Network Protocol For R-Type

January 20, 2013

Laurent Andral
Bertrand Boyer
Erick Clarivet
Nataniel Martin
Clément Petit
Camille Tolsa

EPITECH.
L'ECOLE DE L'EXPERTISE INFORMATIQUE

Contents

List of Figures

Chapter 1

Introduction

This document describes the network protocol for R-Type Project.
It describes how the clients and the server will communicate together.
This protocol is only meant for the communation client-server type.

Status of this memo

This document specifies a standart track protocol for the community of players.
Distribution of this memo is unlimited.

Copyright Notice: This project is Open-Source.

Conventions

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].

Internationalization considerations

This document does not introduce or present any internationalization or localization issues.

Security considerations

An injection of packets in UDP can harm the integrity of the system but it MUST provide the right MAGIC code and the right CHECKSUM hash.
And this is 0.0001% likely to happen.

Chapter 2

Protocol packet

Packet datagram

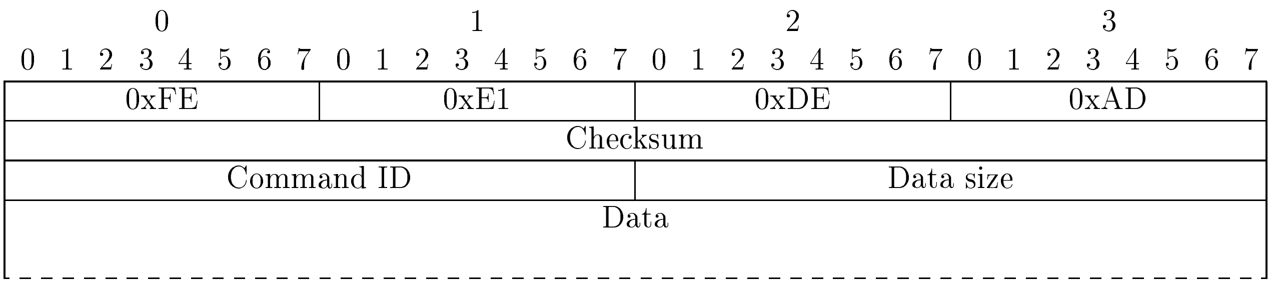

Figure 2.1: R-Type packet's datagram.

Description

- The first 32 bits are used as a magic number to make sure the packet is conform (uint32_t).
- The next 32 bits, "Hashcode" to improve security of the packet and to check the packet integrity (uint32_t).
- The next 32 bits are used to define the timestamp of the packet, in order to check if the packet is to old, irrelevant (uint32_t).
- The next 16 bits are used for the ID command. It defines the incoming data pattern (uint16_t).
- The next 16 bits are used for the Data Size (uint16_t).
- The next XX bits defined by the data size are reserved for the strutured message.

Chapter 3

List of commands

This chapter presents all commands authorized between client and server communication. All unknown command will be ignored.

3.1 Client $\rightarrow$ server

3.1.1 Menu commands - TCP

- AUTHENTIFICATION
  - The client wants to authentify.
  - ID of command: 000.
  - Message: [Login:char[32]].
- SHOW_ROOM
  - Request of the list of rooms.
  - ID of command: 001.
  - Message: [empty].
- CREATE_ROOM
  - The client creates a new room.
  - ID of command: 002.
  - Message: [NameRoom:char[32]][NumberOfMaxPlayers:int][Difficulty:int][IDMap:int].
- JOIN_ROOM
  - The client joins the room.
  - ID of command: 003.
  - Message: [NameRoom:char[32]].

- LEAVE_ROOM
  - The client leaves the room.
  - ID of command: 004.
  - Message: [empty].
- LAUNCH_GAME
  - The client, owner of the room, starts the game.
  - ID of command: 005.
  - Message: [NameRoom:char[32]].

3.1.2 In game commands - UDP

- MOVEMENT
  - The client moves.
  - Id of command: 100.
  - Message: [MovementType:int].
- SHOOTING
  - The client shoots.
  - ID of command: 101.
  - Message: [IDGun:int].
- ASK_DESCRIBE_ENTITY
  - The client ask to describe an entity.
  - ID of command: 102.
  - Message: [IDEntity:int].
- ASK_PLAYER_SCORE
  - The client ask a players' score.
  - ID of command: 103.
  - Message: [IDEntity:int].
- ASK_PLAYER_LIFE
  - The client ask a players' life.
  - ID of command: 104.
  - Message: [IDEntity:int].

3.2 Server → client

3.2.1 Menu commands - TCP

- AUTHENTIFICATION_OK
  - Success of the authentification.
  - ID of command: 200.
  - Message: [empty].
- AUTHENTIFICATION_KO
  - Failure of the authentification.
  - ID of command: 201.
  - Message: [ErrorMessageID:int].
- ROOM_DESCRIPTION
  - The server sends a room description.
  - ID of command: 202.
  - Message: [NameRoom:char[32]][NumberOfMaxPlayers:int] [NumberOfCurrentPlayers:int][Difficulty:int][IDMap:int]
- ROOM_PLAYER_LIST
  - The list of player's name in the room.
  - ID of the command: 203.
  - Message: [NameRoom:char[32]][Name1:char[32]] [Name2:char[32]][Name3:char[32]][Name4:char[32]]
- CREATE_ROOM_OK
  - Success of the room creation.
  - ID of command: 204.
  - Message: [empty].
- CREATE_ROOM_KO
  - Success of the room creation.
  - ID of command: 205.
  - Message: [ErrorMessageID:int].

- JOIN_ROOM_OK
  - Success of entering in the room.
  - ID of command: 206.
  - Message: [empty].
- JOIN_ROOM_KO
  - Failure of entering in the room.
  - ID of command: 207.
  - Message: [ErrorMessageID:int].
- LEAVE_ROOM_OK
  - Success of leaving the room.
  - ID of command: 208.
  - Message: [empty].
- LEAVE_ROOM_KO
  - Failure of leaving the room.
  - ID of command: 209.
  - Message: [ErrorMessageID:int].
- LAUNCH_GAME_OK
  - Success of launching the game.
  - ID of command: 210.
  - Message: [empty].
- LAUNCH_GAME_KO
  - Failure of launching the game.
  - ID of command: 211.
  - Message: [ErrorMessageID:int].

3.2.2 In game commands - UDP

- START_GAME
  - The server launch the server.
  - ID of command: 300.
  - Message: [empty].
- SPAWN_ENTITY
  - An entity has spawn on the map.
  - ID of command: 301.
  - Message: [IDEntity:int][IDType:int][PosX:int][PosY:int].
- DESTROY_ENTITY
  - An entity is destroyed.
  - ID of command: 302.
  - Message: [IDEntity:int].
- MOVE_ENTITY
  - An entity moves.
  - ID of command: 303.
  - Message: [IDEntity:int][PosX:int][PosY:int][MovementType:int].
- LIFE_ENTITY
  - The server send the life of the entity.
  - ID of command: 304.
  - Message: [IDEntity:int][Life:int].
- COLLISION
  - Collision between two entities.
  - ID of command: 305.
  - Message: [IDEntity1:int][IDEntity2:int].
- DESCRIPTION_ENTITY
  - The server send the description of the entity.
  - ID of command: 306.
  - Message: [IDEntity:int][IDType:int][Life:int][PosX:int][PosY:int].

- ENTITY_SCORE
  - The server sends the score of the entity.
  - ID of command: 307.
  - Message: [IDEntity: int][Score:int].
- PLAYER_DISCONNECT
  - The server sends the player id who has just disconnected.
  - ID of command: 308.
  - Message: [IDEntity:int].
- END_GAME
  - The game has ended.
  - ID of command: 309.
  - Message: [MessageID:int].

Chapter 4

Communication Client → Server

This section presents actions and reactions from client and server for each packets sent. And it also gives the mains reasons for eache sending.

4.1 TCP

4.1.1 Actions and reactions

- Green means the action succeed and red it failed.
- n is the number of room on the server (in game or not).

Action	**Reaction**
AUTHENTIFICATION	AUTHENTIFICATION_OK AUTHENTIFICATION_KO
SHOW_ROOM	n * ROOM_DESCRIPTION n * ROOM_PLAYER_LIST
CREATE_ROOM	CREATE_ROOM_OK CREATE_ROOM_KO
JOIN_ROOM	JOIN_ROOM_OK JOIN_ROOM_KO
LEAVE_ROOM	LEAVE_ROOM_OK LEAVE_ROOM_KO
LAUNCH_GAME	LAUNCH_ROOM_OK LAUNCH_ROOM_KO

Figure 4.1: Actions and reactions in TCP.

4.1.2 Reasons

- AUTHENTIFICATION
  - Is the login already exist ?
    - Yes → send AUTHENFICATION_KO.
    - No → send AUTHENFICATION_OK.
- SHOW_ROOM
  - For each room, send ROOM_DESCRIPTION to the client.
  - For each room, send ROOM_PLAYER_LIST to the client.
- CREATE_ROOM
  - Is the room's name already exist ?
    - Yes → send CREATE_ROOM_KO.
    - No → send CREATE_ROOM_OK.
- JOIN_ROOM
  - Is the room already in a game ?
    - Yes → send JOIN_ROOM_KO.
    - No. Is the room full ?
      - Yes → send JOIN_ROOM_KO.
      - No → send JOIN_ROOM_OK.
- LEAVE_ROOM
  - Send ROOM_DESCRIPTION and ROOM_PLAYER_LIST to all players in the room.
  - Is the client the last in the room ?
    - Yes → Delete the room (server side).
- LAUNCH_ROOM
  - Is the server ready for a new game ?
    - Yes → send LAUNCH_GAME_OK to all players in the room.
    - No → send LAUNCH_GAME_KO to the client.

4.2 UDP

4.2.1 Actions and reactions

Here is only presented actions which have an answer from the server.

Action	**Reaction**
MOVEMENT	MOVE_ENTITY
SHOOTING	SPAWN_ENTITY
ASK_DESCRIBE_ENTITY	DESCRIPTION_ENTITY
ASK_PLAYER_SCORE	ENTITY_SCORE
ASK_PLAYER_PLAYER_LIFE	LIFE_ENTITY

Figure 4.2: Actions and reactions in UDP.

These events could be sent by the server without any request from the client.

Event	**Reason**
SPAWN_ENTITY	A new entity spawn on the map.
DESTROY_ENTITY	An entity disapears or is killed.
MOVE_ENTITY	An entity is moving (e.g. a player is pressing a movement key).
LIFE_ENTITY	An ennemy or the player has been shot.
COLLISION	Two entity are touching each other.
DESCRIPTION_ENTITY	After a spawned entity, the server can send this entity description.
ENTITY_SCORE	After an ennemy is killed, the new score is sent to display.
PLAYER_DISCONNECTED	The player leaves the game, the client doesn't need to display it anymore.

Figure 4.3: Events sent by the server.

4.3 Order of actions

This part explains the order of actions for a game.

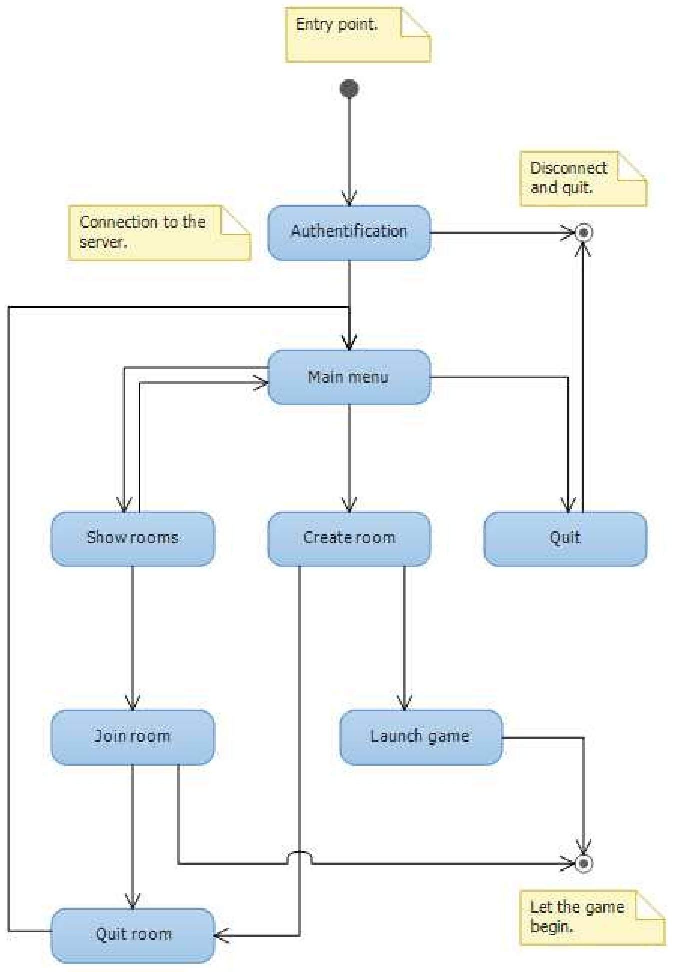

Figure 4.4: Client side, before the game.

Chapter 5

References

[RFC 2119]
[RFC 2223]

Chapter 6

Authors adresses

Laurent Andral	(andral_l)	laurent.andral@epitech.eu
Bertrand Boyer	(boyer_b)	bertrand.boyer@epitech.eu
Erick Clarivet	(clariv_e)	erick.clarivet@epitech.eu
Nataniel Martin	(mart_n)	nataniel.martin@epitech.eu
Clément Petit	(petit_c)	clement.petit@epitech.eu
Camille Tolsa	(tolsa_c)	camille.tolsa@epitech.eu

R-Type Group Project
Epitech Toulouse
19 rue Bayard, 31000 Toulouse, France