4 |
| Serial Number | 0x011F | 4 |
| Server Address | 0x0120 | 4 |
| Server Port | 0x0121 | 4 |
| Status Type | 0x0122 | 4 |
| User Data | 0x0123 | variable (1-64) |

Figure 12-4 Message Parameters

Each parameter is described in the message definition sections below so that their use is understood in the context of each message type. Parameters may be required, optional, or omitted depending on the message type and data product type.

### 12.1 MCP Messages

The MCP messages have been listed in the Protocol Description section above. The format of each message is described below. All messages begin with the common protocol header (see Protocol Header above).

All subsequent fields are used to convey parameters to the receiving host, and use the type-length-value format. Each message description provides a list of the parameters included in that message type and whether each parameter is required or optional. The parameters may appear in the message in any order, not necessarily in the order presented in the list.

### 12.1.1 Query Group Message

The purpose of the Query Group message is to allow a Client to determine a Server's public address when only the Server's IP address is known. The message is sent in unicast mode to the Server. The response to this message is a Join Group message, also sent in unicast mode. A separate Join Group is returned for each public address active at the Server. If there are no public addresses active, then no response is sent by the Server. The Query Group message contains the following parameters.

| | |
|---|---|
| Application Reference String | (Mandatory) |
| Group Options | (Mandatory) |

Application Reference String

Given that there may be different (i.e. incompatible) application layer users of MFTP, this parameter serves to identify the application type operating at the Server and the Clients.

The parameter is an ASCII character string that identifies the application type. For example, the StarBurst file transfer application uses the string "StarBurstMFTP".

In a Client message, such as Query Group, the application running at the Client is identified. Hence, the Server responds to the message only when an application of the same type is running at the Server. In a Server message, such as Join Group (see below), this parameter identifies the application type that is running at the Server. The Client may then use the information to determine if it will respond to the Server message. That is, only Clients that have the same type application layer users should respond to the message.

Group Options

This parameter defines various options associated with group management and is contained in both the Group Query and Join Group messages. The parameter is a combination bit and byte field, as defined below:

| Byte | Bit | Setting |
|---|---|---|
| | | PRODUCT TYPE |
| 0 | 0-7 | 'F' = file product |
| | | 'A' = all product types |
| | | COMMON OPTIONS |
| 1 | 0-7 | Unused, set to zero |
| 2 | 0-7 | Unused, set to zero |
| 3 | 0-7 | Unused, set to zero |

Each option is explained below:

Product Type — This option identifies the product type(s) that a Client wants to receive (as used in Query Group message) or that a Server is transmitting(as used in Join Group message). Only file product is now defined.

Common Options — None currently defined.

### 12.1.2 Join Group Message

This message may be sent as a response to the Query Group message (see above), or as an unsolicited message to one or more Clients. In the former case, it provides the Server's public multicast address to the Client so that the Client may join the group to receive product announcements. In the latter case, it may specify the public address, or any other multicast address that the Server wants the Client(s) to join. When sent as a response to the Query Group message, it is sent in unicast mode. Otherwise, the message may be sent in unicast, broadcast, or multicast modes. The message contains the following parameters.

| | |
|---|---|
| Application Reference | (mandatory) |
| Multicast Address | (mandatory) |
| Group Options | (mandatory) |
| Host List (simple) | (optional) |

Application Reference String

Refer to the Query Group message for a description of this parameter. The Client joins the indicated group only if the Client has an application user of the same type indicated in the message.

Multicast Address

This parameter contains the multicast address that the receiving host should join. Normally, this will be the Server's public address, but may be any multicast address that the Server wants the Client(s) to join.

Group Options

Refer to the Query Group message for a description of this parameter.

The Server sets the Product Type field of this parameter to identify the type of data products that may be received by joining the indicated public multicast address.

Host List (simple)

This parameter is included when the sender wants to specify specific hosts that should act on the message. If the message is to apply to all receiving hosts, this parameter is omitted.

This parameter is an array of host records. Each host record in the array consists of a single 32 bit field, called the host identifier. Each host identifier contains the IP address of a host. The address is stored in network byte order.

There are other messages that may contain the Host List parameter (e.g. Announce, Abort). Whenever the length of the Host List does not fit within a single message, the Host List is divided and sent in two or more copies of the same message type. The multiple messages are considered a single logical message by the Server.

### 12.1.3 Leave Group Message

The purpose of the Leave Group message is to direct one or more remote hosts to leave a multicast group. The message may be sent in unicast, broadcast, or multicast modes. The message contains the following parameters.

| | |
|---|---|
| Multicast Address | (mandatory) |
| Host List(simple) | (optional) |

Multicast Address

This parameter specifies the multicast group that the Client(s) should leave.

Host List (simple)

Refer to the Join Group message for a description of this parameter.

### 12.1.4 Echo Message

The Echo message may be transmitted by any MFTP station. The purpose of the message is to test the network path, and the reliable transfer of data over that path, between the source and destination hosts. The response to this message is the Echo Response message (see next section). This message is similar in concept to the Echo message of the Internet Control Message Protocol (RFC792). This message may be sent in unicast, broadcast, or multicast modes. The message contains

the following parameters.

| | |
|---|---|
| Echo Sequence | (mandatory) |
| Echo Data | (optional) |
| Host List(simple) | (optional) |

Echo Sequence

This parameter is a sequence number to aid in matching a response with a transmitted message.

Echo Data

This optional parameter is an octet string that can be sent with the Echo message and is returned in the Echo Response message.

Host List (simple)

Refer to the Join Group message for a description of this parameter.

### 12.1.5 Echo Response Message

The Echo Response message may be transmitted by either a Server or Client, as a response to a received Echo message. This message is sent in unicast mode always.

All parameters received in the Echo message, except the host list, are echoed back to the sender exactly as received. The host list, if contained in the Echo message, is omitted from the response. Refer to the Echo Message above.

## 12.2 MDP Messages

The MDP messages have been listed in the Protocol Description section described previously. The format of each message is described below. All messages begin with the common protocol header (see Protocol Header, Figure 12-1). All subsequent fields are used to convey parameters to the receiving host, and use the type-name-length-value format. Each message description provides a list of the parameters included in that message type and whether each parameter is required or optional.

Unless otherwise stated, the parameters in messages sent by the Server may appear in the message in any order, not necessarily in the order presented in the list. To allow efficiency in aggregating response messages (when used), most response formats specify that the parameters must be in the order shown.

### 12.2.1 Announce Message

The Announce message is sent only by the Server. The purpose of the

message is to allow the Server to announce the impending transmission of a data product, to describe the data product, and to solicit Registration message responses from Clients (depending on group type). The Announce message may be sent in unicast, broadcast, or multicast

mode. In multicast mode, it is sent to the Public address.

The Announce message contains the parameters listed below. An intermediate network entity that is performing response aggregation may need to monitor Announce messages in order to obtain parameters that are needed for aggregation processing, such as the Response Address. Therefore, the first five parameters (Server Address/Port, Response Address/Port, and Private Address) must appear at the beginning of the message payload area and in the order shown. Other parameters may appear in any order.

| | |
|---|---|
| Server Address | (mandatory) |
| Server Port | (mandatory) |
| Response Address | (mandatory) |
| Response Port | (mandatory) |
| Private Address | (mandatory) |
| Announce Duration | (mandatory) |
| Announce Options | (mandatory) |
| Application Reference | (mandatory) |
| Block Size | (mandatory) |
| Data Rate | (mandatory) |
| Product Name | (mandatory) |
| Data Transfer Duration | (mandatory) |
| DTU Size | (mandatory) |
| Product Size | (mandatory) |
| Error Threshold | (optional) |
| User Data | (optional) |
| Host List (complex) | (optional) |

Server Address

This parameter is the IP address, in network byte order, of the Server. Its purpose is primarily for response aggregation. This parameter is copied to the response by the Client so that the aggregating entity knows where to send the aggregated response.

Server Port

This parameter is the receive UDP port number of the Server. Its purpose is primarily for response aggregation. This parameter is copied to the response by the Client so that the aggregating entity knows where to send the aggregated response.

Response Address

This parameter is the address to which Client response messages are to be sent. This allows responses to be sent to or intercepted by an intermediate aggregating entity. The address may be a unicast or multicast address, in network byte order. If response aggregation is

not being performed, this address is the host IP address of the Server.

Response Port

This parameter is the receive UDP port at the Response Address to which Client response messages are to be sent.

Private Address

This parameter specifies the network address to which the Server will transmit the product data. For a multicast transmission, this is a multicast group address. The Client must join the multicast group to receive the product data. Otherwise, this parameter specifies a unicast or broadcast address.

Announce Duration

This parameter is the time, in seconds, remaining before the product data transmission will begin. It is updated at each Announce message transmission so that it accurately reflects the remaining time. The Client does not act on this information. It is provided primarily as an item of information to the application layer user. The initial value of this parameter is provided by the Announce Time Limit configuration parameter (see Configuration above).

Announce Options

This parameter defines various options associated with a product transmission. The parameter is a combination bit and byte field, as defined below:

| Byte | Bit | Setting |
|---|---|---|
| | | PRODUCT TYPE |
| 0 | 0-7 | 'F' = file product |
| | | COMMON OPTIONS |
| 1 | 0-1 | 01 = closed group |
| | | 10 = open limited group |
| | | 11 = open unlimited group |
| | 2 | 1 = do not leave delivery group |
| | | 0 = leave delivery group |
| | 3-7 | Unused, set to zero |
| | | FILE OPTIONS |
| 2 | 0 | 1 = initial transmission |
| | | 0 = restart transmission |
| | 1 | 1 = overwrite existing file |
| | | 0 = preserve existing file |
| | 2 | 1 = late entry allowed |
| | | 0 = late entry disallowed |
| | 3-7 | Unused, set to zero |

Each option is explained below:

| | |
|---|---|
| Product Type | This option identifies the product type. Only file product is currently defined. |

| | |
|---|---|
| Common Option 1 (bit 0-1) | This group option defines whether the product may be received by any Client, or only by those specified in the Host List parameter (refer to the Group Management section above) |
| Common Option 2 (bit 2) | This option specifies whether the Client should or should not leave the data delivery group after the product delivery has completed. If the Server intends to send more than one product to the same Client group, then group setup time can be minimized by keeping Clients joined to the group. Since this option is contained in each product announcement, the final product announcement can direct Clients to leave the group after the transmission has completed. |
| File Option 1 | The initial/restart option indicates whether this is an initial transmission of the file (i.e. the entire file will be sent) or whether it is a restart of a file previously sent but not received completely by all Clients. |
| File Option 2 | The overwrite/preserve option indicates whether a Client should overwrite or preserve a current file that has the same name as specified in the Product Name parameter (see above). If this option is set to overwrite, then the Client is allowed to receive the new file and overwrite the old file. If this option is set to preserve, then the Client declines to receive the new file and preserves the existing file at the host. |
| File Option 3 | The late entry option defines whether a Client may participate in a file restart if it has not yet received any of the file. Since the file will have to be entirely transmitted for the Client, it will slow down other Clients that may need only a small portion of the file to complete their receipt of it. However, it does allow the Server to transmit the file to Clients that may have been offline when the file was originally sent. If the option is set to allowed, then a Client that has not yet received any of the file may participate in the file transfer. If the option is set to disallowed, then only Clients that have |

previously received some portion of the file
may participate.

Application Reference String

Refer to the Join Group message for a description of this parameter. The Client will not respond to this message unless its application type matches this value.

Block Size

This parameter is the number of DTUs that will be transmitted in each block. It allows the Client to initialize and manage its memory resources used for tracking DTUs that have been received or missed. If the product is a file, this value is limited by the maximum size of the bit map form of the Event List that can be carried in a Status Response message (see Event List below).

Data Rate

This parameter specifies the Server's transmit data rate. The Client may use this information to decide whether it can successfully participate in the product transfer. That is, the Client might decline a product transfer if it knows that the data rate will overrun the Client.

Data Transfer Duration

This parameter specifies the maximum amount of time, in seconds, that the Server will devote to the product delivery (not including the product announcement phase). The Client uses this value as its receive timeout. This parameter is configured (see Configuration section above).

DTU Size

This parameter is the maximum number of data bytes that will be contained in each DTU for a file transfer. The last DTU may contain less than this number of bytes. This value is computed by MFTP based on the size of the network Maximum Transmission Unit, the lower level protocol headers, and the other parameters that must reside in a DTU.

Error Threshold

This parameter defines the percentage of missed DTUs, relative to the total number of DTUs in the transmission, that can occur at a Client before the Client voluntarily removes itself from participation in the product transfer. That is, the Client monitors its own error rate and initiates its own removal from the transfer if this threshold is reached. This mechanism helps to prevent a small number of Clients with high error rates from adversely affecting all other Clients by requesting a high percentage of retransmissions. It is permissible for

the Client to continue to receive and process DTUs in anticipation of a file restart, but the Client must not send any message to the Server.

Product Name

This parameter is an ASCII character string that is the formal name for the product being announced. If the product is a file, this field contains the name of the file that should be created on the Client system to store the product (the filename may be different than the filename at the Server).

Product Size

This parameter is the total length, in bytes, of a file product. It allows the Client to determine if there is sufficient disk space to receive and store the product.

User Data

This parameter is an octet string that is provided by the application layer user. It is delivered to the application layer user at the Client. It could be used, for example, to describe the product being announced. Its use is optional and is limited to 64 octets.

Host List (complex)

This parameter is an array of host records. Each host record in the array consists of two 32 bit fields. The first field, called the host identifier, contains the IP address of the remote host, in network byte order. The second field, called the admin status, contains the status of the host specified in host identifier field. The format of the admin status field is defined below.

| Byte | Bit | Setting |
|---|---|---|
| 0 | 0-7 | 'A' = accepted to receive product<br>'D' = declined status confirmed<br>'P' = pending status |
| 1 | 0-7 | Decline reason |
| 2 | 0-7 | Unused, set to zero |
| 3 | 0-7 | Unused, set to zero |

The usage of the Host List depends on Group type being used (also see Group Management, Section 6). For Closed Groups, the Host List is included in the message and contains the IP address of each Client that is being invited to participate in the product transfer. The Admin status is initially set to 'P'. As Clients send Registration messages, the Server acknowledges receipt of the message by setting the Admin status in the next Announce message that is transmitted. When an invited Client indicates that it will be participating in the product transfer, its Admin status is set to 'A' (accepted). When an invited Client indicates that it is declining the product, its Admin

status is set to 'D' (declined) and the Decline Reason field is set to the value received in the Registration message. If an uninvited Client sends a Registration message for a Closed Group announcement, the Server simply ignores the registration and sends an Abort to the

client.

For Open Limited Groups, the Host List is initially omitted from the Announce message. As Clients begin to send Registration messages, the Server includes the Host List in subsequent Announce messages and adds an entry for each Client that is accepted to participate in the product delivery, and the Admin status is set to 'A'. If a host is rejected by the Server (e.g. the maximum number of hosts that the Server can handle is reached), its registration is simply ignored and an entry is not placed in the Announce message. The Host List is never included for Open Unlimited Groups.

### 12.2.2 Registration Message

The Registration message is sent only by the Client in order to respond to a received Announce message. The Registration message is sent to the Response Address contained in the Announce message. The message contains the following parameters, which must appear in the order shown.

| | |
|---|---|
| Server Address | (mandatory) |
| Server Port | (mandatory) |
| Client Address | (mandatory) |
| Registration Response | (mandatory) |

This response may be aggregated and relayed by an intermediate entity to increase scalability. A single response that is not relayed contains only the parameters shown. A relayed response repeats the Client Address, Serial Number, and Registration Response parameters for each Client that is being relayed.

Server Address

This field is echoed by the Client, exactly as received in the Announce message. It may be used by an intermediate network entity during the aggregation/relaying process.

Server Port

This field is echoed by the Client, exactly as received in the Announce message. It may be used by an intermediate network entity during the aggregation/relaying process.

Client Address

This field identifies the Client that is sending the response, and identifies individual Clients in an aggregated/relayed response. The address is in network byte order.

This parameter indicates the Client's intention to participate or not participate in the transmission of a product announced by a Server. If

the Client accepts the offer to participate in the transmission, the reason for decline field has no meaning and is set to the value zero. This parameter is a byte array with the following definition. Refer to Section 13 for a list of the decline reasons.

| Byte | Bit | Setting |
|---|---|---|
| 0 | 0-7 | 'A' = accept<br>'D' = decline |
| 1 | 0-7 | Decline reason |
| 2 | 0-7 | Unused, set to zero |
| 3 | 0-7 | Unused, set to zero |

### 12.2.3 Data Transfer Message

The Data Transfer message is transmitted only by the Server in order to transfer the data product to the Clients. There is no response by the Client to this message. This message may be sent in unicast, broadcast, or multicast mode. In multicast mode, this message is sent to the Private address. The message contains the following parameters.

| | |
|---|---|
| Pass Number | (mandatory) |
| Block Number | (mandatory) |
| DTU Number | (mandatory) |
| Product Data | (mandatory) |

Pass Number

This parameter contains the current pass number. The first pass is numbered 1, and increments for each additional pass. This parameter allows the Client to unambiguously determine the current pass number. This is primarily useful for reporting reception status to an application layer user.

Block Number

This parameter contains the sequence number of the current block, within the current pass, starting at 1.

DTU Number

This parameter contains the sequence number of the current DTU, within the current block, starting at 1.

Product Data

This parameter is the actual file data bytes being delivered to the Client.

### 12.2.4 Status Request Message

The Status Request message is sent only by the Server in order to query the Client for its reception status. The Client responds with an

ACK or NAK Response message, depending on the type of status requested (only NAKs are normally used). The Status Request message may be sent in unicast, broadcast, or multicast mode. In multicast mode, it is sent to the Private address. The message contains the following parameters.

| | |
|---|---|
| Pass Number | (mandatory) |
| Block Range | (mandatory) |
| DTU Range | (mandatory) |
| Status Type | (mandatory) |
| Host List (simple) | (optional) |

Pass Number

This parameter identifies the pass for which the Server is requesting status. If there is no applicable pass number (e.g. the Server is querying for status prior to a file restart), then this parameter value is set to zero.

Block Range

This parameter identifies the range of blocks for which the Server is requesting status. The parameter consists of two 32-bit fields. The first field contains the number of the first block in the range, and the second field contains the number of the last block in the range. If status is being requested for a single block, both fields are set to the same block number.

DTU Range

This parameter identifies the range of DTUs within a block for which the Server is requesting status. The parameter consists of two 32-bit fields. The first field contains the number of the first DTU in the range, and the second field contains the number of the last DTU in the range. The range cannot span multiple blocks. If status is being requested for a single block, then this parameter may specify a subset of all DTUs in the block. When status for multiple blocks is being requested, this parameter is set to the first and last DTU numbers of the entire block. That is, a DTU subset request is invalid when the Server is requesting status for multiple blocks.

Status Type

This parameter is an integer that identifies the type of status response that is being requested by the Server. The values are shown below:

| | | |
|---|---|---|
| 1 = NAK responses only | - | The Client responds only if it recognizes that it has not received one or more DTUs in the block(s) specified by the Server. It sends a NAK response message. Otherwise, it sends no response at all. This is the preferred mode of operation. |
| 2 = ALL responses | - | The Client always responds regardless of its reception status. If it has received all DTUs in the indicated block, it sends an ACK response message. Otherwise, it sends a NAK response message. |

Host List (simple)

Refer to the Join Group message for a description of this parameter. It is included when the sender wants to specify Clients that should respond to the message. If the message is to apply to all receiving Clients, this parameter is omitted.

### 12.2.5 NAK Response Message

The NAK Response message is sent only by the Client when queried by the Server via the Status Request message. The Client sends this response if the Status Request message specifies either the ALL or NAK response status type and the Client needs to request the retransmission of one or more DTUs in the indicated block. The NAK Response message is sent to the Response Address as specified in the Announce message. The response message contains the following parameters, which must appear in the order shown.

| | |
|---|---|
| Server Address | (mandatory) |
| Server Port | (mandatory) |
| Client Address | (mandatory) |
| Pass Number | (mandatory) |
| Block Number | (mandatory) |
| Event Count | (mandatory) |
| Event Hash | (mandatory) |
| Event List | (mandatory) |

Server Address

This field is echoed by the Client, exactly as received in the Announce message.

Server Port

This field is echoed by the Client, exactly as received in the Announce message.

Client Address

This field identifies the Client that is sending the response. The address is in network byte order.

Pass Number

This parameter is as specified for the Status Request message above. That is, the received value is echoed in the response message.

Block Number

The value of this parameter is set to the block number that is being reported. If the Server requests status for a single block, then this parameter contains the same number. If the Server requests status for a range of blocks, then a separate response message is returned for each block in the range and this parameter is set to the block number that is being reported in each message.

Event Count

This parameter indicates the total number of NAKed DTUs that are being reported in this message. In other words, this is the number of bits that are set in the Event List parameter.

Event Hash

This parameter is the result of a hash function being applied to the Event List parameter. The purpose is for the receiver to be able to easily determine if the Event List for any two or more Clients is identical. The MD4 hash function (RFC 1320) is used. This is useful for aggregation/relaying, when used.

Event List

This parameter identifies the DTUs that the Client has not received for the block in the Block Number field. It is a bit map, where each bit in the map represents a single DTU in the block. A bit set to '1' means that the DTU has not been received and a bit set to '0' means the DTU has been successfully received by the Client. Note that the Server may request the status for a subset of the DTUs in a block via the DTU Range parameter in the Status Request message. The Client still sends a complete bit map, but sets the bits for all DTUs outside the range to '0' (successfully received). Also note that the DTU Range parameter is not echoed in the response message.

### 12.2.6 ACK Response Message

The ACK Response message is sent only by the Client when queried by the Server via the Status Request message. The Client sends this response if the Status Request message specifies the ALL response status type and the Client has successfully received all of the DTUs in the indicated block (Note that this is not the preferred mode of operation). The ACK Response message is sent to the Response Address

as specified in the Announce message. The response message contains the following parameters, which must appear in the order shown.

| | |
|---|---|
| Server Address | (mandatory) |
| Server Port | (mandatory) |
| Pass Number | (mandatory) |
| Block Range | (mandatory) |
| Client Address | (mandatory) |

Server Address

This field is echoed by the Client, exactly as received in the Announce message.

Server Port

This field is echoed by the Client, exactly as received in the Announce message.

Pass Number

This parameter is as specified for the Status Request message above, i.e. the received value is echoed in the response message.

Block Range

This parameter identifies the range of blocks for which the Client is reporting ACK status. The parameter consists of two 32-bit fields. The first field contains the number of the first block in the range, and the second field contains the number of the last block in the range. If status is being reported for a single block, both fields are set to the same block number.

Note that the Server may request the status of a range of blocks, and the Client may return both ACK and NAK responses, according to the status of each block. While NAK responses are sent individually for each block being reported by the Client, the ACK response may report the successful reception of a range of blocks.

Client Address

This field identifies the Client that is sending the response, and identifies individual Clients in an aggregated response. The address is in network byte order.

### 12.2.7 Done Message

The Done message is sent only by the Client to the Server to indicate that a complete product has been received successfully. The message is sent to the Response Address as specified in the Announce message. The message contains the following parameters, which must appear in the order shown.

| | |
|---|---|
| Server Address | (mandatory) |
| Server Port | (mandatory) |
| Client Address | (mandatory) |
| Event Count | (mandatory) |
| Data Transfer Duration | (mandatory) |

Server Address

This field is echoed by the Client, exactly as received in the Announce message.

Server Port

This field is echoed by the Client, exactly as received in the Announce message.

Client Address

This field identifies the Client that is sending the response. The address is in network byte order.

Event Count

This parameter reports the total number of DTUs that the Client reported as missed during reception of the product. In other words, the field contains the total of all DTUs reported in NAK Response messages sent to the Server during reception of the product.

Data Transfer Duration

This parameter reports the actual time, in seconds, that was required for the Client to receive the product.

### 12.2.8 Completion Message

The Completion message is sent only by the Server in order to confirm receipt of Done messages from Clients. When Clients are not sending Done messages (i.e. Open-unlimited groups), the message announces the end of a transmission. This message may be sent in unicast, broadcast, or multicast mode. In multicast mode, this message is sent to the Private address. The message contains the following parameter.

| | |
|---|---|
| Host List (simple) | (optional) |

Host List

This parameter is included only when the Clients are sending Done messages. The inclusion of a Client address in the list confirms receipt of the Client's Done message.

### 12.2.9 Abort Message

The Abort message is sent only by the Server in order to cancel a product announcement or product delivery that is currently in progress or to abort a specific Client or Clients. There is no response to this message by the Client, so multiple Abort messages should be sent to increase the probability of success. The message may be sent in unicast, broadcast, or multicast mode. In multicast mode, the message is sent to the Private address. The message contains the following parameters.

| | |
|---|---|
| Cancel Reason | (mandatory) |
| Host List (simple) | (optional) |

Cancel Reason

This parameter indicates the reason that the Server is aborting a product transmission. Refer to Section 13 for the value list.

Host List (simple)

Refer to the Join Group message for a description of this parameter. This parameter is included when the Server wants to Abort a subset of the Client population. If the parameter is omitted, then all receiving Clients are affected.

### 12.2.10 Quit Message

The Quit message is sent only by the Client in order to terminate its participation in the current product transfer. There is no response to this message by the Server, so multiple messages should be sent to increase the probability of success. The message is sent to the Response Address as specified in the Announce message. It includes the following parameters.

| | |
|---|---|
| Cancel Reason | (mandatory) |

Cancel Reason

This parameter indicates the reason that a Client is terminating reception of a product transmission. Refer to Section 11 for the value list.

### 12.2.11 End Message

The End message is sent only by the Server in order to conveniently indicate to aggregating entities in the network, if present, that the session is over. End messages are sent four times after the Server has determined the session is over on the Public Address. There are no

response messages back to the End message.

The End message includes the following parameters.

| | |
|---|---|
| Server Address | (mandatory) |
| Server Port | (mandatory) |
| Response Address | (mandatory) |
| Response Port | (mandatory) |

## 13.Reason Codes

The various "reason" codes used in PDUs are listed below. Refer to previous sections for a description of the message types and configuration parameters mentioned here.

### Client Registration Message - decline reasons

1001 insufficient disk space

1002 insufficient memory

1003 system error (disk, I/O, etc)

1004 already have file. The Server has indicated in the Announce message that the Client should preserve any existing copy of the file. Therefore, the Client is confirming that it already has the file and will not be participating in the current transfer. Refer to the overwrite/preserve file option in the Announce message.

1005 no file to restart. The Server has indicated that this is a file restart. However, the Client does not currently have any portion of the file and the sender wants to restrict the transfer to only those Clients that already have a portion of the file. Refer to the initial/restart and late entry file options in the Announce message.

1006 unknown application type (ARS). The Announce message specifies an ARS that does not match the ARS set by the application layer user. Refer to the Application Reference String parameter in the Announce message.

1007 no interest in sender. The application layer user has not specified this Server as one of the Servers that it wishes to receive data products from.

1008 no interest in product type. The application layer user has not specified this product type as one of the types that it wishes to receive. Currently, only file product is supported but this provides expansion capability for the future. Refer to the product type specification in the Announce message.

Client Quit Message - cancel reasons

2001 application action. The user application has terminated this product transfer
2002 transfer timeout. The maximum time duration specified for the transfer of this product has expired. Refer to the Data Transfer Duration parameter in the Announce message.
2003 system error. An unrecoverable host error has occurred.
2004 error threshold. The error threshold has been reached for this Client. Refer to the Error Threshold parameter in the Announce message.

Server Abort Message - cancel reasons

3001 application action. The user application has terminated this product transfer.
3002 transfer timeout. The maximum time duration specified for the transfer of this product has expired. Refer to the Delivery Time Limit configuration parameter.
3003 status timeout. The maximum time duration specified for status acquisition has expired. Refer to the Status Retry Timer and Status Retry Count configuration parameters.
3004 system error. An unrecoverable host error has occurred.

## 14. Future Extensions

Work is ongoing to provide improvements to the basic MFTP protocol. Some of the improvements under consideration are:

1. More comprehensive flow control. As currently defined, MFTP handles flow control simply by having client receivers with excessive DTU error rates (excessive as defined by the server) remove themselves from the group.
2. More generalized host lists. Currently, host lists include the IP addresses of the hosts included in the list. It has been suggested that the host lists contain the host names rather than the IP address; however, these will occupy larger fields than the 32 bit address field.
3. Addition of Security (see Section 15).
4. Interfaces to the mmusic protocols.
5. Interfacing to RSVP.

## 15. Security Considerations

This version of the protocol does not include any distinct message types or additional fields for existing messages that will be needed to support security. Security is currently under study and the specification will be updated at a future time to address the security requirements based on the work performed in the IPSEC IETF Working

Group.

## 16. Acknowledgments

The authors would like to thank Scott Bradner (Harvard University), Ken Cates (StarBurst Communications), and Tony Speakman (Cisco Systems) for their comments and suggestions on various aspects of this document.

References

[1] Deering, S., "Host Extensions for IP Multicasting", RFC 1112, August 1989.

[2] Postel, J., Reynolds, J. "File Transfer Protocol (FTP)", RFC 959, October 1985.

[3] Schulzrinne, H., Casner, S., Frederick, R., Jacobson, V., "RTP: A Transport Protocol for Real-Time Applications", RFC 1889, January 1996.

[4] Schulzrinne, H.,"RTP Profile on Audio and Video Conferences with Minimal Control", RFC 1890, January 1996.

Author's Addresses

Kenneth Miller
StarBurst Communications, Inc.
150 Baker Ave.
Concord, MA 01742
Phone: +1 (508) 287-5560
Email: miller@starburstcom.com

Kary Robertson
StarBurst Communications, Inc.
150 Baker Ave.
Concord, MA 01742
Phone: +1 (508) 287-5560
Email: krobertson@starburstcom.com

Alex Tweedly
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134
Phone: +1 (408) 526-8114
Email: agt@cisco.com

Marc White
StarBurst Communications, Inc.
150 Baker Ave.
Concord, MA 01742
Phone: +1 (508) 287-5560
Email: mwhite@starburstcom.com