XEngine Message Queue Docment

目录

[XEngine Storage Service Docment 1](#_Toc8510)

[Preface 4](#_Toc23805)

[Reader 4](#_Toc18956)

[Overview 4](#_Toc31661)

[Associate Module 4](#_Toc12106)

[一 Technical structure 4](#_Toc25124)

[1.1 Upload interface 4](#_Toc10998)

[1.2 Download interface 5](#_Toc3343)

[1.3 Mangement Interface 5](#_Toc20232)

[二 Configure Env 5](#_Toc13952)

[2.1 WINDOWS 5](#_Toc1370)

[2.1.1 Configure Environment 5](#_Toc8375)

[2.1.2 complie and run 6](#_Toc12244)

[2.2 LINUX 6](#_Toc13330)

[2.2.1 Evnironment Configure 6](#_Toc21257)

[2.2.2 Complie and Run 6](#_Toc29123)

[2.3 Version Requirements 7](#_Toc26277)

[2.3.1 System Version 7](#_Toc2303)

[2.3.2 Software Version 7](#_Toc603)

[三 Interface Protocol 7](#_Toc2746)

[3.1 query protocol 8](#_Toc4917)

[3.1.1 file list 8](#_Toc22574)

[3.2 third interface 8](#_Toc25716)

[3.2.1 user auth 8](#_Toc2754)

[3.2.2 upload proxy 9](#_Toc24675)

[3.2.3 complete notify 9](#_Toc18449)

[四 Configure Description 9](#_Toc20678)

[4.1 Serice Configure 9](#_Toc19220)

[4.1.1 basic configure 9](#_Toc25557)

[4.1.2 Max Configure 10](#_Toc29635)

[4.1.3 Time Configure 10](#_Toc8663)

[4.1.4 Log Configure 10](#_Toc30894)

[4.1.5 Database Configure 10](#_Toc8417)

[4.1.6 Storage Configure 10](#_Toc29268)

[4.1.7 Proxy Configure 10](#_Toc28485)

[4.1.8 Limit Configure 11](#_Toc30279)

[4.1.9 Version Configure 11](#_Toc12974)

[4.2 LoadBalance Configure 11](#_Toc26721)

[4.2.1 Basic Configure 11](#_Toc10503)

[4.2.2 Loadbalance Configure 11](#_Toc6792)

[4.2.3 Load Attributes 12](#_Toc24982)

[五 Advanced configuration 12](#_Toc22503)

[5.1 Distributed 12](#_Toc30709)

[5.1.1 Network Distributed 12](#_Toc7801)

[5.1.2 Storage Distributed 13](#_Toc10191)

[appendix 13](#_Toc8792)

[Appendix 1 Type Define 13](#_Toc1931)

[Appendix 2 Protocol Define 13](#_Toc20538)

[Appendix 3 Transformation Definition 13](#_Toc26955)

[Appendix 4 update log 13](#_Toc28911)

|  |  |  |  |
| --- | --- | --- | --- |
| File Status：  [ ] Draft  [√] Release | File Name： | XEngine Message Queue Service Docment | |
| Be A Version： | V1.2 | |
| Released： | 2021-07-16 | |
| Writer： qyt | | |

# Preface

## Reader

Development ,test

## **Overview**

This document contains related technical descriptions and interface definitions for storage services!

## Associate Module

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

And This service used library for jsoncpp

# 一 Technical structure

Used to http and tcp protocol though basic protocol,tcp and http are different.

Should be bind two port when Start service.these are tcp message port and http message port

## TCP

Tcp is fast and more futures.

## HTTP

Send Message through post,and recv message through get

# 二 Configure Env

## 2.1 WINDOWS

Need to download XEngine.

Complie and run and debug by vs2019.

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

### 2.1.1 Configure Environment

After Download XEngine,if you download is zip file.you have to uncompress for xengine and add user environment value.

You need add to follow two user environment in your system

* XEngine\_Include  XEngine is header path
* XEngine\_Library  XEngine is library path

比如:



You need jsoncpp env.you can install through vcpkg

And you can download for youself:https://github.com/open-source-parsers/jsoncpp/

If you download for youself,you need install and complie by youself and configure project attributes vc++ path in your vs2019

### 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 x86(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

### 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\_RunEnv.sh -i 3

of course,and need install jsoncpp environment to your system.

Ubuntu:sudo apt install libjsoncpp-devel -y

Centos:sudo dnf install jsoncpp-devel -y

### 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\_MQServiceApp 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:V7.14

JsonCpp:V1.9.2

# 三 Interface Protocol

## 3.1 TCP

Please Note:The wReserver field of the protocol header will be used as the server reply processing result. 0 means the processing is successful, other values mean failure  
 whether protocol load is general protocol or HTTP protocol,they can be load json data.

### 3.1.1 Post Protocol

To use the message queue, you need to deliver a packet to the message queue server first, so that other programs can get a packet from the message queue service.

#### 3.1.1.1 Request

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

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

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REQPOST

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ) + MSGLEN

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:

typedef struct tag\_XEngine\_ProtocolXmq

{

*TCHAR* tszMQKey[256];

\_int64x nSerial;

int nKeepTime;

int nGetTimer;

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

DATA in tail

#### 3.1.1.2 Reply

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

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

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REPPOST

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:

typedef struct tag\_XEngine\_ProtocolXmq

{

*TCHAR* tszMQKey[256];

\_int64x nSerial;

int nKeepTime;

int nGetTimer;

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

### 3.1.2 Get Protocol

#### 3.1.2.1 Requestion

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

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

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REQGET

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:

typedef struct tag\_XEngine\_ProtocolXmq

{

*TCHAR* tszMQKey[256];

\_int64x nSerial;

int nKeepTime;

int nGetTimer;

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

#### 3.1.2.2 Reply

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

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

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REPGET

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ) + MSGLEN

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:

typedef struct tag\_XEngine\_ProtocolXmq

{

*TCHAR* tszMQKey[256];

\_int64x nSerial;

int nKeepTime;

int nGetTimer;

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

Protocol Body:if sucess,protocol follow get data

### 3.1.3 Delete Protocol

#### 3.1.3.1 Requestion

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

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

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

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:

typedef struct tag\_XEngine\_ProtocolXmq

{

*TCHAR* tszMQKey[256];

\_int64x nSerial;

int nKeepTime;

int nGetTimer;

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

#### 3.1.3.2 Reply

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

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

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

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:

typedef struct tag\_XEngine\_ProtocolXmq

{

*TCHAR* tszMQKey[256];

\_int64x nSerial;

int nKeepTime;

int nGetTimer;

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

### 3.1.4 Create Topic

Post a packet to message queue,you must create a topic,you have to set topic name,data can be enter the message queue

#### 3.1.4.1 Requestion

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

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

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REQCREATE

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:

typedef struct tag\_XEngine\_ProtocolXmq

{

*TCHAR* tszMQKey[256];

\_int64x nSerial;

int nKeepTime;

int nGetTimer;

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

#### 3.1.4.2 Reply

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

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

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REPCREATE

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:

typedef struct tag\_XEngine\_ProtocolXmq

{

*TCHAR* tszMQKey[256];

\_int64x nSerial;

int nKeepTime;

int nGetTimer;

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

### 3.1.5 Delete Topic

#### 3.1.5.1 Requestion

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

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

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REQDELETE

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:

typedef struct tag\_XEngine\_ProtocolXmq

{

*TCHAR* tszMQKey[256];

\_int64x nSerial;

int nKeepTime;

int nGetTimer;

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

#### 3.1.5.2 Reply

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

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

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REPDELETE

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:

typedef struct tag\_XEngine\_ProtocolXmq

{

*TCHAR* tszMQKey[256];

\_int64x nSerial;

int nKeepTime;

int nGetTimer;

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

### 3.1.6 Subscribe Topic

Subscribing to the topic allows the user to actively push a message to the subscribed client every time there is new content in the topic after the request is successfully subscribed

#### 3.1.6.1 Requestion

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

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

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REQNOTIFY

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:

typedef struct tag\_XEngine\_ProtocolXmq

{

*TCHAR* tszMQKey[256];

\_int64x nSerial;

int nKeepTime;

int nGetTimer;

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

#### 3.1.6.2 Reply

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

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

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REPNOTIFY

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:

typedef struct tag\_XEngine\_ProtocolXmq

{

*TCHAR* tszMQKey[256];

\_int64x nSerial;

int nKeepTime;

int nGetTimer;

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

### 3.1.7 Notification Message

Protocol Header:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

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

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_MSGNOTIFY

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ) + message

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

Protocol Body:

typedef struct tag\_XEngine\_ProtocolXmq

{

*TCHAR* tszMQKey[256];

\_int64x nSerial;

int nKeepTime;

int nGetTimer;

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

...body

# 四 Configure Description

## 4.1 Service Configure

Basic Configure File:XEngine\_Config.json

### 4.1.1 basic configure

* bDeamon: 1 deamon process run 0 is terminal run
* tszTopic:Default Topic Name
* nTCPPort:tcp port
* nHttpPort:http port

### 4.1.2 Max Configure

XMax Configure

* MaxClient Allow Max Client Count
* MaxQueue Allow Max Queue
* IOThread:network io process threads number
* nTCPThread:tcp handle threads number
* nHttpThread:http handle threads number

### 4.1.3 Time Configure

XTime Configure

* bHBTime,1 is enable heartbeat,0 disable
* nDBMonth:database save time,default month
* nTimeCheck:check time
* nTCPTimeOut:how time check once
* nHttpTimeOut:same nTCPTimeOut

### 4.1.4 Log Configure

XLog Configure

* MaxSize:Log file size
* MaxCount:Log File Number
* LogLeave:Allow save level

### 4.1.5 Database Configure

XSql Configure

# 五 Advanced configuration

# 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

## Appendix 4 update log