Abstract

A heavy-weight version of the UDP protocol, intended to provide simple fragmentation mechanism with Internet Protocol "Five Fields", along with a stronger checksum at a cost of somewhat higher overhead.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Introduction

This Fragmented Datagram Protocol (FDP) is defined to make datagram mode communication possible in the computing environment, where fragmented transmission and reception are needed.
FDP conceptually is a heavy-weight UDP if you will.

Format
------

Fragmented Datagram Header Format

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                                                               |
 |                       CRC64c Checksum                         |
 |                                                               |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
12 |                                                               |
 | M|       Fragment Offset       |                               |
 |   |                             |                               |
 |   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
16 |                       Fragment Identifier                     |
 |                                                               |
 |                                                               |
 |                          data octets ...                      |
 |                                                               |
 |                                                               |
 |                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
+---------------------+---------------------+---------------------+---------------------+
| (bytes)             | (bytes)             | (bytes)             |
```

Fields
------

CRC64c Checksum (64-bit) -- 64-bit checksum. See CRC64 Checksum.

M flag

1 = more fragments; 0 = last fragment.

Fragment Offset

15-bit unsigned integer. The offset, in 8-byte units, of the data following this header, relative to the start of the datagram.

Fragment Identifier

48-bit identifier is needed to distinguish fragments from different datagrams.

FDP segment must only be assembled when the six-tuple matches:
1. Source and Destination IP address
2. Source and Destination ports / Flow
3. Protocol
4. Same Fragment Identifier

*NOTE: Fragmentation is incompatible with very big Jumbograms.
(IP packets sized > 256 KiB)

CRC64 Checksum
--------------

Consists of parts of the IP header, the FDP header, and the data, padded with zero bytes at the end (if necessary) to make a multiple of two bytes.
This checksum procedure is similar to what is used in TCP.64.

If the computed checksum is zero, it is transmitted as all ones. An all zero transmitted checksum value means that the transmitter generated no checksum (for debugging or for higher level protocols that don’t care or for ILNP purposes).
Design note:

Checksum in FDP covers *only* one-single fragment, not the entire packet. This is done to simplify processing by middleboxes, so for them it will look like a Checksum covers a packet, they don’t need to re-assemble the fragments into a packet to change or validate checksum anymore.

This is done particularly for load-balancers and NAT Routers.

Additionally, because in IP-FF ports are at layer 3, are visible, which further simplifies middlebox design and implementation.

Implied Items
--------------

Ports, while logically belong to Transport Layer, have moved to the IP layer in IP-FF, and this is to speed-up certain decisions.

Source Port is an optional field, when meaningful, it indicates the port of the sending process, and may be assumed to be the port to which a reply should be addressed in the absence of any other information. If not used, a value of zero is inserted.

Destination Port has a meaning within the context of a particular internet destination address.

Length of the data is taken from upper level IPFF protocol.

The pseudo header conceptually prefixed to the FDP header contains the source address, the destination address, the protocol, and the FDP length. This information gives protection against misrouted datagrams.

The header structure must be taken from [IPFF] document RFC.

Protocol Application
---------------------

Existing UDP applications may be migrated over to FDP.

Protocol Number
-----------------

This is protocol number (to be assigned by IANA), when used in the Internet Protocol, Five Fields.

Explaining Fragmentation
-------------------------

Fragmented Datagram Protocol works in a similar way to IPv6 fragmentation, with the added bonus that all the IP-FF extensions, checksums and ports are part of each fragment, which adds simplicity. (at a cost of some overhead)

In order to send a packet that is too large to fit in the MTU of the path to its destination, a source node may divide the packet into fragments and send each fragment as a separate packet, to be reassembled at the receiver.

For every packet that is to be fragmented, the source node generates an Identification value. The Identification must be different than
that of any other fragmented packet sent recently* with the same Source Address, Destination Address, Source Port and Destination Port.

* "recently" means within the maximum likely lifetime of a packet, including transit time from source to destination and time spent awaiting reassembly with other fragments of the same packet. However, it is not required that a source node know the maximum packet lifetime. Rather, it is assumed that the requirement can be met by maintaining the Identification value as a simple, 48-bit, "wrap-around" counter, incremented each time a packet must be fragmented. It is an implementation choice whether to maintain a single counter for the node or multiple counters, e.g., one for each of the node’s possible source addresses, or one for each active (source address, destination address, and ports) combination.

The following rules govern reassembly:

An original packet is reassembled only from fragment packets that have the same Source Address, Destination Address, Source Port, Destination Port, and Fragment Identification.

If all the fragments arrived, with correct checksums, application should receive a complete packet. Else packet must be dropped.

Acknowledgement
-----------------

Originally UDP was written by J.Postel as RFC-768, modified by Alexey Eromenko for the purposes of Internet Protocol "Five Fields".

FDP borrows heavily from IPv6-style-fragmentation by end nodes [RFC-2460].

Author Contacts

Alexey Eromenko
Israel

Skype: Fenix_NBK_
EMail: a14321@gmail.com
Facebook: https://www.facebook.com/technologov

INTERNET-DRAFT
Alexey

expiration date: 2017-03-29