Antoine PLASKOWSKI : antoine.plaskowski@epitech.eu

Spider Network Protocol v1

Table of Contents

1. Introduction

This document describe the network protocol used in the "Spider"project.

This protocol version is 1.

All the integers (size, etc...) are represented in little-endian.

2. Versionning

This protocol is versionned in order to allow implementations to keep a retro-compatibility against older protocol versions.

The protocol version is just an integer starting at "1", which is increased when a major change (non retro-compatible) is bringed in this specification.

Protocol version	Date	Description
1	2015-10-18	First version

Table 0: Version

3. Basic structures

This section will define some basic structures used by the network protocol.

All size are in octet.
uintX_t is a unsigned integer of X bit

3.1. Basic fields

This section will define some basic fields used by the packets in the protocol.

3.1.1. String field

The maximum size of a string MUST be "2^8-1"

The string is encoded with ASCII table.

Name	type	Description
length	uint8_t	String length
data	uint8_t[]	String data

Table 1: String field structure

3.1.2. Array field

The maximum size of an array MUST be "2^8-1".

The type of the array items MUST be unique and defined when using the array field.

Name	type	Description
size	uint8_t	Array size
data	T[]	Array data

Table 2: Array field structure

3.2. Packet header

This structure is present before any packet described in this protocol.

Each packet MUST be assigned to an opcode, as defined in each packet description sections.

The maximum size of the data MUST be "2^16-1"

Name	type	Description
opcode	uint8_t	Opcode
id	uint8_t	Id of packet
size	uint16_t	Size of data

Table 3: Packet header structure

* opcode : It's identify what is the packet type
* id : It's use to identify a packet.
* size : It's the size of the packet without the header.

4. Protocol packets definition

4.1. Result packet

The "Result" packet is used in response of any packet, containing the result of the operations.

Name	Value
opcode	0
id	?
size	2

Table 4: Result packet header structure

Name	type	Description
error	uint8_t	Error code
id	uint8_t	coresponding packet

* error : It's the error code
* id : It's the id from the packet corresponding packet

Table 5: Result packet data structure

Error code	Description
0	No error
1	Ignored
2	Unknown error
3	Client already started
4	Client already stopped
5	Client already muted
6	Client already unmuted
7	Invalid command
8	Invalid keyboard input
9	Invalid mouse input
10	Wrong protocol version
11	Wrong Mac address
12	Connect fail
13	Disconnect fail

Table 6: error code definition

4.2. MAC address packet

The "MAC address" packet is used to send the mac address of the client machine.

Each client must send it identify himself.

Name	Value
Opcode	1
id	?
size	6

Table 7: Mac address packet header structure

Name	Data type	Description
mac	uint8_t[6]	The MAC address of the client machine

Table 8: MAC address packet data structure

4.3. Version packet

The "Version" packet is used by a client to ask if his version of spider protocol is implemented.

Name	Value
opcode	2
id	?
size	1

Table 9: Version packet header structure

Name	Data type	Description
version	uint8_t	Version of spider protocol

Table 10: Result packet data structure

4.4. Connect packet

The "Connect" packet is used by a client to connect to a server.

If the server has enough information to identify the client, he accept the connection.

Name	Value
opcode	3
id	?
size	0

Table 11: Connect packet header structure

4.5. Disconnect packet

The "Disconnect" packet is used by a client to disconnect to a server.

This packet must be send if a client want to disconnect from server.

Name	Value
opcode	4
id	?
size	0

Table 12: Disconnect packet header structure

4.6. ServerCmd packet

The "ServerCmd" is used by the server to control the client's behavior.

Name	Value
Opcode	5
id	?
Data size	1

Table 13: Connect packet header structure

Name	Data type	Description
command	uint8_t	Command code

Table 14: Connect packet data structure

Command code	name	Description
0	start	start the client from handling inputs
1	stop	stops the client from handling inputs
2	mute	start the client from sending inputs
3	unmute	stops the client from sending inputs

Table 15: Command code definition

4.7. ClientLog packet

The "ClientLog" packet is used by the client to send log messages to the server, for debug or other.

Name	Value
Opcode	6

id	?
size	?

Table 16: ClientLog packet header structure

Name	type	Description
msg	array[string]	The message to send

Table 17: ClientLog packet data structure

4.8. Ping packet

When a "Ping" packet is sent, a "Pong" packet MUST be sent back to the sender.

This packet is used to check if the other end of the connection is able to process packets or not.

Name	Value
Opcode	7
id	?
size	0

Table 18: Ping packet header structure

4.9. Pong packet

Name	Value
Opcode	8
id	?
size	0

Table 19: Pong packet header structure

4.10. Keyboard packet

Name	Value
Opcode	9
id	?
size	?

Table 20: Keyboard packet header structure

Name	Type	Description

data	array[keyboard]	Input data array

Table 21: Keyboard packet data structure

name	Type	Description
second	int64_t	The timestamp of the event
nano	int64_t	The timestamp of the event
event	string	Type of the keyboard event
name	string	Name of the key
process	string	proccesus name

Table 22: Keyboard structure

* "name" : The readable name of the key ("a", "b", ";").

* "event" :

event code	name	Description
0	pressed	when the key is press
1	released	when the key is releases

Table 23: Event code definition

4.11. Mouse input

Name	Value
opcode	10
id	?
size	?

Table 24: Mouse packet header structure

Name	type	Description
data	array[mouse]	Input data array

Table 25: Mouse packet data structure

name	Field type	Description
second	int64_t	The timestamp of the event
nano	int64_t	The timestamp of the event
pos_x	uint32_t	Position x of the mouse
pos_y	uint32_t	Position y of the mouse
amount	uint64_t	Amount
event	string	Type of the button event

name	string	Name of the button
process	string	proccesus name

Table 26: Mouse structure

* "name" : The name of the button (1, 2, 3, ...)
This name MUST be a readable name for left, middle and right buttons.

* "type" :

event code	name	Description
0	pressed	when the key is press
1	released	when the key is releases
2	click	when the key is click
3	scroll	when the key is scroll
4	move	when the mouse is move

Table 27: Type code definition

* "pos_x" and "pos_y" : The position of the mouse when the event has been triggered.

* "amount" : The scroll amount for the "scroll" event.
A value under 0 means that the scroll is "up" and a value over 0
means that the scroll is "down" (in the screen PoV)