XEngine Authorize Service Docment

目录

[XEngine Authorize Service Docment 1](#_Toc31409)

[Preface 5](#_Toc12311)

[Reader 5](#_Toc5954)

[Overview 5](#_Toc4574)

[Associate Module 5](#_Toc32547)

[一 Technical structure 5](#_Toc6069)

[1.1. Directory Structure 5](#_Toc1484)

[1.2 Program Directory 5](#_Toc18772)

[1.3 User Leave 6](#_Toc30147)

[1.4 Detailed Protocol 6](#_Toc27151)

[1.5 Encrypted Communication 6](#_Toc17490)

[二 Configure Env 6](#_Toc25668)

[2.1 WINDOWS 6](#_Toc15280)

[2.1.1 Configure Environment 6](#_Toc1365)

[2.1.2 complie and run 7](#_Toc6037)

[2.2 LINUX 7](#_Toc13944)

[2.2.1 Evnironment Configure 7](#_Toc12119)

[2.2.2 Complie and Run 7](#_Toc213)

[2.3 MacOS 7](#_Toc25084)

[2.4 Old Version to update 7](#_Toc24527)

[三 TCP Interface Protocol 8](#_Toc23537)

[3.1 Login Protocol 8](#_Toc14959)

[3.1.1 Request 8](#_Toc19255)

[3.1.2 Reply 9](#_Toc25928)

[3.2 Notification Timeout 9](#_Toc32332)

[四 Websocket Interface Protocol 10](#_Toc22601)

[4.1 user authorize 10](#_Toc2850)

[4.2.1 Request 10](#_Toc24668)

[4.2.2 Reply 10](#_Toc18127)

[4.2 timeout protocol 11](#_Toc4804)

[五 HTTP Management Interface 11](#_Toc23532)

[5.1 Client Management Interface 11](#_Toc14481)

[5.1.1 Get User 11](#_Toc31929)

[5.1.2 User List 12](#_Toc23214)

[5.1.3 Close Client 13](#_Toc22392)

[5.1.4 Modify User 14](#_Toc6988)

[5.1.5 Delete User 14](#_Toc26281)

[5.2 Serial Management API 15](#_Toc12217)

[5.2.1 List Serial 15](#_Toc17411)

[5.2.2 Insert Serial 16](#_Toc9702)

[5.2.3 Delete Serial 17](#_Toc7267)

[5.3 User Interface 17](#_Toc15147)

[5.3.1 User Delete 17](#_Toc5128)

[5.3.2 User Register 18](#_Toc30211)

[5.3.3 User Pay 19](#_Toc9681)

[5.3.4 Get Back Password 19](#_Toc728)

[5.3.5 Fast Verification 20](#_Toc7520)

[5.4 Token Protocol 21](#_Toc5151)

[5.4.1 Login 21](#_Toc2888)

[5.4.2 UPDate 21](#_Toc18509)

[5.4.3 Close 22](#_Toc19374)

[5.5 Function Switch 22](#_Toc27754)

[5.5.1 Get Functions List 22](#_Toc15117)

[5.5.2 Set Functions Switch 23](#_Toc26703)

[5.6 Banned Protocol 24](#_Toc20986)

[5.6.1 Insert 24](#_Toc18902)

[5.6.2 Delete 25](#_Toc26648)

[5.6.3 Query 25](#_Toc20100)

[5.6.4 Modify 26](#_Toc8092)

[5.7 CDKey System 27](#_Toc1902)

[5.7.1 Create CDKey 27](#_Toc17943)

[5.7.2 Register CDKey 29](#_Toc20847)

[5.7.3 Verification CDKey 29](#_Toc7260)

[5.8 Annoucenement Protocol 29](#_Toc27625)

[5.8.1 Insert 30](#_Toc18102)

[5.8.2 Delete 30](#_Toc2722)

[5.8.3 List 31](#_Toc28631)

[5.9 Get Task Protocool 32](#_Toc25471)

[5.9.1 Get Left Time 32](#_Toc15390)

[5.9.2 Dynamic Verification Code 32](#_Toc32558)

[5.9.3 Get Server Notice 33](#_Toc22565)

[5.10 Temporary Verification 34](#_Toc3067)

[5.10.1 List 34](#_Toc1211)

[5.10.2 List 35](#_Toc11359)

[5.10.3 Modify 35](#_Toc21432)

[六 Third-Verification 36](#_Toc18194)

[6.1 User Login 36](#_Toc9283)

[6.1.1 Request 36](#_Toc19622)

[6.1.2 Response 36](#_Toc8839)

[6.2 User Logout 36](#_Toc18163)

[6.2.1 Request 36](#_Toc28148)

[6.2.2 Response 37](#_Toc26839)

[6.3 User Timeout 37](#_Toc25196)

[6.3.1 Request 37](#_Toc18719)

[6.3.2 Response 37](#_Toc9245)

[七 Advanced Features 37](#_Toc2452)

[7.1 Distributed Authorize 37](#_Toc19147)

[7.2 Multi Login 37](#_Toc29563)

[7.3 Serial Number 38](#_Toc12511)

[7.3.1 Second Type 38](#_Toc7433)

[7.3.2 Time Type 38](#_Toc19317)

[7.3.3 Day Type 38](#_Toc1955)

[7.3.4 Custom Date Type 39](#_Toc23605)

[7.4 Client SDK 39](#_Toc25819)

[7.5 CDKey Location Verification 39](#_Toc4812)

[7.5.1 Thought Network 39](#_Toc24706)

[7.5.2 Thought Location 40](#_Toc9010)

[7.6 Database 40](#_Toc90)

[7.6.1 SQLITE 40](#_Toc16167)

[7.6.2 MYSQL 40](#_Toc28192)

[八 Configure Description 40](#_Toc5258)

[8.1 Basic Configure File 40](#_Toc26366)

[8.1.1 Basic Configure 41](#_Toc16248)

[8.1.2 MAX Configure 41](#_Toc9146)

[8.1.3 Verication Configure 41](#_Toc22100)

[8.1.4 Login Configure 41](#_Toc26385)

[8.1.5 Encrypto Configure 42](#_Toc10028)

[8.1.6 Database Configure 42](#_Toc3096)

[8.1.7 Log Configure 42](#_Toc14606)

[8.1.8 Report Configure 42](#_Toc31913)

[8.2 Functions Switch Configure 42](#_Toc31058)

[appendix 43](#_Toc30747)

[Appendix 1 Type Define 43](#_Toc31714)

[Appendix 2 Protocol Define 43](#_Toc5337)

[Appendix 3 Transformation Definition 43](#_Toc12032)

|  |  |  |  |
| --- | --- | --- | --- |
| File Status：  [ ] Draft  [√] Release | File Name： | XEngine Authorize Service Docment | |
| Be A Version： | V3.15.0.1001 | |
| Released： | 2024-11-21 | |
| 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,WEBSOCKET,HTTP protocol though basic protoco

Should be bind three port when Start service

## Directory Structure

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

## 1.2 Program Directory

We provide server and management tools

* XEngine\_AuthorizeApp:WINDOWS Management Tools program.depend XEngine\_AuthorizeService
* XEngine\_AuthorizeService:Network Authorize Mainly Server

## 1.3 User Leave

There are many levels for users, which need to be distinguished by themselves. Currently, except for level 0, which is not verified, all others will be processed. In the future, all levels may be used and require identification.

-1 means ban.

0 is the highest level

> 0 && < 10 are all administrative privileges

> =10 \*\* <= 20 are all VIP member users. Pay users should be in this group

>=20 belongs to ordinary users

## 1.4 Detailed Protocol

The TCP private protocol adopts the protocol defined by XEngine. You can refer to the protocol document of XEngine to learn more about the definition and interpretation of the protocol

## 1.5 Encrypted Communication

Encrypted communication is achieved through XCrypto technology. The encryption only applies to the payload content and does not encrypt the headers. This means that the headers of the TCP XENGINE protocol, WEBSOCKET communication, and HTTP remain unencrypted. Only the payload content is encrypted.

# 二 Configure Env

## 2.1 WINDOWS

Need to download XEngine.

Complie and run and debug by vs.

### 2.1.1 Configure Environment

Follow the instructions in the XEngine Readme file to execute the script to configure the environment.

### 2.1.2 complie and run

When you complete with configuration.you can come in code path.open XEngine.sln by vs.

If environment not have error.complie is succesed.contrain 7 module and 2 exe program

And you need copy XEngine depend Module to your complied dir.

Note: Xengine environment can be copied to your compilation directory through vscopy script, provided that you configure your xengine environment

## 2.2 LINUX

### 2.2.1 Evnironment Configure

If you use linux.you must running on ubuntu(22.04) or RockyLinux(9.x)...

### 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.

## 2.3 MacOS

MacOS requires 13 or above versions, and the compilation and operation mode can refer to linux

## 2.4 Old Version to update

3.3Version up to subsequent version need to create and modify database tables

Create Table:Authorize\_BannedAddr and Authorize\_BannedAddr.refer:XEngine\_SQL/main.sql

Modify Table:AuthReg\_NetVer -> Authorize\_NetVer

AuthReg\_Serial -> Authorize\_Serial

AuthReg\_User -> Authorize\_User

Version 3.10 upgrade: The temporary trial table needs to be modified to a new table name, and the fields are also modified. Please refer to the SQL file

# 三 TCP Interface Protocol

TCP interface protocol is used for application verification such as APP. It can be used for login.Network authorization verification can be performed through this interface. For other management interfaces, please refer to the HTTP interface.

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 XCrypto algorithm. If encryption is required, please fill, otherwise please fill in 0

## 3.1 Login Protocol

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

If TOKEN is 0, the return will be filled in automatically, or you can use the TOKEN of dynamic verification code to fill in, then the system will not regenerate (unless there is an error in the TOKEN)

### 3.1.1 Request

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = can be use dynamic-code token

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

{

XCHAR tszUserName[64]; //user

XCHAR tszUserPass[64]; //pass

XCHAR tszDCode[8]; //dynamic

ENUM\_PROTOCOLCLIENT\_TYPE enClientType; //can be 0

ENUM\_PROTOCOLDEVICE\_TYPE enDeviceType; //must set

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

### 3.1.2 Reply

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = token

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
* 250:enDeviceType does not set
* 251:user not found
* 252:password is error
* 253:loginned
* 254:User does not have permission
* 255:user timeout
* 256:Server internal error

## 3.2 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.

The unOperatorType in JSON refers to the unOperatorType of TCP, and the unOperatorCode refers to the unOperatorCode of TCP, which is filled in decimal

## 4.1 user authorize

Use to login

### 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

{  
    **"wHeader"**:**17**,  
    **"wTail"**:**255**,  
    **"unOperatorType"**:**2**,  
    **"unOperatorCode"**:**8198**,  
    **"wCrypto"**:**0**,  
    **"wReserve"**:**0**  
}

## 4.2 timeout protocol

The code value is :XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_AUTH\_TIMEDOUT

# 五 HTTP Management Interface

This interface is used for HTTP RESTFUL, and the server can be managed through this interface in the future

The HTTP interface will only return 200 success, the error code needs to be obtained through the json of the payload. code and msg

## 5.1 Client Management Interface

management requires Token verification. You need to log in to get the Token value and attach it to the Root of Json before it can be used

To get the value of Token, refer to the protocol in Section 5.4

{

"xhToken":100011213

}

### 5.1.1 Get User

#### 5.1.1.1 Request

Method:POST

API:http://IP:PORT/auth/client/get

Payload:

{

"xhToken":100011213,

"st\_UserInfo":{

"tszUserName":"用户名"

}

}

#### 5.1.1.2 Reply

{

"code":0,

"msg":"success",

"st\_UserTable":{

"tszLeftTime":"还剩下多少时间",

"tszHardCode":"硬件吗",

"enSerialType":1,

"enDeviceType":0,

"st\_UserInfo":{

"tszUserName":"用户名",

"tszUserPass":"密码",

"tszEMailAddr":"电子邮件",

"tszLoginTime":"登录时间",

"tszCreateTime":"注册时间",

"nPhoneNumber":1369999999,

"nIDNumber":5111010101011,

"nUserLevel":5,

"nUserState":0

}

}

}

### 5.1.2 User List

#### 5.1.2.1 Request

Method:POST

API:http://IP:PORT/auth/client/list

Payload:

{

"xhToken":100011213,

"Online":false,

"PosStart":0,

"PosEnd":100

}

#### 5.1.2.2 Reply

{

"Array":[

{

"enDeviceType":0,

"enSerialType":1,

"st\_UserInfo":{

"nIDNumber":511025111111111100,

"nPhoneNumber":1366666666,

"nUserLevel":5,

"nUserState":0,

"tszCreateTime":"2022-05-27 10:09:17",

"tszEMailAddr":"486179@qq.com",

"tszLoginTime":"",

"tszUserName":"123123aa",

"tszUserPass":"123123"

},

"tszHardCode":"ND2N09DN1N11",

"tszLeftTime":"81"

}

],

"code":0,

"msg":"success"

}

### 5.1.3 Close Client

#### 5.1.3.1 Request

Method:POST

API:http://IP:PORT/auth/client/close

Payload:

{

"xhToken":100011213,

"st\_UserInfo":{

"tszUserName":"user"

}

}

#### 5.1.3.2 Reply

{

"code":0,

"msg":"success"

}

### 5.1.4 Modify User

#### 5.1.4.1 Request

Method:POST

API:http://IP:PORT/auth/client/modify

Payload:

{

"xhToken":100011213,

"st\_UserTable":{

"enSerialType":1,

"st\_UserInfo":{

"nIDNumber":511025111111111100,

"nPhoneNumber":1366666666,

"nUserLevel":5,

"tszCreateTime":"2022-05-27 10:09:17",

"tszEMailAddr":"486179@qq.com",

"tszUserName":"123123aa",

"tszUserPass":"123123"

},

"tszHardCode":"ND2N09DN1N11",

"tszLeftTime":"81"

}

}

#### 5.1.4.2 Reply

{

"code":0,

"msg":"success"

}

### 5.1.5 Delete User

#### 5.1.5.1 Request

Method:POST

API:http://IP:PORT/auth/client/delete

Payload:

{

"xhToken":100011213,

"st\_UserInfo":{

"tszUserName":"123123aa"

}

}

#### 5.1.5.2 Reply

{

"code":0,

"msg":"success"

}

## 5.2 Serial Management API

Requires Token Value,refer 5.1

### 5.2.1 List Serial

#### 5.2.1.1 Request

Method:POST

API:http://IP:PORT/auth/serial/list

Payload:

{

"xhToken":100011213,

"PosStart":0,

"PosEnd":100

}

#### 5.2.1.2 Reply

{

"Array":[

{

"bIsUsed":0,

"enSerialType":2,

"tszCreateTime":"2022-08-11 14:45:58",

"tszMaxTime":"2",

"tszSerialNumber":"XAUTH-XYRYD-MRLJ2-YYF85-UQLKI-31115-98XVA-LQYKX-12744-KFQ38",

"tszUserName":"NOT"

}

],

"code":0,

"msg":"success"

}

### 5.2.2 Insert Serial

#### 5.2.2.1 Request

Method:POST

API:http://IP:PORT/auth/serial/list

Payload:

{

"xhToken":100011213,

"st\_SerialInfo":{

"enSerialType":1,

"nNumberCount":5,

"nSerialCount":2,

"tszHasTime":"20"

}

}

enSerialType:serial type

nNumberCount:serial field number

nSerialCount:serial number

tszHasTime:have time

#### 5.2.2.2 Reply

{

"code":0,

"msg":"success"

}

### 5.2.3 Delete Serial

#### 5.2.3.1 Request

Method:POST

API:http://IP:PORT/auth/serial/delete

Payload:

{

"xhToken":100011213,

"Array":[

{

"tszSerialNumber":"XAUTH-XYRYD-MRLJ2-YYF85-UQLKI-31115-98XVA-LQYKX-12744-KFQ38"

}

]

}

#### 5.2.3.2 Reply

{

"code":0,

"msg":"success"

}

## 5.3 User Interface

User api can be used without login.user interface does not take token

### 5.3.1 User Delete

#### 5.3.1.1 Request

Method:POST

API:http://IP:PORT/auth/user/delete

Payload:

{

"st\_UserInfo" :

{

"nIDNumber" : 511025111111111111,

"nPhoneNumber" : 1366666666,

"tszEMailAddr" : "486179@qq.com",

"tszUserName" : "123123aa",

"tszUserPass" : "123123"

}

}

#### 5.3.1.2 Reply

{

"code" : 0,

"msg" : "success"

}

### 5.3.2 User Register

#### 5.3.2.1 Request

Method:POST

API:http://IP:PORT/auth/user/register

Payload:

{

"st\_UserTable" :

{

"st\_UserInfo" :

{

"nIDNumber" : 511025111111111111,

"nPhoneNumber" : 1366666666,

"tszEMailAddr" : "486179@qq.com",

"tszUserName" : "123123aa",

"tszUserPass" : "123123",

"nUserLevel":5

},

"tszHardCode" : "2FDWAD02JD2091"

}

}

#### 5.3.2.2 Reply

{

"code":0,

"msg":"success"

}

### 5.3.3 User Pay

#### 5.3.3.1 Request

Method:POST

API:http://IP:PORT/auth/user/pay

Payload:

{

"st\_UserPay" :

{

"tszSerialNumber" : "XAUTH-XYRYM-ZAA10-59NKW-KOFLP-35838-ZXC36-ZOVTH",

"tszUserName" : "123123aa"

}

}

#### 5.3.3.2 Reply

{

"code":0,

"msg":"success"

}

### 5.3.4 Get Back Password

If password encryption is turned on, the password retrieval function represents password reset, and st\_UserInfo must bring the password field of tszUserPass. Otherwise, it represents password retrieval.

#### 5.3.4.1 Request

Method:POST

API:http://IP:PORT/auth/user/pass

Payload:

{

"st\_UserInfo" :

{

"nIDNumber" : 511025111111111111,

"nPhoneNumber" : 1366666666,

"tszEMailAddr" : "486179@qq.com",

"tszUserName" : "123123aa"

}

}

#### 5.3.4.2 Reply

{

"code":0,

"msg":"success",

"st\_UserAuth":{

"enClientType":0,

"enDeviceType":0,

"tszUserName":"123123aa",

"tszUserPass":"123123"

}

}

### 5.3.5 Fast 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

#### 5.3.5.1 Request

Method:POST

API:http://IP:PORT/auth/user/try

Payload:

{

"st\_VERTemp":{

"tszVSerial":"cpuid:112233"

}

}

#### 5.3.5.2 Reply

{

"code":0,

"msg":"success"

}

## 5.4 Token Protocol

Used to manage API permission verification management

### 5.4.1 Login

Used to get the Token value, this login support management and authorize verifcation login

#### 5.4.1.1 Request

Method:GET

API:http://IP:PORT/api?function=login&user=123123aa&pass=123123&device=36

Parament:User,password,device type,refer:ENUM\_PROTOCOLDEVICE\_TYPE

Payload:none

#### 5.4.1.2 Reply

{

"code":0,

"msg":"success",

"nTimeout":3600,

"tszTimeEnd":"2022-08-25 15:19:12",

"tszTimeStart":"2022-08-25 14:19:12",

"xhToken":1000030115

}

Token has an expiration time. After expiration, you cannot use the management AP

### 5.4.2 UPDate

If you want to continue to use this Token, you can use the renewal agreement to update,provided that token exist.

#### 5.4.2.1 Request

Method:GET

API:http://IP:PORT/api?function=update&token=1000030115

Payload:none

#### 5.4.2.2 Reply

{

"code" : 0,

"msg" : "success",

"nTimeout" : 3600,

"tszTimeEnd" : "2022-08-25 15:31:42",

"tszTimeStart" : "2022-08-25 14:31:42",

"xhToken" : 1000032558

}

### 5.4.3 Close

If you does not use this token,you can close or wait auto destory when timeout.

#### 5.4.3.1 Request

Method:GET

API:http://IP:PORT/api?function=close&token=1000030115

Payload:None

#### 5.4.3.2 Reply

{

"code" : 0,

"msg" : "success"

}

## 5.5 Function Switch

The function switch interface is only a temporary function and will not work after reboot

### 5.5.1 Get Functions List

#### 5.5.1.1 Request

Method:POST

Interface:http://IP:PORT/auth/switch/get

Payload:

{

"xhToken":12345679

}

#### 5.5.1.2 Reply

{

"msg":"success",

"code":0,

"bSwitchDelete":true,

"bSwitchRegister":true,

"bSwitchLogin":true,

"bSwitchPass":true,

"bSwitchPay":true,

"bSwitchTime":true,

"bSwitchCDKey":true,

"bSwitchNotice":true,

"bSwitchDCode":false,

"bSwitchMulti":false,

"bSwitchTry":true,

"bSwitchBanned":true

}

### 5.5.2 Set Functions Switch

#### 5.5.2.1 Request

Method:POST

Interface:http://IP:PORT/auth/switch/set

Payload:

{

"xhToken":123456789,

"st\_SwitchInfo":{

"bSwitchDelete":true,

"bSwitchRegister":true,

"bSwitchLogin":true,

"bSwitchPass":true,

"bSwitchPay":true,

"bSwitchTime":true,

"bSwitchCDKey":true,

"bSwitchNotice":true,

"bSwitchDCode":false,

"bSwitchMulti":false,

"bSwitchTry":true,

"bSwitchBanned":true

}

}

#### 5.5.2.2 Reply

{

"msg":"success",

"code":0

}

## 5.6 Banned Protocol

Can be ban ip and user login or connect

### 5.6.1 Insert

#### 5.6.1.1 Request

Method:POST

API:http://IP:PORT/auth/banned/insert

Payload:

{

"xhToken":123456789,

"st\_Banned":{

"bEnable":true,

"tszIPAddr":"ban ip address",

"tszUserName":"ban user name",

"tszLeftTime":"left time"

}

}

#### 5.6.1.2 Reply

{

"msg":"success",

"code":0

}

### 5.6.2 Delete

#### 5.6.2.1 Request

Method:POST

API:http://IP:PORT/auth/banned/delete

Payload:

{

"xhToken":123456789,

"st\_Banned":{

"tszIPAddr":"ban ip address",

"tszUserName":"ban user name"

}

}

#### 5.6.2.2 Reply

{

"msg":"success",

"code":0

}

### 5.6.3 Query

#### 5.6.3.1 Request

Method:POST

API:http://IP:PORT/auth/banned/list

Payload:

{

"xhToken":123456789

}

#### 5.6.2.2 Reply

{

"ArrayAddr":[

{

"nID":1,

"bEnable":true,

"tszIPAddr":"192.168.1.1",

"tszLeftTime":"2023-12-29 15:00:20",

"tszCreateTime":"2022-12-29 15:00:20"

}

],

"ArrayUser":[

{

"nID":1,

"bEnable":true,

"tszLeftTime":"2023-12-29 15:00:20",

"tszCreateTime":"2022-12-29 14:57:03",

"tszUserName":"123123"

},

{

"nID":2022,

"bEnable":true,

"tszLeftTime":"2023-12-29 15:00:20",

"tszCreateTime":"adadad",

"tszUserName":"2"

}

],

"CountAddr":1,

"CountUser":2,

"code":0,

"msg":"success"

}

### 5.6.4 Modify

#### 5.6.4.1 Request

Method:POST

API:http://IP:PORT/auth/banned/modify

Payload:

{

"xhToken":123456789,

"st\_Banned":{

"bEnable":true,

"tszLeftTime":"2023-12-29 15:00:20",

"tszCreateTime":"adadad",

"tszUserName":"2"

}

}

#### 5.6.4.2 Reply

{

"msg":"success",

"code":0

}

## 5.7 CDKey System

### 5.7.1 Create CDKey

#### 5.7.1.1 Request

Method:POST

Parament:无

Url:http://127.0.0.1:5302/auth/cdkey/create

Payload:

{

"tszAddr": "服务器地址",

"nPort": 5000,

"st\_AuthAppInfo": {

"tszAppName": "自定义名称",

"tszAppVer": "自定义版本",

"nExecTime": 0,

"bInit": false

},

"st\_AuthRegInfo": {

"tszHardware": "HARDWARE CODE",

"tszCreateTime": "CDKEY Create Time",

"tszRegisterTime": "CDKEY Register time",

"tszLeftTime": "CDKEY have time",

"tszStartTime": "CDKEY start time",

"tszExpiryTime": "CDKEY expiry time",

"enSerialType": 1,

"enRegType": 1,

"enHWType": 1,

"enVModeType": 1

},

"st\_JsonSerialInfo": {

"tszTimeSerial": "XAUTH-XXX-XXX-XXX",

"nTimeCount": 10,

"tszDataSerial": "XAUTH-XX-XX-X-X",

"tszDataTime": "2022-01-01 11:11:10",

"bTimeAdd": false,

"tszUNLimitSerial": "XAUTH-X-X-X"

},

"st\_AuthUserInfo": {

"tszUserName": "register user",

"tszUserContact": "contact address",

"tszCustom": "option"

}

}

#### 5.7.1.2 Reply

[Connection]

tszAddr=http://app.xyry.org

nPort=5501

[AppInfo]

tszAppName=XEngine

tszAppVer=1.0.0.1001

nExecTime=0

bInit=0

[AuthReg]

nHasTime=0

enHWType=1

enRegType=2

enSerialType=3

enVModeType=1

tszHardware=5501012NE21N

tszCreateTime=2024-07-15 14:06:20

tszRegisterTime=

tszLeftTime=

tszStartTime=

tszExpiryTime=

[AuthSerial]

tszTimeSerial=XAUTH-CS3GW-KFNQD-06481-O13KP-REPGG-63723

nTimeCount=9999

tszDataSerial=XAUTH-YT2YY-YCQPE-67536-MNMKS-VWSRE-12183

tszDataTime=2025-07-15 14:06:20

bAddTime=0

tszUNLimitSerial=XAUTH-FTSYS-FADPO-16623-CE67A-BOYGP-77135

[AuthUser]

tszUserName=qyt

tszUserContact=486179@qq.com

tszCustom=

### 5.7.2 Register CDKey

#### 5.7.2.1 Request

Method:POST

Parament:NULL

Url:http://127.0.0.1:5302/auth/cdkey/auth

Payload:

5.7.1 reply content

#### 5.7.2.2 Reply

Reply to the content of the successfully authorized CDKey file

### 5.7.3 Verification CDKey

#### 5.7.3.1 Request

Method:POST

Parament:无

Address:http://127.0.0.1:5302/auth/cdkey/ver

Payload:Create CDKEY reply info

#### 5.7.3.2 Reply

Return cdkey context when success,fail reply error msg

## 5.8 Annoucenement Protocol

Notice charset need to utf format

### 5.8.1 Insert

#### 5.8.1.1 Request

Method:POST

Parament:None

Addresss:http://127.0.0.1:5302/auth/notice/insert

Payload:

{

"xhToken":123456789,

"st\_Notice":{

"tszContext":"Notice context"

}

}

#### 5.8.1.2 Reply

{

"msg":"success",

"code":0

}

### 5.8.2 Delete

#### 5.8.2.1 Request

Method:POST

Parament:None

Addresss:http://127.0.0.1:5302/auth/notice/delete

Payload:

{

"xhToken":123456789,

"st\_Notice":{

"nID":1

}

}

#### 5.8.2.2 Reply

{

"msg":"success",

"code":0

}

### 5.8.3 List

#### 5.8.3.1 Request

Method:GET

Parament:无

Addresss:http://127.0.0.1:5302/auth/notice/list

Payload:NONE

#### 5.8.3.2 Reply

{

"Array":[

{

"nID":1,

"tszContext":"Notice 1",

"tszCreateTime":"2023-02-24 11:24:26"

},

{

"nID":2,

"tszContext":"Notice 2",

"tszCreateTime":"2023-02-24 11:27:30"

}

],

"Count":2,

"code":0,

"msg":"success"

}

## 5.9 Get Task Protocool

### 5.9.1 Get Left Time

#### 5.9.1.1 Request

Method:POST

API:http://IP:PORT/api?function=time&token=123123

Payload:NULL

#### 5.9.1.2 Reply

{

"st\_UserTime":{

"enDeviceType":0,

"enSerialType":1,

"nTimeLeft":2,

"nTimeONLine":3,

"tszLeftTime":"2022-08-25 13:56:36",

"tszUserAddr":"127.0.0.1:50168",

"tszUserName":"123123aa"

}

}

### 5.9.2 Dynamic Verification Code

You can enable the function when login.Prevent brute force cracking

You must use token value fill to you login protocol when you Enable the function,otherwise ver will fail.

#### 5.9.2.1 Request

Method:GET

Parament:无

Address:http://127.0.0.1:5302/api?function=dcode

Payload:None

#### 5.9.2.2 Reply

{

"code":0,

"msg":"success",

"nDynamicCode":803798,

"nTimeout":30,

"tszTimeEnd":"2023-04-18 16:56:14",

"tszTimeStart":"2023-04-18 16:55:44",

"xhToken":3411647435

}

### 5.9.3 Get Server Notice

Get announcement information released by the server

#### 5.9.3.1 Request

Method:GET

API:http://IP:PORT/api?function=notice&token=123123

Payload:NULL

#### 5.9.3.2 Reply

{

"Array" :

[

{

"nID" : 1,

"tszContext" : "123123",

"tszCreateTime" : "2024-09-05 17:08:48"

}

],

"Count" : 1,

"code" : 0,

"msg" : "success"

}

## 5.10 Temporary Verification

### 5.10.1 List

#### 5.10.1.1 Request

Method:POST

Parament:NULL

API:http://127.0.0.1:5302/auth/try/list

Payload:

{

"xhToken":123456789,

"PosStart":0,

"PosEnd":100

}

#### 5.10.1.2 Reply

{

"Array" :

[

{

"enVMode" : 2,

"nID" : 2,

"nLTime" : 4,

"nVTime" : 5,

"tszVDate" : "2024-01-09 14:22:42",

"tszVSerial" : "C6DS090081035"

}

],

"Count" : 1,

"code" : 0,

"msg" : "success"

}

### 5.10.2 List

#### 5.10.2.1 Request

Method:POST

Parament:NULL

API:http://127.0.0.1:5302/auth/try/delete

Payload:

{

"xhToken":123456789,

"st\_VERTemp":{

"tszVSerial":"C6DS090081035"

}

}

#### 5.10.2.2 Reply

{

"msg":"success",

"code":0

}

### 5.10.3 Modify

#### 5.10.3.1 Request

Method:POST

Parament:NULL

API:http://127.0.0.1:5302/auth/try/modify

Payload:

{

"xhToken":123456789,

"st\_VERTemp":{

"enVMode":2,

"nID":1,

"nLTime":5,

"nVTime":5,

"tszVDate": "2024-01-09 14:22:42",

"tszVSerial":"C6DS090081035"

}

}

#### 5.10.3.2 Reply

{

"msg":"success",

"code":0

}

# 六 Third-Verification

From version 3.2 after, the authentication service can handle user login, logout, timeout, etc. through authentication interface. If you want to manage the user table yourself, then you need to use this function. If this function is enabled, the server's user database will be no longer working

The Third-Ver uses the HTTP POST method, which will be submitted to the specified URL according to the configuration file

## 6.1 User Login

### 6.1.1 Request

Method:POST

URL:tszPassLogin of Configure

Payload:Response JSon of 5.3.4

### 6.1.2 Response

HTTP 200is success,other is failed.

If sucesss,you need response a user table to our,you can refer 5.1.4 JSON,must fill all field

## 6.2 User Logout

### 6.2.1 Request

Method:POST

URL:tszPassLogout of Configure

Payload:Response JSon of 5.9.1

### 6.2.2 Response

System does not care about response

## 6.3 User Timeout

### 6.3.1 Request

Method:POST

URL:tszPassTimeout of Configure

Payload:Response Json of 5.9.1

### 6.3.2 Response

System does not care about response

# 七 Advanced Features

## 7.1 Distributed Authorize

Distributed authentication is enabled by default, if users want distributed authentication, they can submit to our server through the auth/pass/interface of post, and the server will pass the authentication and return to the authentication side.

Server can process login, logout, expiration

You can also use the 6 sections as a validation forwarding service. The new version provides more flexible authentication methods

## 7.2 Multi Login

You can turn on the multi-login switch in the Switch configuration file.

Multi login now supports all register cards. The multi login device type must be filled in correctly. He confirms the multi-terminal login type according to the type of device logged in.

The minutes of multi-terminal logins can be combined together, and the number of times will be reduced by one each time you log in.

Custom and day cards are processed according to the expiration time, and it is not recommended to open them.

## 7.3 Serial Number

On the serial number management page, there are some fields that need to be filled out. Among them, "Serial Number Quantity" represents the number of fields in the generated serial numbers, for example, 4 corresponds to the format xxx-xxx-xx-xxx, while 5 represents xxx-xxx-xxx-xxx-xxx.

The "Generate Card Quantity" refers to how many cards you want to generate. The card ownership period varies depending on the type, please refer to the information below.

Users are not supported to change the recharge card type, and the replacement will directly delete the past time

### 7.3.1 Second Type

SECOND type, second card, represents the number of seconds the user is online. The card ownership time can be directly filled in the value, for example, fill in 60, which means the user can be online for 60 seconds.

After turning on multi-terminal login, the time can be combined. If two users are online, the available time will be doubled. For example, one user can be online for 60 seconds, and two users can only be online for 30 seconds.

### 7.3.2 Time Type

TIME type, times card, represents the number of available times, which will be reduced by 1 after each login. This type of card has nothing to do with online time. In other words, no matter how long you are online after logging in, it will not expire. It will not be verified unless you log in next time.

This card type is also suitable for multi-terminal login, and each login reduces the number of available logins by one.

The card ownership time is the number of times it can be used. You can directly fill in the value, such as 60, which means you can log in 60 times.

### 7.3.3 Day Type

DAY type. The number of days card, according to the day deduction, set the number of days. Log in no matter how many times, only deduct the number of days a day, for example, it has 5 days. There are 4 days left for how many times you log in.

When this card type is not combined, no processing will be done. Once turned on, it will have no effect. It only allows multi-end login. Users are asked to consider whether to turn on this option.

### 7.3.4 Custom Date Type

CUSTOM card type, custom time card, the card ownership time is the expiration date, please fill in the fixed format, such as: yyyy-mm-dd hh:mm:ss, for example, the expiration time is 2023-11-20 10:10:01 .

When this card type is not combined, no processing will be done. Once turned on, it will have no effect. It only allows multi-end login. Users are asked to consider whether to turn on this option.

## 7.4 Client SDK

Client SDK: AuthorizeModule\_Client module, this module encapsulates the login interface and can be used to handle login and timeout judgment.

After connecting to the server through AuthClient\_Connector\_Connect, log in through the AuthClient\_Connector\_Login function. After successful login, you can use the AuthClient\_Connector\_GetAuth function to determine whether the verification has timed out or is offline.

This module is used in the E language example to implement verification. This module needs to be compiled with x86-release before it can be used.

Please note: The password can only be numbers or letters. You need to close it when not in use.

You need to rely on the following modules to use the functions of this module normally, and you can also reduce the dependent modules through compilation options.

XEngine\_BaseLib.dll(\_XAUTH\_BUILD\_SWITCH\_CLIENT\_TCP)(\_XAUTH\_BUILD\_SWITCH\_CLIENT\_HTTP)

XEngine\_Algorithm.dll(\_XAUTH\_BUILD\_SWITCH\_CLIENT\_TCP)

XEngine\_OPenSsl.dll(\_XAUTH\_BUILD\_SWITCH\_CLIENT\_TCP)

XClient\_Socket.dll(\_XAUTH\_BUILD\_SWITCH\_CLIENT\_TCP)

XClient\_APIHelp.dll(\_XAUTH\_BUILD\_SWITCH\_CLIENT\_HTTP)

XEngine\_SystemApi.dll(\_XAUTH\_BUILD\_SWITCH\_CLIENT\_HTTP)

jsoncpp.dll(\_XAUTH\_BUILD\_SWITCH\_CLIENT\_HTTP)

## 7.5 CDKey Location Verification

CDKEY can be verified by network or local. Regardless of network verification or local verification, the CDKEY file should be encrypted. Network verification should pass plain text or cipher text data according to the requirements of the server

### 7.5.1 Thought Network

It can be created and verified through the network. We do not recommend authorizing CDKEY through the network because it is difficult to control or it is not possible at this stage

### 7.5.2 Thought Location

This is our recommended method. The user creates a CDKEY to a local file, and then sends the file to the administrator. After the administrator obtains the file, he opens the CDKEY through the management program for authorization

#### 7.5.2.1 Thought User

Sequence information can be written when creating CDKEY, so that users can register CDKEY by themselves without any need for management intervention.

You can call the Authorize\_CDKey\_UserRegister function to implement user self-registration. Users can write their own set of interface programs to implement it. You can refer to our example for implementation.

## 7.6 Database

We support two databases, one SLQITE and one MYSQL. You must and can only choose one database to use.

### 7.6.1 SQLITE

SQLITE is a local database. Lightweight users can choose this. It is very convenient. We provide a default database file and you can start the server directly without any specific operations

### 7.6.2 MYSQL

MYSQL needs to configure the environment by itself and import the mysql.sql database file in the XEngine\_SQL directory before it can be used

# 八 Configure Description

## 8.1 Basic Configure File

Configure File:XEngine\_Config.json

### 8.1.1 Basic Configure

* nTCPPort:TCP listen port
* nWSPort:websocket port
* nHTTPPort:http port
* bDeamon:whether deamon run
* bTimeNotify:timeout notify set,true means always notify until the client actively closes

### 8.1.2 MAX Configure

Configure Information:XMax

* nMaxClient:Max Client Count
* nMaxQueue:Max Queue Count
* nIOThread:network io count
* nTCPThread:TCP threads count
* nWSThread:WEBSOCKET threads count
* nHTTPThread:HTTP threads count

### 8.1.3 Verication Configure

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

* nUserTimeout:user login timeout
* nTokenTimeout:Token Timeout,second.
* nVerTime:Quick Verification Time
* nVerMode:Quick Verification Mode
* nTryMode:Try Mode
* nTryTime:Try Time
* st\_XCDKey:cdkey configure for example
* st\_PassCrypto:password crypt configure,password to encrypt when enable
* st\_XToken:auto renewal.-1 not timeout,and set auto renewal times

### 8.1.4 Login Configure

Configure Infomation:XLogin

* bHTTPAuth:whether enable to http authorize verifcaiton,need http heartbeat
* bPassAuth:whether enable to third-ver
* nMultiMode:Multi Login Mode,0 verifies according to classification, 1 verifies according to subtype, 2 does not limit the type
* st\_PassUrl:HTTP Third ver interface
* st\_MultiLogin:Available multi-login client payment types

### 8.1.5 Encrypto Configure

Configure Information:Crypto,It is recommended to use encrypted transmission, otherwise it is easy to be cracked

* bEnable:whether to enable
* nPass:password,just number

### 8.1.6 Database Configure

Configure Information:XSql

* nSQLType:0 SQLITE,1 MYSQL
* SQLAddr:MYSQL address
* SQLPort:MYSQL port
* SQLUser:MYSQL username
* SQLPass:MYSQL password
* SQLFile:SQLITE DATABASE FILE ADDRESS

### 8.1.7 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

### 8.1.8 Report Configure

XReport Configure

* bEnable:whether to enable
* tszServiceName:service name
* tszAPIUrl:report address

## 8.2 Functions Switch Configure

Configure File:XEngine\_SwitchConfig.json

* bSwitchDelete:User Delete Switch
* bSwitchRegister:User Register Switch
* bSwitchLogin:User Login Switch
* bSwitchPay:User Pay Switch
* bSwitchPass:User Get Password Swtich
* bSwitchTime:User Get LeftTime Swtich
* bSwitchCDKey:cdkey create and ver and auth switch
* bSwitchNotice:Notice Function Switch
* bSwitchDCode:Dynamic Verification Code Function Switch
* bSwitchMulti:Multi Login Function Switch
* bSwitchTry:temporary verification function switch
* bSwitchBanned:banned function switch

# 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