XEngine Authorize Service Docment

目录

[XEngine Authorize Service Docment 1](#_Toc2646)

[Preface 4](#_Toc15748)

[Reader 4](#_Toc11356)

[Overview 4](#_Toc791)

[Associate Module 4](#_Toc8143)

[一 Technical structure 4](#_Toc1760)

[1.1 Directory Structure 4](#_Toc31279)

[1.2 Program Directory 4](#_Toc17966)

[二 Configure Env 5](#_Toc21020)

[2.1 WINDOWS 5](#_Toc31107)

[2.1.1 Configure Environment 5](#_Toc4327)

[2.1.2 complie and run 5](#_Toc5215)

[2.2 LINUX 6](#_Toc5566)

[2.2.1 Evnironment Configure 6](#_Toc29751)

[2.2.2 Complie and Run 6](#_Toc24381)

[2.3 Version Requirements 7](#_Toc16463)

[2.3.1 System Version 7](#_Toc22108)

[2.3.2 Software Version 7](#_Toc18778)

[三 TCP Interface Protocol 7](#_Toc28191)

[3.1 Delete Protocol 7](#_Toc28419)

[3.1.1 Request 7](#_Toc9584)

[3.1.2 Reply 8](#_Toc716)

[3.2 Register Protocol 9](#_Toc9475)

[3.2.1 Request 9](#_Toc6891)

[3.2.2 Reply 9](#_Toc16703)

[3.3 Login Protocol 10](#_Toc7436)

[3.3.1 Request 10](#_Toc10363)

[3.3.2 Reply 10](#_Toc18463)

[3.4 Pay Protocol 11](#_Toc2062)

[3.4.1 Request 11](#_Toc32157)

[3.4.2 Reply 11](#_Toc1801)

[3.5 Get Password 12](#_Toc19944)

[3.5.1 Request 12](#_Toc11155)

[3.5.2 Reply 12](#_Toc10415)

[3.6 Get Time 13](#_Toc10293)

[3.6.1 Request 13](#_Toc22914)

[3.6.2 Reply 14](#_Toc14506)

[3.7 Network Verification 14](#_Toc31227)

[3.7.1 Request 14](#_Toc12766)

[3.7.2 Reply 15](#_Toc3616)

[3.8 Get User 15](#_Toc18249)

[3.8.1 Request 15](#_Toc25839)

[3.8.2 Reply 16](#_Toc22478)

[3.9 Set User 16](#_Toc26161)

[3.9.1 Request 16](#_Toc26294)

[3.9.2 Reply 17](#_Toc11)

[3.10 Notification Protocol 17](#_Toc18833)

[3.11 Notification Timeout 17](#_Toc14818)

[四 Websocket Interface Protocol 18](#_Toc6647)

[4.1 user info protocol 18](#_Toc24459)

[4.1.1 Request 18](#_Toc12934)

[4.1.2 Reply 19](#_Toc3812)

[4.2 user authorize 19](#_Toc30859)

[4.2.1 Request 19](#_Toc838)

[4.2.2 Reply 20](#_Toc54)

[4.3 user pay protocol 21](#_Toc5487)

[4.3.1 Request 21](#_Toc18263)

[4.3.2 Reply 21](#_Toc23510)

[4.4 user time protocol 21](#_Toc25317)

[4.4.1 Request 21](#_Toc30343)

[4.4.2 Reply 21](#_Toc7651)

[4.5 fast authorize 22](#_Toc11001)

[4.5.1 Request 22](#_Toc15463)

[4.5.2 Reply 22](#_Toc22398)

[4.6 note protocol 22](#_Toc27590)

[4.6.1 Request 22](#_Toc17207)

[4.6.2 Reply 23](#_Toc28981)

[4.7 timeout protocol 23](#_Toc14010)

[4.7.1 Request 23](#_Toc29279)

[4.7.2 Reply 23](#_Toc27333)

[五 Configure Description 23](#_Toc20416)

[5.1 Basic Configure 23](#_Toc1464)

[5.2 Verication Configure 24](#_Toc1734)

[5.3 Encrypto Configure 24](#_Toc26026)

[5.4 EMail Configure 24](#_Toc517)

[5.5 Database Configure 24](#_Toc9297)

[5.6 Log Configure 24](#_Toc12290)

[appendix 25](#_Toc27424)

[Appendix 1 Type Define 25](#_Toc15714)

[Appendix 2 Protocol Define 25](#_Toc8508)

[Appendix 3 Transformation Definition 25](#_Toc30077)

|  |  |  |  |
| --- | --- | --- | --- |
| File Status：  [ ] Draft  [√] Release | File Name： | XEngine Authorize Service Docment | |
| Be A Version： | V2.3 | |
| Released： | 2021-10-22 | |
| Writer： qyt | | |

# Preface

## Reader

Developer ,tester

## **Overview**

This document contains related technical descriptions and interface definitions

## Associate Module

The service used XEngine as Network Toolki.if you want to use code,you have to installed XEngine

# 一 Technical structure

Used to tcp protocol though basic protoco

Should be bind two port when Start service

## Directory Structure

* XEngine\_Apps:example code path
* XEngine\_Docment:docment path
* XEngine\_Release:complie release path
* XEngine\_Source:code path

## Program Directory

We have two version of the program,source code path in XEngine\_APPService.we recommend you to use the interface version

* XEngine\_AuthorizeApp:WINDOWS Have Interface version
* XEngine\_AuthorizeService:Cross-platform version without interface

# 二 Configure Env

## 2.1 WINDOWS

Need to download XEngine.

Complie and run and debug by vs2019.

Download address:https://gitee.com/xyry/libxengine

https://github.com/libxengine/xengine

### 2.1.1 Configure Environment

Follow the instructions in the XEngine Readme file to execute the script to configure the environment. If succeed, you will see the following values in your system environment variables.



### 2.1.2 complie and run

When you complete with configuration.you can come in code path.open XEngine\_MQServiceApp.sln by vs2019 and choice x86(debug or release) or x64(just release) complie.

If environment not have error.complie is succesed.contrain 4 module and 1 exe program

And you need copy file under XEngine\_Release to your complied dir.next step copy file under XEngine depend Module and jsoncpp module to your complied dir.

If not have error,you can see follow the infomation



Note: You can run the program directly, the system will prompt you what you need, you can directly enter the XEngine directory to search.

## 2.2 LINUX

Linux Just used to XEngine\_AuthorizeService Version without interface

### 2.2.1 Evnironment Configure

If you use linux.you must running on ubuntu(20.04) or centos(8.x)...

After download complete.you can install xengine it by the shell file.

Execute command:sudo XEngine\_LINEnv.sh -i 3

### 2.2.2 Complie and Run

Configure complete.you can complie it.open terminal in you xengine\_storage dir and execute command.

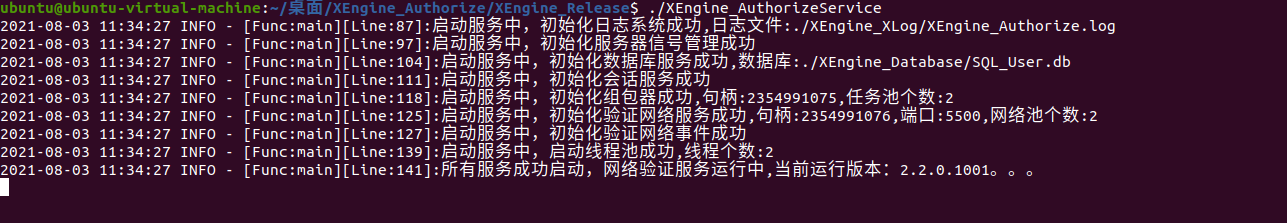
complie:make

install:make FLAGS=InstallAll

clean:make FLAGS=CleanAll

If there is no error.you can see complied XEngine\_AuthorizeService file in XEngine\_Release

You can running at terminal.if there is no error,you can infomation:



## 2.3 Version Requirements

### 2.3.1 System Version

Minimum version requirements:

WINDOWS: win7 sp1

Ubuntu:20.04

Centos:8.x

### 2.3.2 Software Version

Minimum version requirements:

XEngine:V7.18

# 三 TCP Interface Protocol

TCP interface protocol is used for application verification such as APP. It can be used for login, registration, logout, deletion, etc.

Please note: the verification protocol use to reserved field wReserve of the protocol header to reply the client request, telling the client operation is succeeded or failed, 0 means success, other means failure.

wCrypto means encryption and decryption, currently only supports type 4, XCrypto algorithm. If encryption is required, please fill in 4, otherwise please fill in 0

## 3.1 Delete Protocol

Use to delete user.

### 3.1.1 Request

Request a delete user,you need to use the header protocol + register protocol to achieve.

The content of this agreement is used to judge and verify whether the user has the right to delete himself. Of course, with the use of TOKEN, you can also realize the distribution of permissions. Use TOKEN to verify whether this user has the right to delete another user

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_AUTH

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_AUTH\_REQDEL

unPacketSize = sizeof(XENGINE\_PROTOCOL\_USERINFO)

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:

typedef struct tag\_XEngine\_Protocol\_UserReg

{

*TCHAR* tszUserName[64];

*TCHAR* tszUserPass[64];

*TCHAR* tszEMailAddr[64];

*TCHAR* tszLoginTime[64];

*TCHAR* tszCreateTime[64];

\_\_int64x nPhoneNumber;

\_\_int64x nIDNumber;

int nUserLevel;

*int nUserState*;

}XENGINE\_PROTOCOL\_USERINFO, \*LPXENGINE\_PROTOCOL\_USERINFO;

### 3.1.2 Reply

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_AUTH

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_AUTH\_REPDEL

unPacketSize = 0

byVersion = 0

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

## 3.2 Register Protocol

### 3.2.1 Request

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_AUTH

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_AUTH\_REQREGISTER

unPacketSize = sizeof(XENGINE\_PROTOCOL\_USERINFO)

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:

typedef struct tag\_XEngine\_Protocol\_UserReg

{

*TCHAR* tszUserName[64];

*TCHAR* tszUserPass[64];

*TCHAR* tszEMailAddr[64];

*TCHAR* tszLoginTime[64];

*TCHAR* tszCreateTime[64];

\_\_int64x nPhoneNumber;

\_\_int64x nIDNumber;

int nUserLevel;

*int nUserState*;

}XENGINE\_PROTOCOL\_USERINFO, \*LPXENGINE\_PROTOCOL\_USERINFO;

### 3.2.2 Reply

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_AUTH

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_AUTH\_REPREGISTER

unPacketSize = 0

byVersion = 0

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

## 3.3 Login Protocol

The login protocol is the first step that the client needs to do when requesting the server to exchange data

### 3.3.1 Request

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_AUTH

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_AUTH\_REQLOGIN

unPacketSize = sizeof(XENGINE\_PROTOCOL\_USERAUTH)

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:

typedef struct tag\_XEngine\_Protocol\_Auth

{

*CHAR* tszUserName[64];

*CHAR* tszUserPass[64];

ENUM\_PROTOCOLCLIENT\_TYPE enClientType;

ENUM\_PROTOCOLDEVICE\_TYPE enDeviceType;

}XENGINE\_PROTOCOL\_USERAUTH, \*LPXENGINE\_PROTOCOL\_USERAUTH;

### 3.3.2 Reply

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_AUTH

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_AUTH\_REPLOGIN

unPacketSize = 0

byVersion = 0

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

#### 3.3.2.1 Reply Type

The Server reply success or failure can be custom by developer,we are make sure through wReserver,the value on the login protocol,return value means:

* 0:success
* 251:user not found
* 252:password is error
* 253:User does not have permission
* 254:user timeout
* 255:Server internal error

## 3.4 Pay Protocol

The user recharge agreement is recharged through a recharge card, and the recharge card needs to be generated through our serial number module

### 3.4.1 Request

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_AUTH

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_AUTH\_REQPAY

unPacketSize = sizeof(AUTHREG\_PROTOCOL\_USERPAY)

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:

typedef struct

{

*CHAR* tszUserName[XENGINE\_AUTHREG\_SERVICE\_SQL\_MAX\_USERNAME];

*CHAR* tszSerialNumber[128];

}AUTHREG\_PROTOCOL\_USERPAY, \*LPAUTHREG\_PROTOCOL\_USERPAY;

### 3.4.2 Reply

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_AUTH

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_AUTH\_REPPAY

unPacketSize = 0

byVersion = 0

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

## 3.5 Get Password

### 3.5.1 Request

Protocol Header:User And IDNumber and phone number can be empty

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_AUTH

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_AUTH\_REQGETPASS

unPacketSize = sizeof(XENGINE\_PROTOCOL\_USERINFO)

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:

typedef struct tag\_XEngine\_Protocol\_UserReg

{

*TCHAR* tszUserName[64];

*TCHAR* tszUserPass[64];

*TCHAR* tszEMailAddr[64];

*TCHAR* tszLoginTime[64];

*TCHAR* tszCreateTime[64];

\_\_int64x nPhoneNumber;

\_\_int64x nIDNumber;

int nUserLevel;

*int nUserState*;

}XENGINE\_PROTOCOL\_USERINFO, \*LPXENGINE\_PROTOCOL\_USERINFO;

### 3.5.2 Reply

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_AUTH

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_AUTH\_REPGETPASS

unPacketSize = sizeof(XENGINE\_PROTOCOL\_USERAUTH)

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:

typedef struct tag\_XEngine\_Protocol\_Auth

{

*CHAR* tszUserName[64];

*CHAR* tszUserPass[64];

ENUM\_PROTOCOLCLIENT\_TYPE enClientType;

ENUM\_PROTOCOLDEVICE\_TYPE enDeviceType;

}XENGINE\_PROTOCOL\_USERAUTH, \*LPXENGINE\_PROTOCOL\_USERAUTH;

#### 2.5.2.1 Error Code

* 291:User not exist
* 292:ver infomation is not success

## 3.6 Get Time

Get Time mean is get user left time,login is requested

### 3.6.1 Request

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_AUTH

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_AUTH\_REQGETTIME

unPacketSize = 0

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

### 3.6.2 Reply

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_AUTH

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_AUTH\_REPGETTIME

unPacketSize = sizeof(AUTHREG\_PROTOCOL\_TIME)

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:

typedef struct

{

CHAR tszUserAddr[XENGINE\_AUTHREG\_SERVICE\_SQL\_MAX\_USERNAME];

CHAR tszUserName[XENGINE\_AUTHREG\_SERVICE\_SQL\_MAX\_USERNAME];

CHAR tszLeftTime[64];

\_\_int64x nTimeLeft;

\_\_int64x nTimeONLine;

ENUM\_AUTHREG\_GENERATESERIALTYPE enSerialType;

}AUTHREG\_PROTOCOL\_TIME, \* LPAUTHREG\_PROTOCOL\_TIME;

## 3.7 Network Verification

Temporary Network Authentication Protocol,Can be used without registering and logging in,Register through a unique identification code (CPUID, motherboard ID, hard disk ID).This registration will be sent to the server, the server will automatically save the record, and identify whether it has expired, and it is convenient for users to use.

This protocol is more secure than local authentication. It can be convenient for some users who don't want to register..

### 3.7.1 Request

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_AUTH

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_AUTH\_REQTRYVER

unPacketSize = IDLEN

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:填写唯一标识码,可以是CPUID等

### 3.7.2 Reply

Protocol Header:Success will return the following content, indicating that the time has not expired

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_AUTH

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_AUTH\_REPTRYVER

unPacketSize = 0

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

#### 3.7.2.1 Error Code

* 0x2D1:Try to timeout
* 0x2D2:Server not enable
* 0x2D3:Server internal occur

## 3.8 Get User

Need to verification user and pass

### 3.8.1 Request

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_AUTH

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_AUTH\_REQGETUSER

unPacketSize = sizeof(XENGINE\_PROTOCOL\_USERAUTH)

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:

typedef struct tag\_XEngine\_Protocol\_Auth

{

*CHAR* tszUserName[64];

*CHAR* tszUserPass[64];

ENUM\_PROTOCOLCLIENT\_TYPE enClientType;

ENUM\_PROTOCOLDEVICE\_TYPE enDeviceType;

}XENGINE\_PROTOCOL\_USERAUTH, \*LPXENGINE\_PROTOCOL\_USERAUTH;

### 3.8.2 Reply

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_AUTH

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_AUTH\_REPGETUSER

unPacketSize = sizeof(XENGINE\_PROTOCOL\_USERINFO)

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Data(XENGINE\_PROTOCOL\_USERINFO)

## 3.9 Set User

Need to verification some basic information

### 3.9.1 Request

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_AUTH

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_AUTH\_REQSETUSER

unPacketSize = sizeof(XENGINE\_PROTOCOL\_USERINFO)

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:

Data(XENGINE\_PROTOCOL\_USERINFO)

### 3.9.2 Reply

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_AUTH

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_AUTH\_REPSETUSER

unPacketSize = 0

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

## 3.10 Notification Protocol

This protocol is only used to send text content. It is used to notify users of messages.

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_AUTH

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_AUTH\_NOTIFYMSG

unPacketSize = MSGLEN

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:The body of the agreement is the content to be sent.

## 3.11 Notification Timeout

This protocol is a notification protocol, and the server actively issues it. When the client receives this protocol, it needs to actively disconnect, otherwise the server will continue to issue this notification.

This agreement indicates that the client's time has expired and there is no time left.

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_AUTH

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_AUTH\_TIMEDOUT

unPacketSize = 0

byVersion = 0

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

# 四 Websocket Interface Protocol

WEBSOCKET Protocol referred to as ws protocol.

In addition to the above TCP protocol, the WS protocol can support HTML or web processing authorization verification, so in addition to APP programs, we can also support authorization verification for websites

The authorization verification of the web is realized by WEBSOCKET protocol load JSON.

## 4.1 user info protocol

Use to delete and register and get pass and user set

### 4.1.1 Request

{  
    **"unOperatorType"**:**2**,  
    **"unOperatorCode"**:**8193**,  
    **"byIsReply"**:**1**,

**"wCrypto"**:**0**,  
    **"st\_UserInfo"**:{  
        **"tszUserName"**:**"123123aa"**,  
        **"tszUserPass"**:**"123123"**,  
        **"tszEMailAddr"**:**"486179@qq.com"**,  
        **"nPhoneNumber"**:**1369999999**,  
        **"nIDNumber"**:**5000000000**  
    }  
}

### 4.1.2 Reply

#### 4.1.2.1 user delete,register set

{  
    **"wHeader"**:**17**,  
    **"wTail"**:**255**,  
    **"unOperatorType"**:**2**,  
    **"unOperatorCode"**:**8197**,  
    **"byIsReply"**:**0**,  
    **"wCrypto"**:**0**,  
    **"wReserve"**:**0**  
}

#### 4.1.2.2 get password

{  
    **"wHeader"**:**17**,  
    **"wTail"**:**255**,  
    **"unOperatorType"**:**2**,  
    **"unOperatorCode"**:**8197**,  
    **"byIsReply"**:**0**,  
    **"wCrypto"**:**0**,  
    **"wReserve"**:**0**,  
    **"st\_UserAuth"**:{  
        **"tszUserName"**:**"123123aa"**,  
        **"tszUserPass"**:**"123123"**,  
        **"enClientType"**:**10**,  
        **"enDeviceType"**:**25**  
    }  
}

## 4.2 user authorize

Use to login,get user protocol

### 4.2.1 Request

{  
    **"unOperatorType"**:**2**,  
    **"unOperatorCode"**:**8197**,  
    **"byIsReply"**:**1**,

**"wCrypto"**:**0**,  
    **"st\_UserAuth"**:{  
        **"tszUserName"**:**"123123aa"**,  
        **"tszUserPass"**:**"123123"**,  
        **"enClientType"**:**10**,  
        **"enDeviceType"**:**25**  
    }  
}

### 4.2.2 Reply

#### 4.2.2.1 Login

Reference 4.1.2.1

#### 4.2.2.2 get user

{  
    **"unOperatorType"**:**2**,  
    **"unOperatorCode"**:**8203**,  
    **"byIsReply"**:**1**,  
    **"wCrypto"**:**0**,  
    **"st\_UserInfo"**:{  
        **"tszUserName"**:**"user"**,  
        **"tszUserPass"**:**"pass"**,  
        **"tszCreateTime"**:**"register time"**,  
        **"tszEMailAddr"**:**"email"**,  
        **"tszLoginTime"**:**"last login time"**,  
        **"nIDNumber"**:**5151515151**,  
        **"nPhoneNumber"**:**13131313**,  
        **"nUserLevel"**:**1**,  
        **"nUserState"**:**0**  
    }  
}

## 4.3 user pay protocol

### 4.3.1 Request

{  
    **"unOperatorType"**:**2**,  
    **"unOperatorCode"**:**8199**,  
    **"byIsReply"**:**1**,

**"wCrypto"**:**0**,  
    **"st\_UserPay"**:{  
        **"tszUserName"**:**"123123aa"**,  
        **"tszSerialNumber"**:**"123123"**  
    }  
}

### 4.3.2 Reply

Reference 4.1.2.1

## 4.4 user time protocol

Use to get user time

### 4.4.1 Request

{  
    **"unOperatorType"**:**2**,  
    **"unOperatorCode"**:**8203**,  
    **"byIsReply"**:**1,**

**"wCrypto"**:**0**  
}

### 4.4.2 Reply

{  
    **"wHeader"**:**17**,  
    **"wTail"**:**255**,  
    **"unOperatorType"**:**2**,  
    **"unOperatorCode"**:**8204**,  
    **"byIsReply"**:**0**,  
    **"wCrypto"**:**0**,  
    **"wReserve"**:**0**,  
    **"st\_UserTime"**:{  
        **"tszUserName"**:**"123123aa"**,  
        **"tszUserAddr"**:**"123123"**,  
        **"tszLeftTime"**:**"2021-12-22 10:11:23"**,  
        **"nTimeLeft"**:**25**,  
        **"nTimeONLine"**:**10**,  
        **"enDeviceType"**:**1**,  
        **"enSerialType"**:**1**  
    }  
}

## 4.5 fast authorize

Reference 3.6

### 4.5.1 Request

{  
    **"unOperatorType"**:**2**,  
    **"unOperatorCode"**:**8203**,  
    **"byIsReply"**:**1**,  
    **"wCrypto"**:**0**,  
    **"lpszPayload"**:**"hard code or other"**  
}

### 4.5.2 Reply

Reference 4.1.2.1

## 4.6 note protocol

Reference 3.10

### 4.6.1 Request

{  
    **"unOperatorType"**:**2**,  
    **"unOperatorCode"**:**12272**,  
    **"byIsReply"**:**1**,  
    **"wCrypto"**:**0**,  
    **"lpszPayload"**:**"send message"**  
}

### 4.6.2 Reply

无

## 4.7 timeout protocol

Reference 3.11

### 4.7.1 Request

{  
    **"unOperatorType"**:**2**,  
    **"unOperatorCode"**:**12287**,  
    **"byIsReply"**:**0**,  
    **"wCrypto"**:**0**  
}

### 4.7.2 Reply

无

# 五 Configure Description

Configure File:XEngine\_Config.ini

## 5.1 Basic Configure

Configure Information:ServiceConfig

* TCPPort:TCP listen port
* WSPort:websocket port
* ThreadPool:Thread Number.
* UserVerTimed:Allow Ver Timeout
* AutoStart:is auto start?
* TimeNotify:timeout notify set,1 means always notify until the client actively closes

## 5.2 Verication Configure

Configure Information:Verification,Quick verification does not require login, and you can get verification authorization information by directly requesting after connecting.

* VerTime:Quick Verification Time
* VerMode:Quick Verification Mode
* TryMode:Try Mode
* TryTime:Try Time

## 5.3 Encrypto Configure

Configure Information:Crypto

* Enable:whether to enable
* Pass:password,just number

## 5.4 EMail Configure

Configure Information:SmtpConfig

* SmtpService:email service address
* SmtpFromAddr:reply address
* SmtpUser:user
* SmtpPass:password

## 5.5 Database Configure

Configure Information:XSql

* tszSQLite:Database file address

## 5.6 Log Configure

Configure Information:XLog just only use to version without interface

* nMaxSize:max log file size
* nMaxCount:file back number
* nLogLeave:log level
* tszLogFile:log save address

# appendix

## Appendix 1 Type Define

reference file:XEngine\_CommHdr.h

## Appendix 2 Protocol Define

reference file:XEngine\_ProtocolHdr.h

## Appendix 3 Transformation Definition

reference file:XEngine\_Types.h only LINUX