R-Type Group A. Levy

Request for Comments: 4242 R-Type, Epitech

Category: Video Games November 2019

R-Type Protocol (RTP) Transport over UDP.

Abstract

This document describes how the R-Type Protocol (RTP), works using and   
 UDP connection.

This protocol requires use of a R-Type server and at least one R-Type   
 client.

This protocol is using binary communication with the simplest client   
 being any socket program capable of connecting to the server.

The RTP was developed during the R-Type student project.

Status of This Memo

This document is the official specification of the R-Type Protocol,

and defines an Experimental Protocol for the R-Type community.

Discussion and suggestions for improvement are requested.

Distribution of this memo is unlimited.

Copyright Notice

Copyright (c) 2019 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 in effect on the date of

publication of this document (http://trustee.ietf.org/license-info).

Please review these documents carefully, as they describe your rights

and restrictions with respect to this document.

Table of Contents

1. Introduction...........................................................

1.1 Conventions Used in This Document...............................

2. Session Management ....................................................  
3. Message Exchange ......................................................

4. Network Protocol ......................................................

4.1 Types ..........................................................

4.2 Packet .........................................................

5. References ............................................................

6. Author's Address ......................................................

[1](https://tools.ietf.org/html/rfc1459#section-1). Introduction

The R-Type protocol has been developed on systems using the TCP   
 network protocol and UDP network protocol, although there is no   
 requirement that this remain the only sphere in which it operates.

IP is described in [RFC0791]. TCP is described in [RFC0793].

UDP is described in [RFC0768].

1.1. Conventions Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",

"SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this

document are to be interpreted as described in [RFC2119].

2. Session Management

The RTP session management facilities onto the UDP and TCP service is  
 straightforward. For TCP, the RTP session first requires a creation of  
 a TCP connection between two peers, one that initiates the connection   
 request and one that responds to the connection request. The initiating   
 peer is called the "client", and the responding peer is called the   
 "server". An RTP server MUST listen for UDP or TCP connection requests   
 on a standard TCP or UDP port.

An RTP session is normally ended by the client issuing and RTP   
 Disconnected packet. A server receiving and RTP Disconnected Packet  
 MUST end the RTP session, close the UDP connection if the socket was in  
 UDP mode, and broadcast this disconnection on all client in the party.  
 In UDP mode, the client MAY end the RTP session by closing the socket.

3. Message Exchange

In UDP, the user is connected in a party when the RTP server returns 0   
 in the Error packet.

The user MUST be connected using the Connect packet to take part in   
 the parties.  
  
 After this, both client and server can initiate a RTP binary command.  
 In some commands, the server SHOULD NOT return a response for avoid   
 flooding the clients. For farthing details, go to the 4. Section.

The RTP packet’s command MUST be struct-packed for better transfer.

4. Network Protocol

4.1 Types

**CommandType:**

Error = -1

Connection = 1

Positive\_response = 2

All\_player\_conneted = 3

Change\_position = 4

Change\_ships = 5

#4 = envoie du nombre de player in game

#5 = envoie du deplacement de mon player

#6 = envoie du deplacement d'un autre player

#7 = envoie du tire de mon player

#8 = envoie du tire d'un autre player

#9 = mon player a gagner 1 point

#10 = la fiche de score

**ObjectType:**

Ship:

Missile:

Monster:

**ShipType:**

Standard: 1

Second Player : 2

Third Player : 3

Fourth Player : 4

Change\_ships = 5

ScoreType :

AddPoint = 9

**MissileType**:

Standard: 0

**GameStatusType:**

Waiting: 0

Playing : 1

4.2 Packet

**Error:**

Struct server\_buffer

This command can be invoqued by the client and the server.

The "cmdType" MUST be {CMD}.  
 The "server\_buffer" MUST be:  
  
 0: OK  
 -1: NOT FOUND

**Connection - client:**

[0] [sizeof(msg)] [1] [(nom du joueur)]

This command can be invoqued only by the client.

The "size" MUST be the size of the "msg" field.  
 The "msg" MUST be "port;ip".

On Success : 0

On error : -1

**Connection - server:**

[0] [sizeof(msg)] [2] [id du joueur]

The "size" MUST be the size of the "msg" field.  
The "msg" MUST be "port;ip".

On Success = 0

On error = -1

On case error the command is :

[0] [sizeof(msg)] [-1] [msg d'erreur]

5. References

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791,

September 1981.

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7,

RFC 793, September 1981

[RFC0768] Postel, J., "User Datagram Protocol", RFC 793,

28 August 1980

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate

Requirement Levels", BCP 14, RFC 2119, March 1997.

6. Author's Address

Antoine Levy

R-Type, Epitech.

24 rue pasteur, 94270

Le Kremlin-Bicêtre.

Email: antoine1.levy@epitech.eu