RFC: 400008

DATA TRANSFER PROTOCOL

CREATING VIRTUAL NETWORKS OVER TCP/IP LAYER

PROTOCOL SPECIFICATION

April 2019

prepared for

Defense Distributed Applications

Programming Project

Creating Virtual Networks over TCP/IP layer

Lebanese University

1700 Nabatiyeh, Lebanon

by

Master 1 Students

Lebanese University

Faculty Of Sciences

TABLE OF CONTENTS

PREFACE ........................................................ 3

1. INTRODUCTION ..................................................... 4

1.1 Motivation .................................................... 4

1.2 Scope ......................................................... 4

1.3 Operation ..................................................... 4

2. OVERVIEW ......................................................... 6

2.1 Relation to Other Protocols ................................... 6

2.2 Model of Operation ............................................ 6

2.3 Function Description .......................................... 7

3. SPECIFICATION ................................................... 10

3.1 Internet Header Format ....................................... 10

References . . . . . . . . . . . . . . . . . . . . . . . . . . . . 13

PREFACE

This document specifies a part of the DoD Standard Data Transfer Protocol.

This edition revises aspects of addressing, option codes, and the security, precedence, compartments, and handling restriction features of the internet protocol.

Maria Afara

Ali Fakih

Editor

CREATING VIRTUAL NETWORKS OVER TCP/IP LAYER

PROTOCOL SPECIFICATION

1. INTRODUCTION

1.1. Motivation

The Data Transfer Protocol is designed for use as a highly reliable

Node-to-node protocol between nodes in the virtual networks, and in

interconnected systems of such networks. The data transfer protocol

provides for transmitting blocks of data called datagrams from sources

to destinations, where sources and destinations are nodes identified by

fixed length ip addresses and hostname. The data transfer protocol also

provides for fragmentation and reassembly of long datagrams, if

necessary, for transmission through "small packet" networks.

1.2. Scope

The data transfer protocol is specifically limited in scope to provide

the functions necessary to deliver a package of bits (a data transfer

datagram) from a source to a destination over an interconnected system

of virtual networks. There are no mechanisms to augment end-to-end

data reliability, flow control, sequencing, or other services commonly

found in node-to-node protocols. The data transfer protocol can

capitalize on the services of its supporting virtual networks to

provide various types and qualities of service.

1.3. Operation

The data transfer protocol implements two basic functions:

addressing and fragmentation.

The data transfer modules use the addresses carried in the data transfer header to transmit data transfer datagrams toward their destinations.

The selection of a path for transmission is called routing.

The data transfer modules use fields in the data transfer header to

fragment and reassemble data transfer datagrams when necessary

for transmission through "small packet" networks.

The model of operation is that a data transfer module resides in each node engaged in internet communication and in each gateway that

interconnects virtual networks. These modules share common rules for

interpreting address fields and for fragmenting and assembling

data transfer datagrams. In addition, these modules (especially in

gateways) have procedures for making routing decisions and other

functions.

The data transfer protocol treats each data transfer datagram as an

independent entity unrelated to any other data transfer datagram.

There are no connections or logical circuits (virtual or otherwise).

The data transfer protocol uses three key mechanisms in providing its

service: Type of Service, Time to Live, and Header Checksum.

The Type of Service is used to indicate the quality of the service

desired. The type of service is an abstract or generalized set of

parameters which characterize the service choices provided in the

networks that make up the internet. This type of service indication

is to be used by gateways to select the actual transmission parameters

for a particular network, the network to be used for the next hop, or

the next gateway when routing an internet datagram.

The Time to Live is an indication of an upper bound on the lifetime of

an internet datagram. It is set by the sender of the datagram and

reduced at the points along the route where it is processed. If the

time to live reaches zero before the internet datagram reaches its

destination, the internet datagram is destroyed. The time to live can

be thought of as a self destruct time limit.

The Header Checksum provides a verification that the information used

in processing data transfer datagram has been transmitted correctly.

The data may contain errors. If the header checksum fails, the data transfer datagram is discarded at once by the entity which detects the error.

2. OVERVIEW

2.1. Relation to Other Protocols

The following diagram illustrates the place of the data transfer

protocol in the protocol hierarchy:

+------+ +-----+ +-----+ +-----+

|Telnet| | FTP | | TFTP| ... | ... |

+------+ +-----+ +-----+ +-----+

| | | |

+-----+ +-----+ +-----+

| TCP | | UDP | ... | ... |

+-----+ +-----+ +-----+

| | |

+--------------------------+----+

| Data transfer Protocol & ICMP |

+--------------------------+----+

|

+---------------------------+

| Local Network Protocol |

+---------------------------+

Protocol Relationships

Figure 1.

Data transfer protocol interfaces on one side to the higher level

Host-to-host protocols and on the other side to the local network

protocol.

2.2. Model of Operation

The model of operation for transmitting a datagram from one

application program to another is illustrated by the following

scenario:

We suppose that this transmission will involve one intermediate

Gateway(Router).

The sending application program prepares its data and calls on its

local data transfer module to send that data as a datagram and passes

the destination address, hostname and other parameters as arguments

of the call.

The data transfer module (Pc program in our case) prepares a datagram header and attaches the data to it. The data transfer module determines a local virtual network address for this data transfer address, in this case it is the address of a gateway.

It sends this datagram and the local virtual network address to the local network interface (Router) using TCP Socket.

The Router update the local network header, then it strips off this

header, and determines from the data transfer address and hostname

that the datagram is to be forwarded to another node in a second network.

The data transfer module determines a local net address for the

destination node. It calls the socket specify to communicate with that

virtual network to send the datagram.

At this destination host the datagram is stripped of the packet

header then checked it then get the data.

Application Application

Program Program

\ /

Data transfer Module data transfer Module data transfer Module

\ / \ /

LNI-1 LNI-1 LNI-2 LNI-2

\ / \ /

Local Virtual Network 1 Local Virtual Network 2

Transmission Path

Figure 2

2.3. Function Description

The function or purpose of data transfer Protocol is to move datagrams

through an interconnected set of networks. This is done by passing the datagrams from one data transfer module to another until the

destination is reached. The data transfer modules reside in hosts and

gateways in the internet system. The datagrams are routed from one data transfer module to another through individual networks based on the

interpretation of an data transfer address. Thus, one important

mechanism of the data transfer protocol is the data transfer address.

In the routing of messages from one data transfer module to another,

datagrams may need to traverse a network whose maximum packet size is

smaller than the size of the datagram. To overcome this difficulty, a

fragmentation mechanism is provided in the data transfer protocol.

Fragmentation

Fragmentation of an internet datagram is necessary when the size of

the packet is bigger than the size allowed.

An internet datagram can be marked "don't fragment." Any internet

datagram so marked is not to be internet fragmented under any

circumstances. If internet datagram marked don't fragment cannot be

delivered to its destination without fragmenting it, it is to be

discarded instead.

The internet fragmentation and reassembly procedure needs to be able

to break a datagram into an almost arbitrary number of pieces that

can be later reassembled. The receiver of the fragments uses the

identification field to ensure that fragments of different datagrams

are not mixed. The fragment offset field tells the receiver the

position of a fragment in the original datagram. The fragment

offset and length determine the portion of the original datagram

covered by this fragment. The more-fragments flag indicates (by

being reset) the last fragment. These fields provide sufficient

information to reassemble datagrams.

The identification field is used to distinguish the fragments of one

datagram from those of another. The originating protocol module of

an internet datagram sets the identification field to a value that

must be unique for that source-destination pair and protocol for the

time the datagram will be active in the internet system.

The originating protocol module of a complete datagram sets the

more-fragments flag to zero and the fragment offset to zero.

To fragment a long internet datagram, an internet protocol module

(for example, in a gateway), creates two new internet datagrams and

copies the contents of the internet header fields from the long

datagram into both new internet headers. The data of the long

datagram is divided into two portions. The more-fragments flag is

set to one. The second portion of the data is placed in the second new internet datagram, and the total length field is set to the length of the second datagram. The more-fragments flag is set to zero.

This procedure can be generalized for an n-way split, rather than

the two-way split described.

To assemble the fragments of an internet datagram, an internet protocol module (for example at a destination host) combine internet datagrams that all have the same value for the four fields:

identification, source, destination, and protocol. The combination

is done by placing the data portion of each fragment in the relative

position indicated by the fragment offset in that fragment's

internet header. The first fragment will have the fragment offset

zero, and the last fragment will have the more-fragments flag reset

to zero.

The fields which may be affected by fragmentation include:

(1) options field

(2) more fragments flag

(3) fragment offset

(4) internet header length field

(5) total length field

(6) header checksum

3. SPECIFICATION

3.1. Data Transfer Header Format

A summary of the contents of the Data Transfer header follows:

0 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

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|Version| DTHL |Type of Service| Total Length |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

| Identification | Flags |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

| Time to Live | Protocol | Header Checksum |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

| Source Ip Address |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

| Destination Ip Address |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

| Source HostName | Destination Hostname |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

Example Data Transfer Datagram Header

Figure 4.

Note that each tick mark represents one bit position.

Version:

The Version field indicates the format of the Data Transfer header.

This document describes version 4+.

DTHL:

Data transfer Header Length (DTHL) is the length of the Data Transfer

header in 32 bit words, and thus points to the beginning of the data.

Note that the minimum value for a correct header is 6, so

the minimum size is 24B (with no option filed)

and the Maximum is 64B.

Type of Service:

The Type of Service provides an indication of the abstract

parameters of the quality of service desired. These parameters are

to be used to guide the selection of the actual service parameters

when transmitting a datagram through a particular network. Several

networks offer service precedence, which somehow treats high

precedence traffic as more important than other traffic (generally

by accepting only traffic above a certain precedence at time of high

load).

Bits 0-2: Precedence.

Bit 3: 0 = Normal Delay.

Bits 4: 0 = Normal Throughput.

Bits 5: 0 = Normal Reliability.

Bit 6-7: Reserved for Future Use.

0 1 2 3 4 5 6 7

+-----+-----+-----+-----+-----+-----+-----+-----+

| | | | | | |

| PRECEDENCE | D | T | R | 0 | 0 |

| 000 | 0 | 0 | 0 | | |

+-----+-----+-----+-----+-----+-----+-----+-----+

Precedence alwayse Routine

000 - Routine

Total Length:

Total Length is the length of the datagram, measured in octets,

including data transfer header and data.

This field allows the length of a datagram to be up to 65,535 octets.

Such long datagrams are impractical for most nodes and virtual networks.

Identification:

An identifying value assigned by the sender to aid in assembling

the fragments of a datagram.

Flags:

Various Control Flags.

Bit Value: (MF) 0 = Last Fragment, 1 = More Fragments.

Time to Live:

This field indicates the maximum time the datagram is allowed to

remain in the internet system. If this field contains the value

zero, then the datagram must be destroyed. This field is modified

in data transfer header processing. The time is measured in units

of seconds, but since every module that processes a datagram must

decrease the TTL by at least one even if it process the datagram in

less than a second, the TTL must be thought of only as an upper

bound on the time a datagram may exist. The intention is to cause

undeliverable datagrams to be discarded, and to bound the maximum

datagram lifetime.

Protocol:

This field indicates the TCP protocol which will be the next level

protocol used in the data portion of the internet datagram .The value

for this protocol is specified in its "Assigned Number" which is 6.

Header Checksum:

A checksum on the header only. Since some header fields change

(e.g., time to live), this is recomputed and verified at each point

that the internet header is processed.

The checksum algorithm is:

The checksum field is the 16 bit one’s complement of the one’s

complement sum of all 16 bit words in the header. For purposes of

computing the checksum, the value of the checksum field is zero.

This is a simple to compute checksum and experimental evidence

indicates it is adequate, but it is provisional and may be replaced

by a CRC procedure, depending on further experience.

Source Ip Address:

The source address.

Destination Ip Address:

The destination address.

Source Hostname:

The source port.

Destination HostName:

The destination Port.

References

[1] RFC 791 INTERNET PROTOCOL

[2] Geeksforgeeks

[3] Info 402 Inter Network and Network Security Course in

Lebanese University Faculty of Science.