XEngine网络存储服务文档

目录

[XEngine网络存储服务文档 1](#_Toc22536)

[前言 4](#_Toc10567)

[阅读者 4](#_Toc30513)

[概述 4](#_Toc4656)

[相关模块 4](#_Toc5615)

[XEngine 4](#_Toc9134)

[三方库 4](#_Toc10733)

[系统环境 4](#_Toc22890)

[一 技术结构 5](#_Toc21045)

[1.1 上传接口 5](#_Toc10309)

[1.2 下载接口 5](#_Toc8754)

[1.3 管理接口 5](#_Toc18367)

[二 配置环境 5](#_Toc17928)

[2.1 WINDOWS 5](#_Toc17295)

[2.2 LINUX 6](#_Toc21060)

[三 接口协议 6](#_Toc32498)

[3.1 管理协议 7](#_Toc25205)

[3.1.1 文件列表 7](#_Toc24442)

[3.1.2 增加文件 8](#_Toc1328)

[3.1.3 删除文件 8](#_Toc17261)

[3.1.4 任务查询协议 9](#_Toc10633)

[3.1.5 文件夹协议 10](#_Toc18413)

[3.1.6 请求文件 11](#_Toc8460)

[3.2 三方接口 12](#_Toc17787)

[3.2.1 用户验证 12](#_Toc8886)

[3.2.2 完成通知 13](#_Toc23968)

[3.3 P2XP协议 13](#_Toc24192)

[3.3.1 加入网络 13](#_Toc2743)

[3.3.2 同步列表 14](#_Toc28281)

[3.3.3 用户查询 16](#_Toc27440)

[3.3.4 请求连接 17](#_Toc6534)

[3.4 上传下载 18](#_Toc3013)

[3.4.1 上传 18](#_Toc15674)

[3.4.2 下载 18](#_Toc20595)

[四 配置说明 18](#_Toc8045)

[4.1 服务器配置 18](#_Toc23103)

[4.1.1 基本配置 19](#_Toc11636)

[4.1.2 最大配置 19](#_Toc30729)

[4.1.3 时间配置 19](#_Toc7716)

[4.1.4 日志配置 19](#_Toc25330)

[4.1.5 数据库配置 19](#_Toc21781)

[4.1.6 存储配置 20](#_Toc19251)

[4.1.7 代理配置 20](#_Toc6325)

[4.1.8 限制配置 20](#_Toc8267)

[4.1.9 P2XP配置 20](#_Toc31272)

[4.1.10 证书配置 21](#_Toc6120)

[4.1.11 版本配置 21](#_Toc1296)

[4.2 分布式配置 21](#_Toc20748)

[4.2.1 基本配置 21](#_Toc16049)

[4.2.2 负载均衡 21](#_Toc7462)

[4.2.3 本地负载 21](#_Toc30474)

[4.2.4 负载属性 22](#_Toc1859)

[五 高级配置 22](#_Toc17368)

[5.1 分布式 22](#_Toc12679)

[5.1.1 网络分布式 22](#_Toc31909)

[5.1.2 存储分布式 22](#_Toc9085)

[5.2 HTTPS 23](#_Toc2122)

[FAQ 23](#_Toc14356)

[附录 23](#_Toc3986)

[附录1 类型定义 23](#_Toc4151)

[附录2 协议定义 23](#_Toc1501)

[附录3 转换定义 23](#_Toc22365)

[附录4 更新历史 23](#_Toc16372)

|  |  |  |  |
| --- | --- | --- | --- |
| 文件状态：  [ ] 草稿  [√] 正式发布 | 文件名称： | XEngine网络存储服务文档 | |
| 适用版本： | V3.4 | |
| 发布日期： | 2022-01-21 | |
| 拟 制： qyt | | |

# 前言

## 阅读者

开发人员,测试人员

## **概述**

此文档包含了存储服务的相关技术说明和接口定义!

## 相关模块

### XEngine

此服务使用了XEngine作为开发包.要使用此服务代码必须配置安装好XEngine开发环境.

下载地址:<https://gitee.com/xyry/libxengine>

<https://github.com/libxengine/xengine>

### 三方库

此服务还使用了第三方模块jsoncpp

Windows:

你可以通过VCPKG来安装JSONCPP.

也可以自己下载:https://github.com/open-source-parsers/jsoncpp/

如果是自己下载,你需要自己安装和编译.并且在我们的项目中-属性-VC++目录 配置你的JSONCPP目录才可使用.

Linux:

Ubuntu:sudo apt install libjsoncpp-devel -y

Centos:sudo dnf install jsoncpp-devel -y

## 系统环境

Windows需要在WIN7SP1 以及以上系统.拥有最新补丁.使用VS2019编译运行

Linux需要Ubuntu20.04或者Centos8以上系统,使用X64.使用makefile编译

# 一 技术结构

采用的是HTTP协议来作为基础通信协议.上传,下载,管理接口都是HTTP

P2P需要用TCP协议,因为要保持连接

服务器启动会绑定四个端口,来作为区分上传,下载,管理,P2P.

四个端口不可以跨端口使用,每个端口做的事情是固定的

## 上传接口

上传接口采用HTTP实现,上传接口需要使用HTTP的PUT方法实现.

上传接口不需要使用form-data头字段.

上传的路径需要由用户通过接口创建或者指定.

上传的路径为PUT /dir/name HTTP/1.1 取中间的URL地址

## 下载接口

下载接口通过GET实现

下载的文件通过 GET /bucketkey/dir/name HTTP/1.1 中间的URL地址获得

BucketKey为分布式存储的每个存储路径所编码的唯一键值.

## 管理接口

管理接口通过POST实现,POST可以为空的负载信息,可以带JSON作为负载信息.根据每个接口的不同,负载的内容也不同,但是都是通过JSON来实现的

管理接口需要使用 /api/type/name 固定三元组形势提供.表示API版本,API类型,API名称.

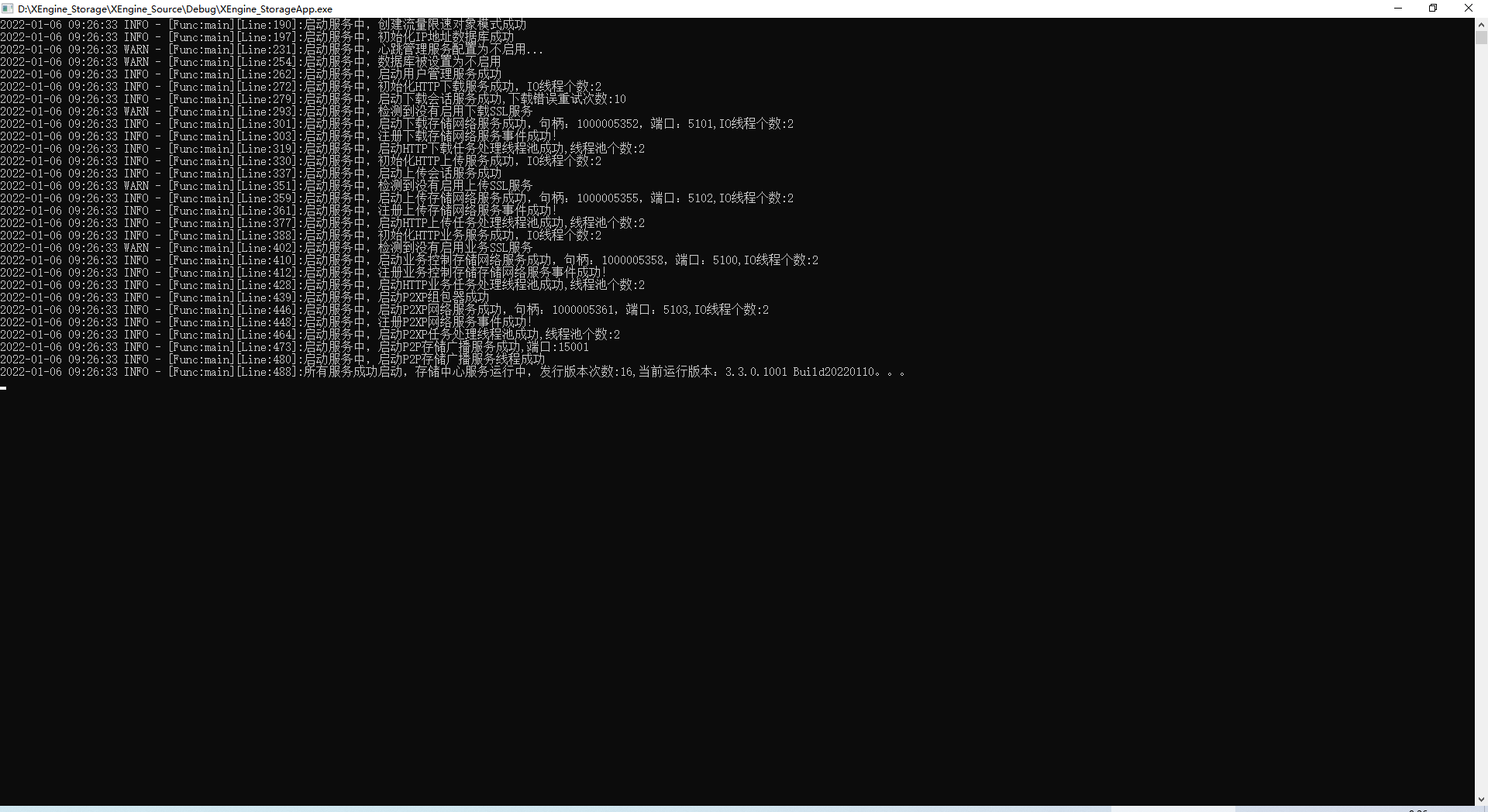
# 二 配置环境

## 2.1 WINDOWS

在完成环境配置后.你可以进入代码目录.使用VS2019打开XEngine\_StorageApp.sln,然后选择x86 debug 编译.

如果环境没有错误,编译会直接成功.包含6个模块和一个EXE程序

成功后,你需要拷贝XEngine\_Release下的文件到你编译的目录下.然后拷贝依赖的XEngine模块与JSONCPP模块到你的编译目录下,运行XEngine\_StorageApp.exe即可.如果没有错误,你可以看到下面的界面信息



注意：你可以拷贝VSCopy.bat脚本到你的程序目录,此脚本会自动拷贝依赖模块.

## 2.2 LINUX

配置完环境后,你可以编译它,打开终端,进入源码目录执行以下命令

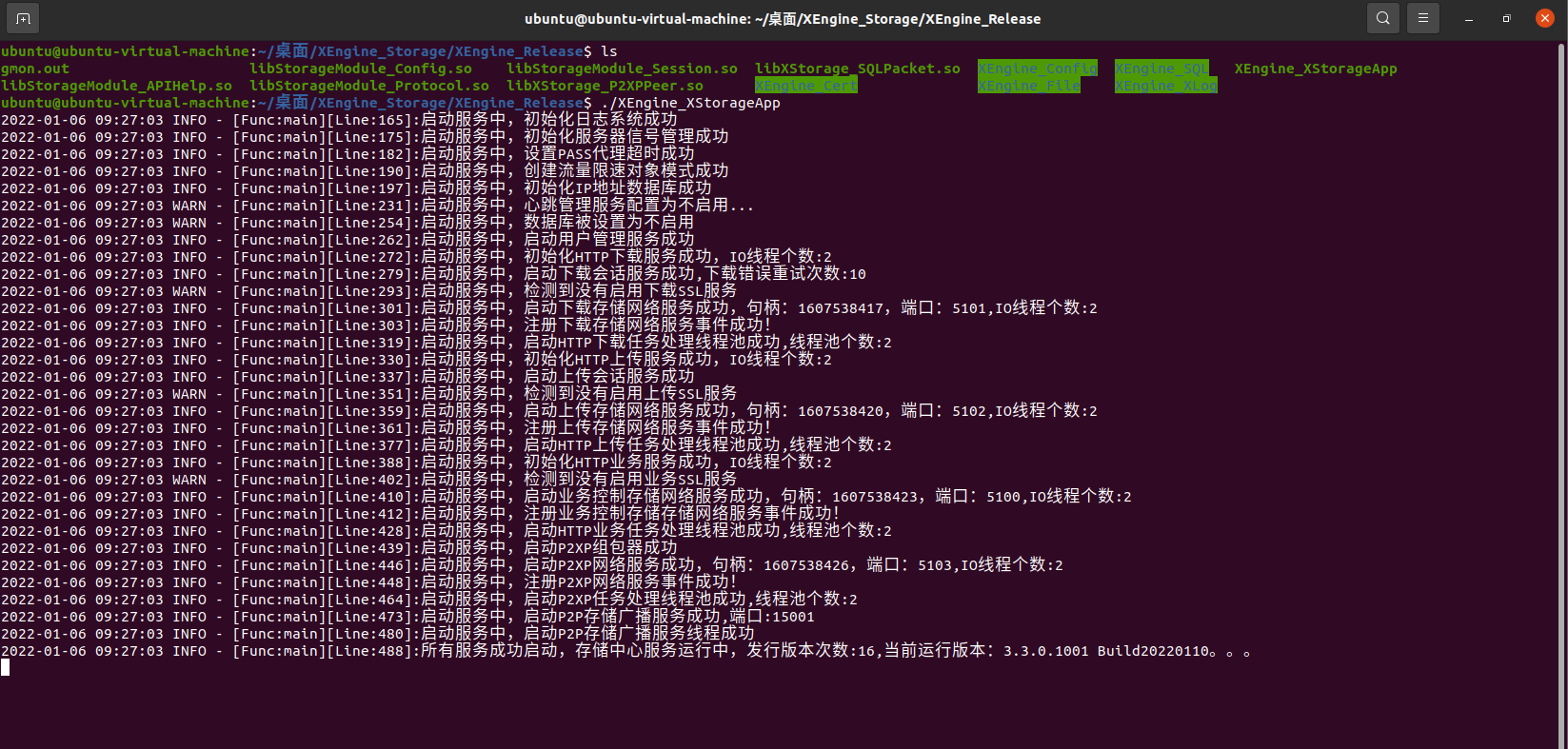
编译:make

安装:make FLAGS=InstallAll

清理:make FLAGS=CleanAll

如果没有错误,你可以在XEngine\_Release目录下看到编译的XEngine\_StorageApp

然后直接在终端运行它即可.如果没有错误,你可以看到下面的信息:



# 三 接口协议

我们的接口都是POST接口协议,也就是业务端口.

## 3.1 管理协议

### 3.1.1 文件列表

#### 3.1.1.1 请求

请求接口:/Api/Manage/Query

请求方法:POST

请求内容:可空或者如下:

{  
    **"lpszTimeStart"**:**"开始时间,可NULL"**,  
    **"lpszTimeEnd"**:**"结束时间,可NULL"**,

**"lpszBuckKey"**:**"指定查询的BUCKET,可以为NULL"**,  
    **"lpszFileName"**:**"文件名,可NULL"**,  
    **"lpszFileHash"**:**"文件HASH,可NULL"**  
}

#### 3.1.1.2 回复

{  
    **"Code"**:**0**,  
    **"Count"**:**1**,  
    **"List"**:[  
        {  
            **"nFileSize"**:**23897183**,

**"tszBuckKey"**:**"Bucket名称"**,  
            **"tszFileHash"**:**"85E62C9D28FCE775A68DD126E0519F25"**,  
            **"tszFileName"**:**"1.docx"**,  
            **"tszFilePath"**:**"./XEngine\_File"**,  
            **"tszFileTime"**:**"2021-07-12 05:59:17"**,  
            **"tszFileUser"**:**""**  
        }  
    ],  
    **"Msg"**:**"ok"**,  
    **"lpszTimeEnd"**:**""**,  
    **"lpszTimeStart"**:**""**  
}

### 3.1.2 增加文件

如果想要P2P存储网络找到这个文件,你需要增加这个文件到存储系统中.

基于P2P的文件传输工作需要在数据库里面有记录,文件下载需要用户自己实现,所以当下载完成一个文件后,想要让这个文件加入P2P网络,那么需要提交一个记录到本地的存储服务中(请注意:不是提交给服务器,而是本机存储网络服务器).

#### 3.1.2.1 请求

tszFileName和tszFilePath字段不能为空,表明文件位置.其他可以全为空

请求接口:/Api/Manage/Insert

请求方法:POST

请求内容:tszBuckKey和tszFilePath只能有一个存在

{  
    **"Count"**:**1**,  
    **"List"**:[  
        {  
            **"nFileSize"**:**23897183**,

**"tszBuckKey"**:**"Bucket Name"**,  
            **"tszFileHash"**:**"85E62C9D28FCE775A68DD126E0519F25"**,  
            **"tszFileName"**:**"1.docx"**,  
            **"tszFilePath"**:**"./XEngine\_File"**,  
            **"tszFileTime"**:**"2021-07-12 05:59:17"**,  
            **"tszFileUser"**:**""**  
        }  
    ]  
}

#### 3.1.2.2 回复

HTTP CODE 200 = 成功

### 3.1.3 删除文件

删除请求可以直接删除文件,也可以提交一个删除请求给服务器

#### 3.1.3.1 请求

可以使用文件地址的方式,也可以只使用HASH

请求接口:/Api/Manage/Delete

请求方法:POST

请求内容:tszBuckKey和tszFilePath只能有一个存在

{  
    **"Count"**:**1**,  
    **"List"**:[  
        {  
            **"tszFileHash"**:**"85E62C9D28FCE775A68DD126E0519F25"**,  
            **"tszFileName"**:**"1.docx"**,

**"tszBuckKey"**:**"Bucket Key Name",**  
            **"tszFilePath"**:**"./XEngine\_File"**  
        }  
    ]  
}

#### 3.1.3.2 回复

HTTP CODE

### 3.1.4 任务查询协议

任务查询协议可以查询得到当前存储服务正在上传和下载的所有任务信息,方便用户做管理接口.

#### 3.1.4.1 请求

请求接口:/Api/Manage/Task

请求方法:POST

请求内容:无

#### 3.1.4.2 回复

* ullCount:文件总大小
* ullRWCount:客户端需要读写的总大小
* ullRWLen:当前读写的大小
* ullPosStart:客户端要求的开始偏移
* ullPosEnd:客户端要求的结尾偏移

{  
    **"Code"**:**0**,  
    **"Msg"**:**"ok"**,  
    **"DLCount"**:**1**,  
    **"UPCount"**:**1**,  
    **"DLList"**:[  
        {  
            **"tszClientAddr"**:**"客户端IP地址"**,  
            **"tszFilePath"**:**"文件路径"**,  
            **"ullCount"**:**100**,  
            **"ullRWCount"**:**90**,  
            **"ullRWLen"**:**50**,  
            **"ullPosStart"**:**10**,  
            **"ullPosEnd"**:**100**  
        }  
    ],  
    **"UPList"**:[  
        {  
            **"tszClientAddr"**:**"客户端IP地址"**,  
            **"tszFilePath"**:**"文件路径"**,  
            **"ullCount"**:**100**,  
            **"ullRWCount"**:**90**,  
            **"ullRWLen"**:**50**,  
            **"ullPosStart"**:**10**,  
            **"ullPosEnd"**:**100**  
        }  
    ]  
}

### 3.1.5 文件夹协议

此协议可以提供文件夹查询,创建,删除操作

#### 3.1.5.1 请求

请求接口:/Api/Manage/Diry

请求方法:POST

请求内容:

{  
    **"nOPerator"**:**0**,

**"lpszUserDir"**:**"要操作的子文件夹",**  
    **"lpszBuckKey"**:**"Bucket name"**  
}

* 0:查询
* 1:创建
* 2:删除

#### 3.1.5.2 回复

创建和删除回复的标准HTTP信息,查询回复的下面的内容

{  
    **"Code"**:**0**,  
    **"Count"**:**3**,  
    **"List"**:[  
        {  
            **"tszFilePath"**:**"./XEngine\_File/Scene/scene1"**  
        },  
        {  
            **"tszFilePath"**:**"./XEngine\_File/Scene/scene1/tmp"**  
        },  
        {  
            **"tszFilePath"**:**"./XEngine\_File/Scene/scene2"**  
        }  
    ],  
    **"Msg"**:**"ok"**  
}

### 3.1.6 请求文件

请求分布式文件,用于请求本地局域网中的文件列表

#### 3.1.6.1 请求

请求接口:Api/Manage/P2P

请求方法:POST

请求内容:HASH和名字只需要一个存在

{  
    **"lpszBuckKey"**:**"Bucket name"**,  
    **"lpszFileHash"**:**"File Hash"**,  
    **"lpszFileName"**:**"File Name"**  
}

#### 3.1.6.2 回复

如果回复的tszTableName为127.0.0.1表示本地文件存在.不会请求局域网文件.否则这个值为远程地址+端口.

{  
    **"Code"**:**0**,  
    **"Count"**:**1**,  
    **"List"**:[  
        {  
            **"nFileSize"**:**92674736**,

**"tszBuckKey"**:**"bucket key name"**,  
            **"tszFileHash"**:**"EC9B9B75A04F3B323EFD348F9B795539"**,  
            **"tszFileName"**:**"qq.exe"**,  
            **"tszFilePath"**:**"./XEngine\_File/scene1"**,  
            **"tszFileTime"**:**"2021-07-30 11:01:04"**,  
            **"tszFileUser"**:**""**,  
            **"tszTableName"**:**"127.0.0.1"**  
        }  
    ],  
    **"Msg"**:**"ok"**  
}

## 3.2 三方接口

三方接口是用来接入第三方服务器的接口

### 3.2.1 用户验证

用户验证目前仅仅支持BASIC的HTTP验证.你可以通过配置我们的UserList.txt文件来实现用户验证.也可以使用HTTP PASS代理来实现验证.

#### 3.2.1.1 本地验证

在XEngine\_Config文件夹的UserList.txt里面有用户列表.

每一行代表一个用户,中间用空格分开,前面表示用户名,后面表示密码.你只需要配置这个文件即可实现HTTP 基本的验证.没有验证通过将无法继续操作.

#### 3.2.1.2 代理验证

代理验证是由服务器发送一条HTTP POST消息给一个指定地址,返回200才表示成功,其他值表示失败.

发送的HTTP POST代理地址由用户指定,负载内容由服务器指定,负载为JSON格式,内容如下:

{

**"lpszMethod"**:**"使用的方法"**,  
    **"lpszPostUrl"**:**"客户端提交的URL"**,  
    **"lpszClientAddr"**:**"客户端的IP地址"**,  
    **"lpszUser"**:**"用户"**,  
    **"lpszPass"**:**"密码"**  
}

如果成功,那么需要返回200.

### 3.2.2 完成通知

完成通知表示服务器在接受到一个上传和下载请求处理完毕后,是否需要给指定服务发送一个HTTP POST协议的通知.可以通过配置文件配置.

完成通知的内容负载是相同的,根据URL来区分,负载信息如下.

bDown 1表示下载,0表示上传

{

**"bDown"**:1,

**"lpszBuckKey"**:**"Bucket key name"**,  
    **"lpszFileName"**:**"文件名"**,  
    **"lpszFileHash"**:**"文件HASH,可能为NULL"**,  
    **"lpszClientAddr"**:**"请求的客户端地址"**,  
    **"nFileSize"**:**33333**  
}

## 3.3 P2XP协议

P2XP同样使用HTTP协议实现,P2XP需要用到用户信息,要使用P2XP协议,调用者可能需要单独开发一套验证接口.用于验证用户名信息.

当用户通过P2XP登录后,每隔5秒还需要进行心跳保持信息,只有超过指定时间没收到,那么才会认为不在线.

P2XP使用P2XP端口.

P2XP协议使用标准协议头(参考XEngine的SDK文档)+JSON方式传输数据

### 3.3.1 加入网络

只有发送这个协议请求,服务器才会记录这个客户端,客户端才能加入P2XP网络,否则将无法使用P2XP网络.

此功能主要为了跨网段使用分布式存储网络,如果你没有这方便的需求,可以不使用

#### 3.3.1.1 请求

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_P2XP

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

unPacketSize = JSONSIZE

byVersion = 2

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:

{  
    **"tszUserName"**:**"用户名,由使用者管理,服务器不做验证"**,  
    **"tszPrivateAddr"**:**"私有地址,本机内部IP地址"**,  
    **"tszPublicAddr"**:**"公有地址"**,  
    **"dwConnectType"**:**0**,  
    **"dwPeerType"**:**0**  
}

#### 3.3.1.2 回复

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_P2XP

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

unPacketSize = JSONSIZE

byVersion = 2

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:

{  
    **"nCode"**:**0**,  
    **"lpszMsgBuffer"**:**"sucess"**  
}

### 3.3.2 同步列表

内网地址只同步内网IP,外网地址同步可以同步在这个外网IP地址下面的所有内网IP地址,对于一些大型网络公司和环境复杂的内网,使用此功能可以方便判断出在同一网络下的用户.

#### 3.3.2.1 请求

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_P2XP

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_P2XP\_REQLANLIST

unPacketSize = JSONSIZE

byVersion = 2

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:

{  
    **"tszPublicAddr"**:**"公有地址"**,  
    **"tszPrivateAddr"**:**"私有地址,为NULL表示获取公网下所有地址"**  
}

#### 3.3.2.2 回复

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_P2XP

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_P2XP\_REPLANLIST

unPacketSize = JSONSIZE

byVersion = 2

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:

{  
    **"nCode"**:**0**,  
    **"lpszMsgBuffer"**:**"sucess"**,  
    **"ClientCount"**:**2**,  
    **"ClientArray"**:[  
        {  
            **"ClientAddr"**:**"192.168.1.101"**,  
            **"ClientUser"**:**"123123aa"**  
        },  
        {  
            **"ClientAddr"**:**"192.168.1.102"**,  
            **"ClientUser"**:**"123123bb"**  
        }  
    ]  
}

### 3.3.3 用户查询

#### 3.3.3.1 请求

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_P2XP

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_P2XP\_REQUSERQUERY

unPacketSize = JSONSIZE

byVersion = 2

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:

{  
    **"tszUserName"**:**"要查询的用户"**  
}

#### 3.3.3.2 回复

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_P2XP

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_P2XP\_REPUSERQUERY

unPacketSize = JSONSIZE

byVersion = 2

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:

{  
    **"nCode"**:**0**,  
    **"lpszMsgBuffer"**:**"sucess"**,  
    **"dwConnectType"**:**1**,  
    **"dwPeerType"**:**1**,  
    **"tszConnectAddr"**:**"192.168.1.101:33990"**,  
    **"tszPrivateAddr"**:**"192.168.1.101"**,  
    **"tszPublicAddr"**:**"114.114.114.114"**,  
    **"tszUserLocation"**:**"四川省成都市"**,  
    **"tszUserISP"**:**"电信"**,  
    **"tszUserName"**:**"123123aa"**  
}

### 3.3.4 请求连接

请求连接协议可以要求一个客户端连接到你的指定地址和端口.这个协议用于点对点传输数据的时候最开始请求的协议,服务器在收到请求连接协议后,会把这条数据转发给客户端,客户端会受到与请求服务器一样的协议,这个时候客户端需要主动连接到指定的IP地址和端口!

我们没有制定点对点传输的协议,所以需要开发者自己指定,我们只负责打通数据!

#### 3.3.4.1 请求

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_P2XP

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_P2XP\_REQCONNECT

unPacketSize = JSONSIZE

byVersion = 2

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:

{  
    **"tszSourceUser"**:**"请求的用户"**,  
    **"tszDestUser"**:**"目标用户"**,  
    **"tszConnectAddr"**:**"连接的地址"**,  
    **"nDestPort"**:**5000**,  
    **"bIsTcp"**:**1**  
}

#### 3.3.4.2 回复

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_P2XP

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_P2XP\_REPCONNECT

unPacketSize = JSONSIZE

byVersion = 2

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:

{  
    **"nCode"**:**0**,  
    **"lpszMsgBuffer"**:**"sucess"**  
}

## 3.4 上传下载

### 3.4.1 上传

上传可以使用POST和PUT方法,上传接口是否需要指定bucket需要由系统配置.上传的接口必须包含Content-Length 字段,如果想使用断点续传,那么需要包含Range字段.

上传完毕后支系统返回200,并且返回3.2.2的JSON负载信息.

### 3.4.2 下载

下载接口使用GET方法获取.下载文件支持断点续传.

在你请求下载后,系统会在HTTP头中返回一个FileHash 字段,表明这个文件的文件HASH值.后续数据就是下载的数据

# 四 配置说明

## 4.1 服务器配置

基础配置文件:XEngine\_Config.json

### 4.1.1 基本配置

* bDeamon: 1为守护进程后台运行0为前台运行
* nCenterPort:业务处理端口,HTTP POST处理端口
* nStorageDLPort:下载端口GET 协议
* nStorageUPPort:上传端口 PUT协议
* nP2XPPort:P2XP协议传输端口

### 4.1.2 最大配置

XMax 配置

* MaxClient 最大允许客户端
* MaxQueue 最大允许队列
* IOThread:网络IO处理线程个数
* CenterThread:业务处理线程个数
* nStorageUPThread:上传处理线程个数
* nStorageDLThread:下载处理线程个数
* nP2XPThread:P2XP处理线程个数

### 4.1.3 时间配置

XTime配置

* bHBTime是否启用,1为启用0不启用心跳
* nDBMonth:数据库保存时间,默认月
* nTimeCheck:检测次数
* nStorageTimeOut:间隔多久检查一次
* nP2XPTimeOut:P2XP 心跳超时时间

### 4.1.4 日志配置

XLog 配置

* MaxSize:日志文件大小
* MaxCount:日志文件个数
* LogLeave:允许保存的级别

### 4.1.5 数据库配置

XSql 配置.支持MYSQL或者SQLITE

如果想启用P2P支持,必须在其他客户端上使用SQLITE数据库

* SQLType:数据库启用内型,0为不使用数据库,1为MYSQL,2为SQLITE
* SQLFile:类型为2有效,表示SQLITE数据库保存位置

### 4.1.6 存储配置

XStorage 配置

* nHashMode:HASH算法,1MD5,2HASH1,具体值查看OPENSSL的定义
* bResumable:Whether to enable resumable transmission
* bUPHash:是否强制上传文件带上文件hash字段

### 4.1.7 代理配置

XProxy 配置

#### 4.1.7.1 验证代理

XProxyAuth 配置

* bAuth:是否启用验证
* tszUserList: 用户列表地址
* tszAuthProxy:使用远程验证,通过HTTP POST来验证,参考3.2.1.2,如果为空表示使用本地列表,否则请填写一个HTTP地址

#### 4.1.7.2 完成通知

XProxyPass 配置

* bUPGet:是否启用上传完成通知
* bDLGet:是否启用下载完成通知
* tszUPPass:用户上传完成通知地址
* tszDLPass:用户下载完成通知地址

注意:配置了这个,服务将等待你的返回结果,并且根据是否是200来返回成功和失败给客户端

### 4.1.8 限制配置

XLimit 配置

* nDLTry:下载服务重试次数
* nMaxUPLoad:最大上传速率.0不限制,单位字节(BYTE)
* nMaxDNLoad:最大下载速率,同上

### 4.1.9 P2XP配置

P2XP配置:XP2xp

nMode:0,不启用,1使用广播模式,2使用直链模式.中心服务器不能使用P2P

nTime:最大处理存储网络时间,单位秒,会一直处理完此时间

nSDPort:发送端口,用于广播

nRVPort:接受端口,用于广播

tszQQWryFile:加载的IP地址库路径

### 4.1.10 证书配置

证书配置:XCert

bDLEnable:1启用下载SSL证书

bUPEnable:1启用上传SSL证书

bCHEnable:1启用业务SSL证书

nSslType:SSL还是TLS,参考XEngine\_OPssl模块

tszCertChain:证书链地址

tszCertKey:证书密钥

### 4.1.11 版本配置

XVer 配置:用户显示当前版本号

## 4.2 分布式配置

配置文件:XEngine\_LBConfig.json

### 4.2.1 基本配置

* bDistributed:是否启用分布式,如果为0,下面的内容全不起作用

### 4.2.2 负载均衡

LBDistributed配置,用于重定向

* nCenterMode:中心服务分布式配置,模式,0 不启用,1 随机选择,2 顺序选择,3倒序选择
* nUPLoadMode:上传分布式配置
* nDownldMode:下载分布式配置
* nStorageMode:存储分布式配置

### 4.2.3 本地负载

LBLocation配置

* nCenterMode:暂时无效
* nUPLoadMode:存储到key的选择模式,1 随机选择,2 顺序选择,3倒序选择,4 必须指定
* nDownldMode:暂时无效
* nStorageMode:暂时无效

### 4.2.4 负载属性

下面的内容通过重定向实现,客户端必须支持302重定向请求

* CenterAddr:任务处理负载地址池
* DownloadAddr:下载服务地址列表
* UPLoaderAddr:上传服务地址列表
* StorageAddr:分布式存储

bEnable:是否启用

nLevel:存储优先级别,越小优先级越高

Size:允许大小,KB,MB,GB

XEngine\_Key:Bucket key,用于隐藏路径.会对应path

XEngine\_Path:保存的路径

# 五 高级配置

## 5.1 分布式

### 5.1.1 网络分布式

分布式服务通过HTTP重定向实现,客户端必须支持302才能实现分布式协议.

通过分布式配置来处理.你可以通过数组配置一种或者多种分布式处理逻辑

在使用分布式的时候,最后的逻辑分布式配置文件一定是关闭的.

#### 5.1.1.1 配置多个

分布式配置:LoadBalance

你可以在数组中添加多个地址指向其他服务器.这个时候,此服务器就是个分布式服务器,他不处理你指定的nUseMode业务配置.而是交给多个后端来处理.

那么后端处理就需要设置nUseMode为0关闭.

### 5.1.2 存储分布式

存储分布式需要在你的客户端机器上部署此存储服务器,并且启用SQLITE数据库.

现在可以通过局域网查询和下载文件了,你可以优先请求本地的存储服务,存储服务会自动广播查询局域网中的文件.如果存在,会返回你3.1.1.2节的内容,并且会增加一个字段tszTableName表示服务器IP和端口.他会返回所有查询到的可用本地局域网节点列表,你可以选择一个下载或者通过HTTP Range字段来连接多个做分布式块下载.

## 5.2 HTTPS

在3.1版本后,我们的服务支持HTTPS访问了.要使用HTTPS,你需要拥有一个证书,证书可以自己创建或者去申请一个免费的或者买一个.

我们使用的证书是证书链证书加证书密钥的方式(就像NGINX一样的方式),不验证对方证书.如果你有高级需求,可以修改代码使用其他证书类型或者要求客户端也有证书.

# FAQ

1. 我们不建议使用上传限速.最好设置为0.

# 附录

## 附录1 类型定义

参考文件 XEngine\_CommHdr.h

## 附录2 协议定义

参考文件XEngine\_ProtocolHdr.h

## 附录3 转换定义

参考文件 XEngine\_Types.h

## 附录4 更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| 更新日期 | 更新说明 | 修改者 | 适用版本 |