IEN: 97

FLEXIBLE DATAGRAM PROTOCOL

Version 1

April 1979

prepared for

Defense Communication Agency
WWMCCS ADP Directorate
Command and Control Technical Center
11440 Isaac Newton Square
Reston, Va. 22090

by

MITRE Corporation
1820 Dolley Madision Blvd.
McLean Va. 22102

TABLE OF CONTENTS

Preface

This is not a formal protocol specification. As such there are descriptions that are weak and open to interpretation. This is a working document describing the kinds of functions that we feel will be needed in a cable-bus transport level protocol. As our implementation progresses, certain functions may be done away with completely or may be subsumed into other higher level protocols. If the implementation is successful, and if there is sufficient interest, a less ambiguous version will follow.

A description of the project which will use this protocol is contained in [1]. Reference 1 is recommended as a closely coupled companion to this IEN.

The protocol was constructed by taking pieces from other definitions. The Internet Datagram Protocol [2] and the Transmission Control Protocol [3] were used as models both in terms of mechanisms and document format. Many thanks to the originators of those documents.

Steve Holmgren

Introduction

The Flexible Datagram Protocol (FDP) defines a set of rules to govern the transport of blocks of data, called datagrams, over interconnected cable-bus networks with binary degrees of reliability, flow control, addressing, and other common transport level protocol mechanisms. FDP uses variable length datagram headers. Each header contains a bit-map specifying the "shape" or attributes of the remaining portion of the header. These attributes are groups of data fields which, if specified, cause the protocol mechanisms referred to above to be invoked. If an attribute is not specified, default processing mechanisms will be invoked and attribute data fields are not placed in the header.

Motivation

A protocol may be viewed as a collection of mechanisms to support a specific service. Traditional mechanisms include: flow control, addressing, and reliability (e.g. checksums, parity etc.). Newer mechanisms include datagram fragmentation and reassembly to enable their passage through "smaller-sized" networks, and handling of a datagram security level.

The Flexible Datagram Protocol is motivated by a need to support a cable bus user community with widely varying transport protocol requirements. The user community ranges from cable-based telephone users who need audio, and soon video, capabilities, to the somewhat classic terminal-to-computer and computer-to-computer users.

The FDP meets these needs by allowing a user to dynamically specify the mechanisms to be applied to a datagram. If a user requires a mechanism to support a particular type of data transfer, the price is paid in terms of header overhead and processing cycles. No penality is paid if a user has no need for a particular mechanism.

Scope

The Flexible Datagram Protocol is intended to provide a full range of mechanisms to support the communication of packets of data, called datagrams, between low-level nodes of interconnected cable-busses. This version defines the selection of the following protocol mechanisms:

1. network addressing,
2. host addressing,
3. data reliability,
4. flow control,
5. datagram fragmentation,
6. higher protocol layer,
7. type of service, and
8. options.

Each of the mechanisms is described below.

Interfaces

On one side, FDP interfaces to a higher level protocol; possibly a virtual circuit protocol such as Transmission Control Protocol. On the other side it interfaces directly with the lowest level software to transfer a datagram between itself and a cable-bus.

OVERVIEW

This section gives an overview of the framework used to select the mechanisms to be applied to a datagram and then an overview of the operation of the mechanisms.

Framework

The framework consists of a bit map, called an Attribute Specification, and a pre-defined sequence in which a fixed-format group of data fields, called attributes, are processed. The Attribute Specification defines whether an attribute has been placed in the header. If the specification indicates that an attribute is in the header, the appropriate number of bytes are handed to the cooresponding protocol mechanism for processing. If an attribute is not in the header, default processing takes place. The next attribute in the pre-defined sequence is then checked. This cycle continues until the Attribute Specification is exhausted.

Protocol Mechanisms

Attributes are processed in the same sequence in which they are described in this section.

Network Addressing. The Network Address Attribute is provided to allow users to address sites on a remote network via a gateway or series of gateways which interconnect two or more networks.

The Network Addressing Attribute fields specify the source and destination networks for the datagram. Since the cable-bus is broadcast in nature, all gateways to other networks will "see" any request for transmission to a remote network. The gateway(s) to the specified network is (are) responsible for accepting the datagram, processing the Attribute specification and performing the appropriate routing of the remaining portion of the datagram to the remote network.

Host Addressing. The Host Addressing Attribute is necessarily provided to allow users to address datagrams to specific destinations.

The Host Addressing Attribute fields specify the source and destination host numbers for the datagram.

Reliability. The Reliability Attribute is provided to insure that datagrams are delivered without enroute damage.

The Reliability Attribute has a checksum field and a length field. The checksum is the complement of the sum of the datagram octets. The length field specifies the number of all octets in the datagram.

Flow Control. The Flow Control Attribute allows a receiver to control the speed at which a transmitter may send datagrams. It uses a sliding window acknowledgement strategy to acknowledge previously received datagrams and to detect duplicate and out-of-sequence datagrams.

Each flow controlled datagram contains a sequence number ordering the datagram in relation to previous and future datagrams, an acknowledgement field acknowledging datagrams previously received by the transmitter, and a window field specifying a range of acceptable per datagram sequence numbers.

Fragmentation. The Fragmentation Attribute is included to allow the transmission of large (greater than 256 octets) messages as a series of datagrams which are reassembled at the destination before delivery to a user. Further, it enables a more direct interconnection of cable bus systems with "small-sized" networks.

The Fragmentation Attribute contains a sequence number defining the relationship between previous and future fragments of a larger message, a message id relating fragments of a message, a flags field controlling further fragmentation and last fragment indications, and a life time specification which is decremented as the datagram passes through different internetwork gateways. If the life time field reaches zero, the datagram is assumed to be looping through a sequence of gateways and is discarded.

Higher Protocol Layer. The Higher Protocol Layer Attribute specifies the next layer of protocol which is to receive the datagram. It is included to allow the use of FDP by several higher level protocol implementations within the same host. By specifying the next protocol layer within a lower layer, the format of the headers of the higher level protocols are not restricted to a common preamble which would be required to demultiplex messages.

Type of Service. The Type of Service Attribute is included to allow the user to give some indication of the priority which is to be applied to a datagram. Initially, the priority may be restricted to linkage of datagrams to the front of internal queues so that immediate attention is given to their processing. Later, this may be expanded to the notion of preemptive allocation of resources, and selection of higher speed, less congested transmission channels.

The Type of Service Attribute contains a field to specify the priority of the message (lowest to highest), and a field to specify the requested speed of the message (again highest to lowest) within the priority level.

Options. The Options Attribute provides control functions needed or useful in some situations but unnecessary for routine communications. It is also provided to support experimental mechanisms that at some point may be elevated to Attribute status.

The Options Attribute includes provisions for special routing, error messages, protocol version specification, datagram security level, and special low-level signals such as reset.

SPECIFICATION

Header Format

The header contains an Attribute Specification followed by a variable number of attributes. Each attribute is a group of data fields. This is the format of a header specifying all attributes. The brackets attempt to delineate attribute boundaries.

```
                    1                   2                   3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 !  Att. Spec.   !  Att. Spec.   ! [ Dest. Net.  !   Src. Net ]  !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 ! [       Dest. Host            !           Src. Host        ]  !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 ! [ Chksum      !     Len     ] ! [    Ack      !    Window     !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 !             Seq. Num.       ] ! [   Flags     !  Life Time    !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 !         Frag. Offset          !            Msg. Id         ]  !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 ![ Nxt. Proto. ]![Type of Serv.]! [          Options         ]  !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 !                             Data                              !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Note that each tick mark corresponds to a bit position.

Attribute Specification: 8 bits (replicated)

Each bit in the Attribute Specification determines whether an attribute is present in the header. The order in which the attributes are processed corresponds to the bit positions in the Attribute Specification. The order in which the attribute fields are stored in the header is shown in the figure above. Note that the Attribute Specification is repeated in the header so that damage to it may be detected. If a damaged Attribute Specification is detected, the datagram is discarded.

The figure below shows the correspondence between Attribute Specification bits and attributes.

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
!N H R F F P T O!
!A A E L R R O P!
!D D L O G O S T!
+-+-+-+-+-+-+-+-+
```

NAD: Network Addressing	FRG: Fragmentation
HAD: Host Addressing	PRO: Next Level Protocol

```
        REL: Reliability                TOS: Type of Service
        FLO: Flow Control               OPT: Options
```

If a bit is on in the Attribute Specification, the attribute fields will be found in the header. If a bit is off, the attribute fields are not present in the header. The following example shows a header with Host Address and Reliability Attributes. The brackets deliniate attribute boundaries.

```
                      1                   2                   3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 !0 1 1 0 0 0 0 0!0 1 1 0 0 0 0 0! [        Dest Host            !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 !          Src. Host            ] ! [    Chksum     !     Len   ] !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 !          Data                                 !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

ATTRIBUTES

Network Addressing: (Bit 0)

The Network Addressing Attribute has a Destination and a Source Network field.

If not specified the datagram is not routed outside the local network. See Addressing Operation below.

Dest. Net. (Destination Network): 8 bits
Contains the number of the network to which the datagram is to be routed.

Src. Net. (Source Network): 8 bits
Contains the number of the network from which the datagram originated.

Host Addressing: (Bit 1)

The Host Addressing Attribute has a Destination and Source Host field.

If not specified, the datagram is a broadcast message to all hosts. See Addressing Operation below.

Dest. Host. (Destination Host): 16 bits
Contains the number of the host to which the datagram is to be routed.

Src. Host (Source Host): 16 bits
Contains the number of the host which originated the datagram.

Reliability: (Bit 2)

The Reliability Attribute contains a checksum and a length field.

If not specified, the datagram is assumed to be undamaged and its length is obtained from the hardware interface.

Chksum (Checksum): 8 bits
The Checksum field contains the one's complement of the one's complement byte sum of the datagram. The Checksum field is set to zero while the checksum is being computed.

Len. (Length): 8 bits
The Length field specifies the number of octets in the datagram. This field counts all header and data octets.

Flow Control: (Bit 3)

The Flow Control Attribute contains an Acknowledgement field, a Window field, and a Sequence Number field.

If not specified, the datagram is not subject to flow control considerations. See Flow Control Operation below.

Ack. (Acknowledgement): 8 bits
The Acknowledgement field contains a sequence number greater than or equal (cyclically) to the sequence numbers of all successfully received datagrams.

Window: 8 bits
The Window field contains the number of datagrams beyond the sequence number in the Acknowledgement field which the sender of the datagram is willing to accept.

Seq. Num. (Sequence Number): 16 bits
The Sequence Number field contains the sequence number of the datagram.

Fragmentation: (bit 4)

The Fragmentation Attribute contains a Flags field, a Life Time field, a Fragment Offset field, and a Message identifer.

If the Fragmentation Attribute is not specified, gateways are free to fragment the datagram into smaller messages if required by the destination network. See Fragmentation Operation below.

Flags: 8 bits
Various Control Flags.

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
!0 D M 0 0 0 0 0!
!0 F F 0 0 0 0 0!
+-+-+-+-+-+-+-+-+
```

```
Bit 0:  reserved, must be zero.
Bit 1:  Don't Fragment This Datagram (DF).
Bit 2:  More Fragments Field (MF).
Bit 3:  Unused, must be zero.
Bit 4:  Unused, must be zero.
Bit 5:  Unused, must be zero.
Bit 6:  Unused, must be zero.
Bit 7:  Unused, must be zero.
```

Life Time: 8 bits
This field is decremented for each hop taken through the internetwork system. If it decrements to zero, the datagram is presumed to be in an internetwork loop and should be discarded.

Frag. Offset (Fragmentation Offset): 16 bits
This field relates the datagram to previous and future fragments. Each fragment datagram is given a sequence number. This field orders the fragment in relation to other fragments.

Msg. Id. (Message Identifer): 16 bits
This field is an arbitrary identifer for the message that was fragmented. Each message fragment contains the identifer to be used as an aid in reassembling the fragments of the message.

Next Higher Level Protocol: (bit 5)

This attribute contains the Next Protocol field.

If this attribute is not specified, a default higher level protocol receives the datagram.

Nxt. Proto. (Next Protocol): 8 bits
This field contains an identifier of the next higher level protocol which is to receive that contents of the data portion of the datagram.

Type of Service: (bit 6)

This attribute contains the Type of Service field. This field, if present, defines the priority and relative speed requirements within the priority which the sender wishes to attach to the datagram.

If not specified, no special handling is given to the datagram.

Type of Serv. (Type of Service): 8 bits
The Type of Service field contains two sub-fields with define the priority of the datagram and the speed which is to be applied to the datagram.

```
                 0 1 2 3 4 5 6 7
                +-+-+-+-+-+-+-+-+
                !  Pri   ! Spd !
                +-+-+-+-+-+-+-+-+

           Priority               Speed
           0  - Lowest            0 - Slowest
           31 - Highest           7 - Fastest
```

Options: (bit 7)

This attribute contains a variable number of fields. The format is an option-type octet, an option-length octet, and the actual option-data octets. There are two special case options which have only the option-type octet (End of Options List and Nop).

The option-length octet includes the option-type octet and the option-length octet in the octet count of the option length.

The option-type octet can be viewed as having three fields:

1 bit	reserved, must be zero
2 bits	option class
5 bits	option number

The option classes are:

0 = control
1 = internet error
2 = experimental debugging and measurement
3 = reserved for future use

If not specified, no special option processing is requested.

The following options are defined:

Class	Number	Length	Description
0	0	-	End of Option List.
0	1	-	No Operation
0	2	4	S/P/T. Security, Precidence, TCC
0	3	var.	Source Routine.
0	31	4	Reset
0	30	var.	Status
1	1	var.	General Error Report.
2	4	var.	Internet Timestamp
2	5	var.	Satellite Timestamp

Specific Option Definitions

End of Option List. This option indicates the end of option list. It is always used to terminate the list of all options.

```
+---------+
!00000000!
+---------+
```

No Operation. This option may be used between options to align the beginning of a subsequent option on a 32 bit boundary.

```
        +--------+
        !00000001!
        +--------+
```

S/P/T. This option provides a way for AUTODIN II hosts to send security, precedence, and TCC (closed user groups) parameters through networks whose transport leader does not contain fields for this information.

```
        +--------+--------+--------+--------+
        !00000010!00000100!Prec!Sec!   TCC  !
        +--------+--------+--------+--------+
```

```
        Precedence: 4 bits
                Specifies one of 16 levels of precedence.

        Security: 4 bits
                Specifies one of 16 levels of security.

        Transmission Control Code (TCC): 8 bits
                Provides a means to compartmentalize traffic
                and define controlled communities of interest
                among subscribers.
```

Source Routing. The source routing option provides a means for the source of a datagram to supply routing information to be used by gateways in forwarding the datagram to the destination.

A source route is composed of a series of internet addresses. The pointer is initially zero, which indicates the first octet of the source route. The segment is routed to the address in the source route indicated by the pointer. At the internet module the pointer is advanced to the next address in the source route. This routing and pointer advancement is repeated until the source address is exhausted. At that point, the destination may have been reached, if not, the protocol module must attempt to route the packet to the destination in the destination address field by the ordinary routing procedure.

```
        +--------+--------+--------+--------+-----/ /-----+
        !00000011! length ! pointer!  source route        !
        +--------+--------+--------------------/ /------+
```

Reset. The reset option allows a host on a network to signal other hosts that its network operations have been restarted. The data field contains the address of the restarted host.

```
        +--------+--------+--------+--------+
        !00011111!00000100!  Host Address   !
        +--------+--------+--------+--------+
```

Status. The status option allows a host to transmit status information to a remote host. The conditions for eliciting the information and the content of the data fields are network dependent.

```
        +--------+-------+--------+--------/ /------+
        !00011110! length !   Status Info          !
        +--------+-------+--------+--------/ /------+
```

General Error Report. The general error report is used to report an error detected in the processing of a datagram to the originator of the datagram. The "err code" indicates the type of error detected and the "id" is copied from the message id field of the datagram, if it exists. Additional octets of error information may be present depending on the error code.

```
        Err Code:
            0  -  Undetermined Error Used when no in-
            formation is available about the type  of
            error  or  the  error does not fit in any
            defined class.

            No  error codes for specific classes have
            been defined.
```

```
        +--------+-------+--------+--------+----/ /------+
        !00100001! length !err code!   id   !            !
        +--------+-------+--------+--------+----/ /------+
```

Internet Timestamp. No information is available on the specific format of Timestamps.

```
        +--------+-------+--------+--------+----/ /-----+
        !01000100! length !                             !
        +--------+-------+--------+--------+----/ /-----+
```

Satellite Timestamp. No information is available on the specific format of Timestamps.

```
        +--------+-------+--------+--------+----/ /-----+
        !01000101! length !                             !
        +--------+-------+--------+--------+----/ /-----+
```

Addressing Operation

A distinction is made between names, addresses, and routes [4]. A name indicates what we seek. An address indicates where it is. A route indicates how to get there. The Flexible Datagram Protocol deals only with addresses. It is the task of higher level protocols to make the mapping from names to addresses. It is the task of lower level procedures (i.e. internet gateways) to make the mapping from addresses to routes.

When the Network Address Attribute is specified, the network fields have values obtained from reference [5]. When a message is transmitted to the cable-bus, all internet gateways watch for messages with destination networks to which they have access. If a match is found, the remaining attributes of the header are processed according to specified convention. The datagram is then passed along the route to the remote network after possible fragmentation.

When the Host Address Attribute is specified, hosts compare the destination host field with their address. If a match is found, and the destination network number (if specified) matches the local network number, the host processes the remaining Attributes in the header of the datagram and passes the data portion of the datagram to the next higher level protocol.

Flow Control Operation

Flow control regulates the transfer of data. Each receiver controls the amount of data a transmitter may send. Each receiver can dynamically update this control without loss or duplication of data.

Each datagram containing the Flow Control Attribute is assigned a sequence number. The sequence numbers range from 0 to 65535 and are used cyclically; i.e. 0 follows 65535. The sequence number for each datagram is placed in the header of each outgoing datagram containing the Flow Control Attribute. The first sequence number used is zero.

The Ack field contains the sequence number of the last datagram accepted by the transmitter. The receiver can consider all sequence numbers (cyclically) less than or equal to the number in the Ack field to have been received by the other end and can free buffers accordingly.

The Window field contains the number of datagrams, beyond that denoted by the Ack field, the transmitter is currently prepared to accept. The datagrams will have sequence numbers (Ack + 1) through (Ack + Window) cyclically calculated. A window value of zero indicates that the transmitter is not prepared to accept any datagrams until further notice. This does not mean that a transmitter may not send a datagram. It means that retransmission intervals should be increased significantly.

The sending of a datagram with a non-zero window does not irrevocably commit a transmitter to accept that number of datagrams. Changing conditions may cause an untimely reduction in the window size. These conditions may prevail at the same time other transmitters are sending datagrams (for which they were given a non-zero window) to the afflicted host. Sequences of this kind can generate duplicate and discarded datagrams.

A receiver must be able to detect and discard duplicate datagrams. In order for duplicate detection to be possible, the Window field must not contain a value greater than half the sequence number space (i.e. 32768) and no more than 32768 datagrams may be unacknowledged at any time. A receiver may identify duplicate datagrams as those with sequence numbers in the range

((last acknowledged) - 32767) through (last acknowledged)

A receiver should discard any datagrams with sequence numbers in this range.

Sending a window size greater than 32768 is prohibited. Receiving a window size greater than 32768 should be adjusted to 32768.

Certain combinations of events can generate the receipt of datagrams out of sequence. A receiver may discard out-of-sequence datagrams or it may save them for later insertion into the proper sequence.

It is possible for datagrams to arrive for which a window does not currently exist. A receiver may discard these datagrams.

A transmitter should be aware of these situations and have sufficient mechanisms to retransmit a datagram after a reasonable time has elapsed. Various strategies for defining "reasonable" are under study.

The simplest strategy a receiver can employ is to accept only the next datagram in sequence and discard all others. This works if the receiver employs a somewhat linear window policy.

Acknowledgements should be sent as soon as possible. They may be carried by datagrams flowing the other way. If no datagram is available for carrying the response after a "reasonable" time a datagram containing appropriate Address Attributes and the Flow Control Attribute should be artifically constructed and transmitted. It may be reasonable to employ a time-out mechanism controlling generation of "acknowledgement only" datagrams.

Fragmentation Operation

Fragmentation of a datagram may be necessary when it originates in a remote network that allows a large datagram size and must traverse the local network which limits datagrams to a smaller size.

The Message Identification field is used together with the source and destination address (if present), and the Next Protocol field (if present) to identify fragments for reassembly.

The More Fragments flag bit (MF) is set if the datagram is not the last fragment. The Fragment Offset field identifies the fragment number relative to the beginning of the original unfragmented datagram; zero is the first fragment, one the second and so on.

When fragmentation occurs, options are generally not copied, but remain with the first fragment. Some options, such as source routing, must be copied.

The fields which may be affected by fragmentation include:

(1) options field
(2) more fragments flag
(3) fragment offset
(4) checksum (if present)

If the Don't Fragment flag (DF) bit is set then fragmentation of the datagram is not permitted, although it may be discarded. This is used where the receiving host does not have resources to reassemble fragments.

The choice of Message Identifier for a datagram is based on the need to provide a way to uniquely identify the fragments of a particular datagram. The protocol module assembling fragments judges fragments to belong ot the same datagram if they have the same source, destination, Next Higher Level Protocol (if present), and Message Identifier. Thus, the sender must choose the Identifier to be unique for this source and destination pair and protocol over the time the datagram (or any fragment of it) could be alive in the internetwork.

It is appropriate for some higher level protocols to choose the identifier. For example, TCP modules may retransmit an identical TCP segment, and the probability for correct reception would be enhanced if the retransmission carried the same identifier as the original transmission since fragments of either datagram could be used to reconstruct a correct TCP segment.

REFERENCES

[1] Skelton, A.P., Holmgren, S.F., "The MITRE Cablenet Project", IEN 96, April 1979

[2] Information Sciences Institute, "Internet Datagram Protocol", Version 4, IEN 80, February 1979

[3] Information Sciences Institute, "Transmission Control Protocol", Version 4, IEN 81, February 1979

[4] Shoch, J., "A Note On Inter-Network Naming, Addressing, and Routing," Xerox Palo Alto Research Center, IEN 19, January 1978.

[5] Postel, J., "Assigned Numbers," RFC 750, NIC 45500, 26 September 1978.