

廈門大學



信息学院软件工程系

《计算机网络》实验报告

题 目 实验五 利用 Socket API 实现许可认证软件

班 级 数字媒体技术 2022 级 1 班

姓 名 魏清晨

学 号 37220222203790

实验时间 2024 年 11 月 6 日

2024 年 11 月 6 日

填写说明

- 1、本文件为 Word 模板文件，建议使用 Microsoft Word 2021 打开，在可填写的区域中如实填写；
- 2、填表时勿改变字体字号，保持排版工整，打印为 PDF 文件提交；
- 3、文件总大小尽量控制在 1MB 以下，最大勿超过 5MB；
- 4、应将材料清单上传在代码托管平台上；
- 5、在实验课结束 14 天内，按原文件发送至课程 FTP 指定位置。

1 实验目的

通过完成实验，掌握应用层文件传输的原理；了解传输过程中传输层协议选用、应用层协议设计和协议开发等概念。

2 实验环境

操作系统：Win11

语言：C#、PHP

IDE：Rider、PHPStorm

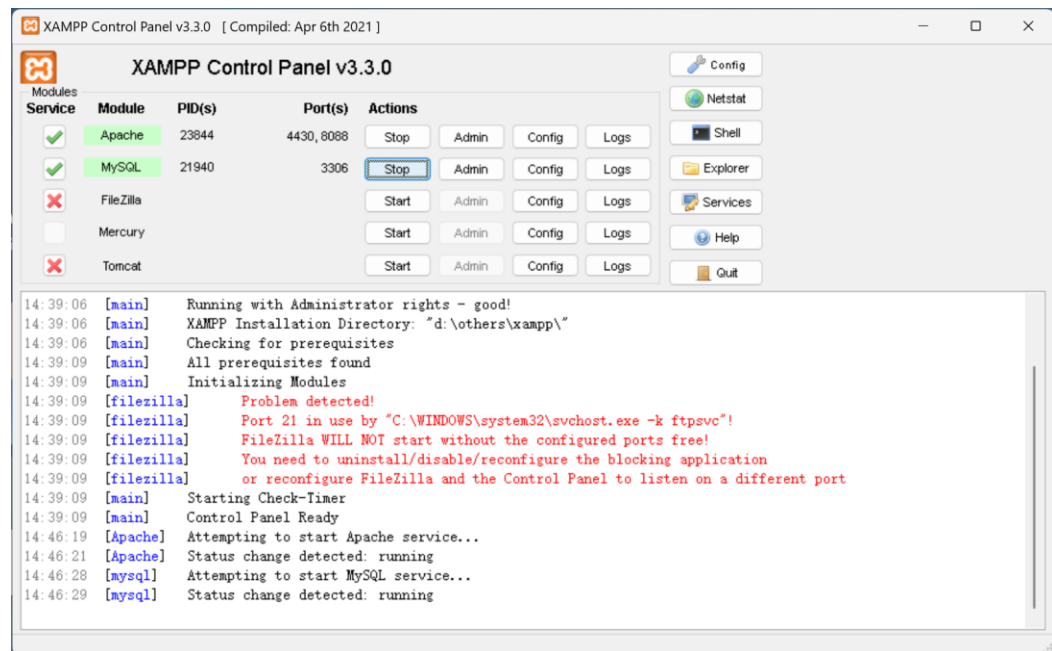
服务器、数据库环境工具：xampp

3 实验结果

一、实验配置

1. 服务器配置

服务器平台使用 xampp



https 端口为 4430

```
#
# When we also provide SSL we have to listen to the
# standard HTTP port (see above) and to the HTTPS port
#
Listen 4430
```

本地服务器的域名以及 ssl 证书的配置（此处域名、ssl 证书都通过华为云平台申请获得）

（httpd-ssl.conf）

```
<VirtualHost _default_:4430>

#   General setup for the virtual host
DocumentRoot "E:/code/webroot"
ServerName visionary.net.cn:4430
ServerAlias *.visionary.net.cn
ServerAdmin admin@visioanry.net.cn
ErrorLog "D:/others/xampp/apache/logs/error.log"
TransferLog "D:/others/xampp/apache/logs/access.log"

#   SSL Engine Switch:
#   Enable/Disable SSL for this virtual host.
SSLEngine on

SSLCertificateFile "conf/ssl.crt/scs1731818806597_visionary.net.cn_server.crt"
# SSLCertificateFile "conf/ssl.crt/server.crt"

SSLCertificateKeyFile "conf/ssl.key/scs1731818806597_visionary.net.cn_server.key"
# SSLCertificateKeyFile "conf/ssl.key/server.key"

SSLCertificateChainFile "${SRVROOT}/conf/scs1731818806597_visionary.net.cn_ca.crt"
```

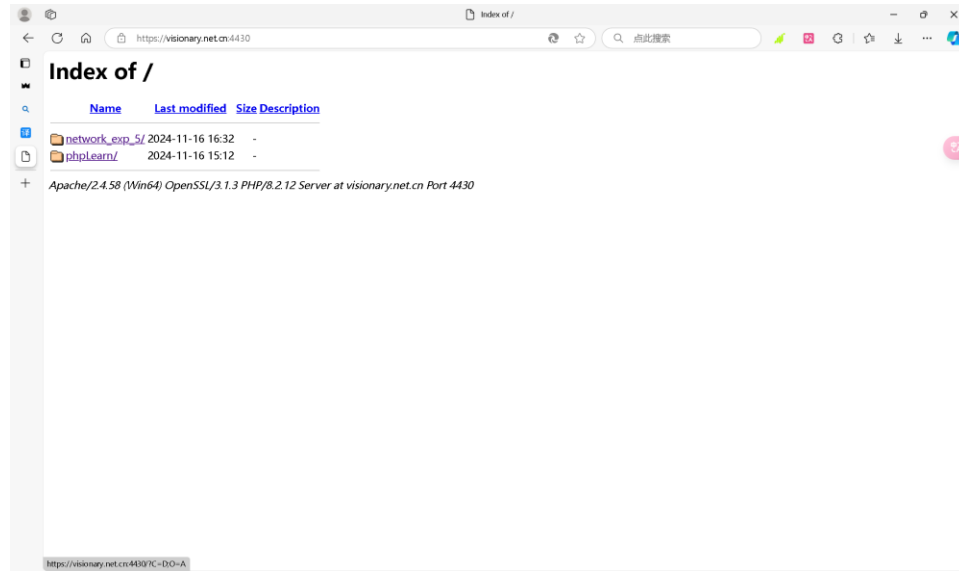
(httpd-vhost.conf)

```
<VirtualHost *:4430>
    DocumentRoot "E:/code/webroot"
    ## 文件存放根目录
    ServerName visionary.net.cn:4430
    ## 服务器域名
    SSLEngine on
    SSLCertificateFile "D:/others/xampp/apache/conf/ssl.crt/scs1731818806597_visionary.net.cn_server.crt"
    SSLCertificateKeyFile "D:/others/xampp/apache/conf/ssl.key/scs1731818806597_visionary.net.cn_server.key"
</VirtualHost>
```

另外，在 hosts 文件中配置了域名指向的 ip 地址（本机）

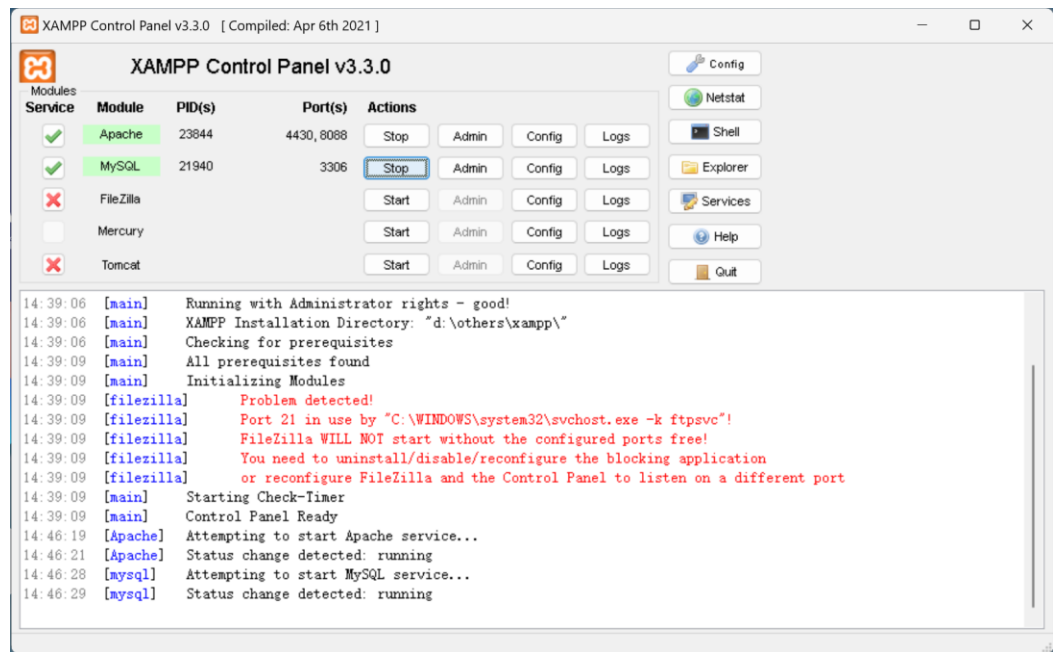
```
127.0.0.1 visionary.net.cn
```

效果



2. 数据库配置

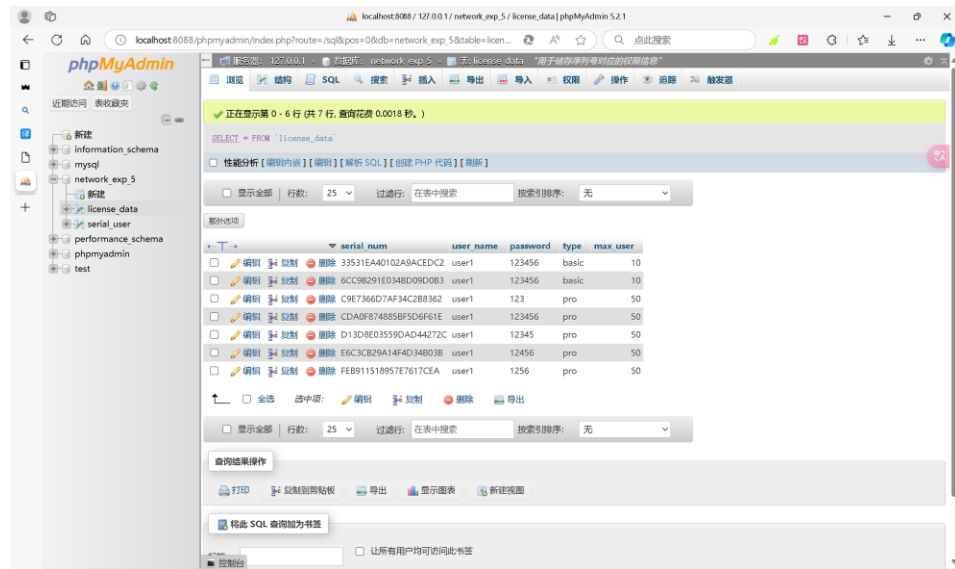
数据库平台也使用 xampp，具体是 mysql 数据库



因为本机没有安装过 mysql，所以端口没有进行修改，是默认的 3306

本地服务器 MainPort 设为 8088，通过这个端口访问数据库软件

效果



二、 接口配置

这里将三个软件分别放到一个类中，通过 Program 脚本通过不同的指令进入不同的软件

```
using E5_3790;

if (args[0] == "server")
{
    var server = new LicenseServer();
    server.Start();
}

if (args[0] == "client")
{
    var client = new LicenseClient();
    client.Start();
}

if (args[0] == "license")
{
    var manager = new LicenseManager();
    manager.Start();
}
```

1. 服务器端

```

namespace E5_3790;

// 1 用法
class LicenseServer
{
    private const int TcpPort = 12345;
    private const int HttpPort = 4430;
    private Dictionary<string, LicenseData> _licenseUsage = new ();

    private HttpClient _client;

    // 启动服务器
    // 1 用法
    public void Start() { ... }

    // 对tcp报文进行处理
    // 1 用法
    private void HandleClient(object? obj) { ... }

    // 等5秒，如果超时就认定程序A崩溃
    // 2 用法
    private async Task WaitForTimeout(Cancellation token, TcpClient tcpClient) { ... }

    // 1 用法
    private async Task<string> HandleRegisterRequest(string licenseString) { ... }

    // 1 用法
    private async Task<string> HandleSeekRequest() { ... }

    // 1 用法
    private async Task<string> HandleVerifyRequest(string serialNumber) { ... }
}

```

2. 程序 A（用户端）

```

class LicenseClient
{
    private const string ServerAddress = "127.0.0.1"; // 服务器地址
    private const int ServerPort = 12345; // 服务器端口
    private TcpClient? _client;
    private Timer? _heartbeatTimer;

    // 启动程序A
    // 1 用法
    public void Start() { ... }

    // 程序A运行
    // 2 用法
    private void Run() { ... }

    // 连接到服务器
    // 2 用法
    private void ConnectToServer() { ... }

    // 发送心跳信息，确认程序存活
    // 1 用法
    private void SendHeartbeat(object? obj) { ... }

    // 封装对发送tcp报文以及接收对应的服务器的信息
    // 2 用法
    private KeyValuePair<int, string> Communicate(NetworkStream stream, string content) { ... }

    // 关闭和服务器的tcp连接
    // 2 用法
    private void Close(TcpClient client, NetworkStream stream) { ... }
}

```

3. 许可证程序


```

[Serializable]
图3 用法
public class LicenseData{...}

图1 用法
public class LicenseManager
{
    private const string ServerAddress = "127.0.0.1"; // 服务器地址
    private const int ServerPort = 12345; // 服务器端口
    private TcpClient? _client;
    private Timer? _heartbeatTimer;
    // 用于生成许可证的密钥
    private const string SecretKey = "WQC200437";
    private readonly Dictionary<string, int> _typeToMaxUsers = new()
    {...};

    // 启动许可证程序
    图1 用法
    public void Start(){...}

    // 连接到服务器
    图1 用法
    private void ConnectToServer(){...}

    // 发送心跳信息，确认程序存活
    图1 用法
    private void SendHeartbeat(object? obj){...}

    // 封装对发送tcp报文以及接收对应的服务器的信息
    图1 用法
    private KeyValuePair<int, string> Communicate(NetworkStream stream, string content){...}

    // 生成许可证序号
    图1 用法
    private LicenseData GenerateLicense(string[] inputString){...}

    // 关闭和服务器的tcp连接
    图1 用法
    private void Close(TcpClient client, NetworkStream stream){...}
}

```

4. PHP 网页接口

Seek.php

// 用于查找本机的 mac 地址是否在服务器中有记录，即是否有许可证资格

Register.php

// 用于注册新的许可证

Verify.php

// 用于验证激活用许可证是否存在，以及人数是否已满，如果存在且未滿，存入数据库

三、 核心代码展示

1. 服务器

建立连接

```
public void Start()
{
    _client = new HttpClient();

    var listener = new TcpListener(localaddr: IPAddress.Any, TcpPort);
    listener.Start();
    Console.WriteLine("License Server started...");

    while (true)
    {
        // 接受客户端连接
        var client = listener.AcceptTcpClient();
        ThreadPool.QueueUserWorkItem(HandleClient, client);
    }
}
```

获取请求

```
// 读取客户端请求
while (true)
{
    if (!tcpClient.Connected) break;

    int bytesRead;
    while (stream.CanRead && (bytesRead = stream.Read(buffer, offset:0, count:buffer.Length)) != 0)
    {
        var request:string[] = Encoding.UTF8.GetString(buffer, index:0, count:bytesRead).Split(' ');
        var response = "";

        cts.Cancel();
        cts.Dispose();

        Console.WriteLine($"收到请求: {request[0]}");
        switch (request[0]){...}

        Console.WriteLine(response);
        var responseBytes:byte[] = Encoding.UTF8.GetBytes(response);
        stream.Write(responseBytes, offset:0, count:responseBytes.Length);

        // 重新计时
        cts = new CancellationTokensource();
        timeoutTask = WaitForTimeout(cts.Token, tcpClient);
    }
}
```

处理请求（取一个作为样例）

```
private async Task<string> HandleRegisterRequest(string licenseString)
{
    var url:string = "https://visionary.net.cn:" + HttpPort + "/network_exp_5/api/register.php";
    var postData = new FormUrlEncodedContent(nameValueCollection:new[]
    {
        new KeyValuePair<string, string>("license", licenseString)
    });

    var response = await _client.PostAsync(url, postData);
    var responseString = await response.Content.ReadAsStringAsync();

    return responseString;
}
```

2. 程序 A

登录、激活过程

```

ConnectToServer();
var stream = _client.GetStream();

var (seekRes:int, seekInfo:string) = Communicate(stream, content:"seek");
Console.WriteLine(seekInfo);

// 没有注册过序列号
if (seekRes == 0)
{
    Console.WriteLine("请输入序列号进行激活:");
    //
    var serialNum:string? = Console.ReadLine();

    var (verifyRes:int, verifyInfo:string) = Communicate(stream, content:$"verify {serialNum}");
    Console.WriteLine(verifyInfo);

    if (verifyRes != 1)
    {
        Close(_client, stream);
        return;
    }
    else
    {
        Run();
    }
}
else
{
    Console.WriteLine("登录成功");
    Run();
}
}

```

运行期间，如果服务器关闭，进行重连

```

// 程序A运行
// 2.用法
private void Run()
{
    var threadStop = false;
    // 时刻检查是否掉线
    var thread = new Thread(start:() =>
    {
        while (true)
        {
            if (threadStop) return;

            if (_client == null || !_client.Connected)
            {
                ConnectToServer();
                Console.WriteLine("重连成功");
            }
        }
    });
    thread.Start();

    Console.WriteLine("应用运行中，q键退出");

    while (true)
    {
        // 其他工作

        var key = Console.ReadKey(intercept: true);

        if (key.Key == ConsoleKey.Q) { ... }
    }
}

```

封装通信和接收回复

```
// 封装对发送tcp报文以及接收对应的服务器的信息
// 2 用法
private KeyValuePair<int, string> Communicate(NetworkStream stream, string content)
{
    try
    {
        var buffer:byte[] = Encoding.UTF8.GetBytes(content);

        stream.Write(buffer, offset:0, count:buffer.Length);
        stream.Flush();

        // 接收授权结果
        buffer = new byte[256];
        int bytesRead = 0;
        while ((bytesRead = stream.Read(buffer, offset:0, count:buffer.Length)) > 0) {}
        var response:string = Encoding.UTF8.GetString(buffer, index:0, count:bytesRead);

        var segments:string[] = response.Split(' ');
        if (!int.TryParse(segments[0], out var result:int)) result = 0;
        var info:string = segments[1];

        return new KeyValuePair<int, string>(result, info);
    }
    catch (Exception e){...}
}
```

3. 许可证软件

产生许可证并验证

```
public void Start()
{
    ConnectToServer();

    var stream = client.GetStream();

    while (true)
    {
        Console.WriteLine("输入用户名、口令、许可证类型获取序列号");
        var input:string[]? = Console.ReadLine()?.Split(" ");
        if (input != null)
        {
            var licenseData = GenerateLicense(input);
            if (licenseData == null) continue;

            var jsonString = JsonSerializer.Serialize(licenseData);

            var (registerRes:int, registerInfo:string) = Communicate(stream, content:$"register {jsonString}");
            Console.WriteLine(registerInfo);

            if (registerRes == 1)
            {
                Console.WriteLine($"您的序列号为{licenseData.SerialNumber}, 可激活主机数上限为{licenseData.MaxUsers}人");
            }
        }

        Console.WriteLine("输入空格后继续, 输入q退出");
        var quit = false;
        while (true){...}

        if (quit) break;
    }
}
```

4. PHP 网页接口

Seek.php

```
<?php
include("db.php");
include 'mac.php';

$conn = db_connect();
if (!$conn) {
    echo "连接失败: " . mysqli_connect_error();
    exit;
}

// 获取mac地址
$mac_address = get_mac();
// 查找对应的序列号
$sql = "SELECT serial_num, user_mac FROM 'serial_user' WHERE user_mac = '$mac_address'";

$stmt = $conn->prepare($sql);
// $stmt->bind_param("s", $mac_address);
$stmt->execute();
$result = $stmt->get_result();

if ($result->num_rows > 0) {
    echo "1 验证完毕, 登陆成功";
} else {
    // 没找到, 是首次使用
    var_dump($mac_address);
    echo "0 没有使用记录";
}

$stmt->close();
$conn->close();
```

Register.php

```
register.php x mac.php verify.php seek.php
1 <?php
2 include 'db.php';
3
4
5 $conn = db_connect();
6 if (!$conn) {
7     echo "连接失败: " . mysqli_connect_error();
8     exit;
9 }
10
11
12 $license_data = $_POST['license'];
13 $license = json_decode($license_data);
14
15 // 我是否已有该序列号
16 $sql = "SELECT * FROM license_data WHERE serial_num = ?";
17 $stmt = $conn->prepare($sql);
18 $stmt->bind_param("s", $license->SerialNumber); // 假设 serial_num 是字符串类型
19 $stmt->execute();
20 $result = $stmt->get_result();
21
22 // 插入序列号
23 if (mysqli_num_rows($result) == 0) {
24     $insert_sql = "INSERT INTO license_data (serial_num, user_name, password, type, max_user) VALUES (?, ?, ?, ?, ?)";
25     $insert_stmt = $conn->prepare($insert_sql);
26     $insert_stmt->bind_param("sssss", $license->SerialNumber, $license->UserName, $license->Password, $license->Type, $license->MaxUsers);
27
28     if ($insert_stmt->execute()) {
29         echo ("1 许可证申请通过");
30     }
31     else {
32         echo ("0 序列号存储失败");
33     }
34
35     $insert_stmt->close();
36 }
37 else {
38     echo ("0 许可证序列号已存在");
39 }
40
41 $stmt->close();
42 $conn->close();
43 ?>
```

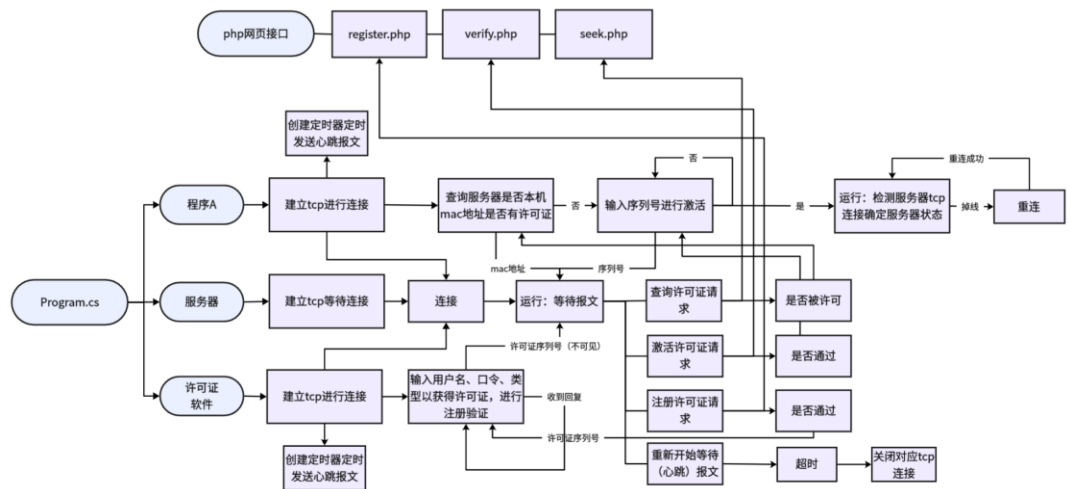
Verify.php

```

1  <?php
2  include 'db.php';
3
4
5  $conn = db_connect();
6  if (!$conn) {
7      echo "连接失败: " . mysqli_connect_error();
8      exit;
9  }
10
11
12  $license_data = $_POST['license'];
13  $license = json_decode($license_data);
14
15  // 我是否有该序列号
16  $sql = "SELECT * FROM license_data WHERE serial_num = ?";
17  $stmt = $conn->prepare($sql);
18  $stmt->bind_param("s", $license->SerialNumber); // 假设 serial_num 是字符串类型
19  $stmt->execute();
20  $result = $stmt->get_result();
21
22  // 插入序列号
23  if (mysqli_num_rows($result) == 0) {
24      $insert_sql = "INSERT INTO license_data (serial_num, user_name, password, type, max_user) VALUES (?, ?, ?, ?, ?)";
25      $insert_stmt = $conn->prepare($insert_sql);
26      $insert_stmt->bind_param("sssss", $license->SerialNumber, $license->UserName, $license->Password, $license->Type, $license->MaxUsers);
27
28      if ($insert_stmt->execute()) {
29          echo ("1 许可证申请通过");
30      }
31      else {
32          echo ("0 序列号存储失败");
33      }
34
35      $insert_stmt->close();
36  }
37  else {
38      echo ("0 许可证序列号已存在");
39  }
40
41  $stmt->close();
42  $conn->close();
43  ?>

```

四、程序流程图



五、运行展示

(此处演示中数据库数据初始为空)

1. 启动服务器

```
Windows PowerShell
版权所有 (C) Microsoft Corporation. 保留所有权利。

安装最新的 PowerShell, 了解新功能和改进! https://aka.ms/PSWindows

PS E:\code\practice\Computer_Network\Experiment\cni-exp\E5_3790\bin\net8.0> ./E5_3790 server
License Server started...
```

2. 启动软件 A, 没有记录, 为第一次使用, 需要输入序列号

服务器端收到 seek 请求调用 php 接口, 此外还有应用 A 的 4 秒一次的心跳报文

```
Windows PowerShell
版权所有 (C) Microsoft Corporation. 保留所有权利。

安装最新的 PowerShell, 了解新功能和改进! https://aka.ms/PSWindows

PS E:\code\practice\Computer_Network\Experiment\cni-exp\E5_3790\bin\net8.0> ./E5_3790 client
没有使用记录
请输入序列号进行激活:
```

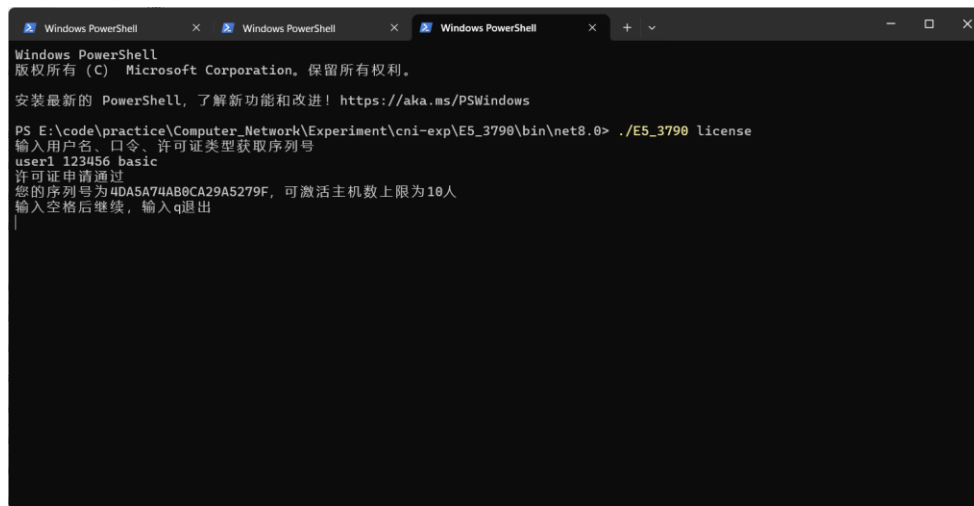
```
Windows PowerShell
版权所有 (C) Microsoft Corporation. 保留所有权利。

安装最新的 PowerShell, 了解新功能和改进! https://aka.ms/PSWindows

PS E:\code\practice\Computer_Network\Experiment\cni-exp\E5_3790\bin\net8.0> ./E5_3790 server
License Server started...
收到请求: seek
0 没有使用记录
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
```

3. 启动许可证程序，输入信息后，获得许可证序列号

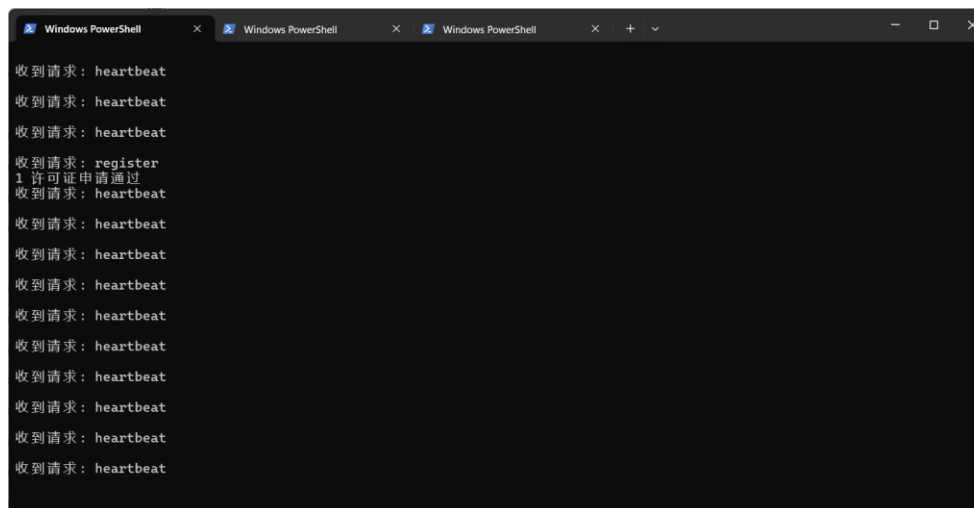
服务器这里收到了 register 请求，调用 php 接口，然后收到了申请通过的信息



```
Windows PowerShell
版权所有 (C) Microsoft Corporation。保留所有权利。

安装最新的 PowerShell，了解新功能和改进！https://aka.ms/PSWindows

PS E:\code\practice\Computer_Network\Experiment\cni-exp\E5_3790\bin\net8.0> ./E5_3790 license
输入用户名、口令、许可证类型获取序列号
user1 123456 basic
许可证申请通过
您的序列号为 4DA5A74AB8CA29A5279F，可激活主机数上限为 10 人
输入空格后继续，输入 q 退出
```



```
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: register
1 许可证申请通过
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
```

4. 这里测试使用不存在的许可证序列号，发现不能通过，程序退出

相对应的，服务器 5 秒内每收到该程序的心跳报文，认定为崩溃，所以断开 TCP 连接


```
Windows PowerShell
版权所有 (C) Microsoft Corporation。保留所有权利。

安装最新的 PowerShell, 了解新功能和改进! https://aka.ms/PSWindows

PS E:\code\practice\Computer_Network\Experiment\cni-exp\E5_3790\bin\net8.0> ./E5_3790 client
没有使用记录
请输入序列号进行激活:
111
序列号不存在
程序将在2秒后关闭
PS E:\code\practice\Computer_Network\Experiment\cni-exp\E5_3790\bin\net8.0>
```

```
Windows PowerShell
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: verify
0 序列号不存在
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
5秒内未收到心跳, 断开连接
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: seek
收到请求: heartbeat
0 没有使用记录
收到请求: heartbeat
```

5. 重新打开软件 A, 输入存在的序列号发现可以运行 (这里不小心在服务器终端按了 CTRL+C, 结果退出去了, 所以报了点错, 正好把之后的一块做了)

同时服务器也收到 verify 请求, 调用对应 php 接口

[illegible]

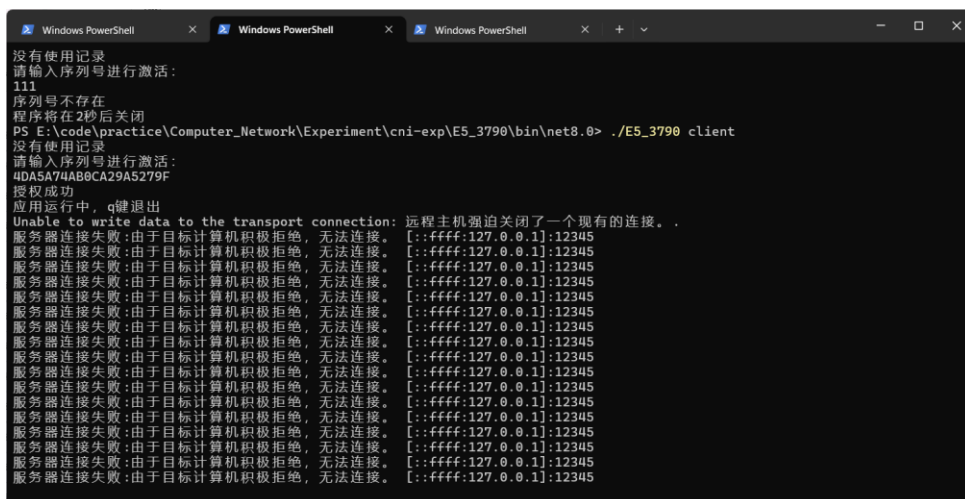
The screenshot shows a Windows PowerShell terminal window with the following output:

```
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: verify
1 授权成功
收到请求: heartbeat
收到请求: heartbeat
```

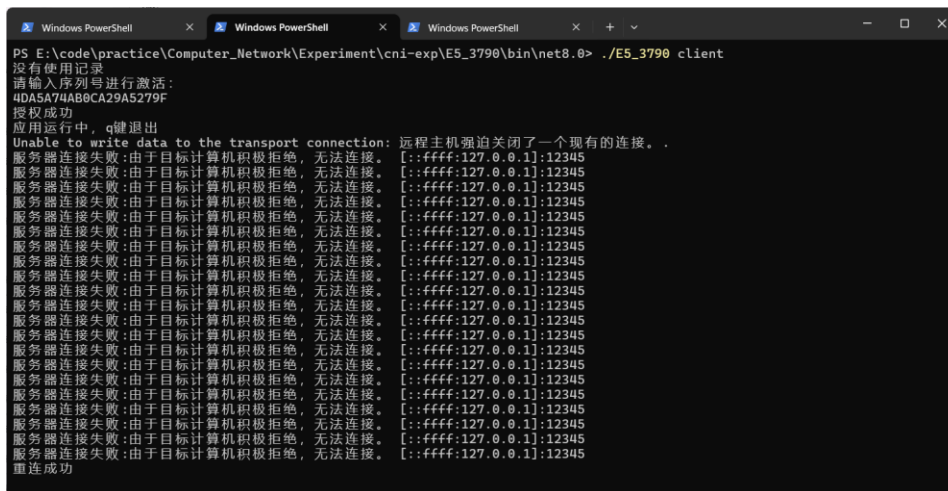
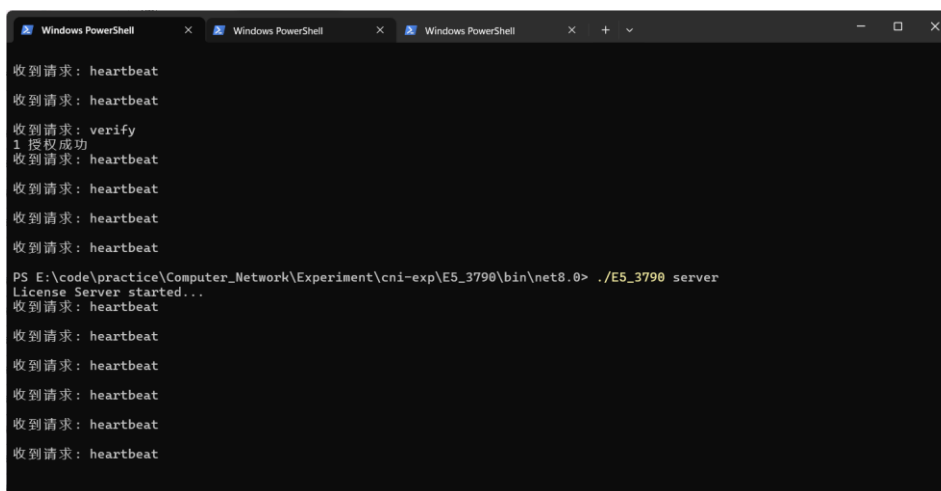
6. 服务器意外断开后，软件 A 会尝试进行重连

The screenshot shows a Windows PowerShell terminal window with the following output:

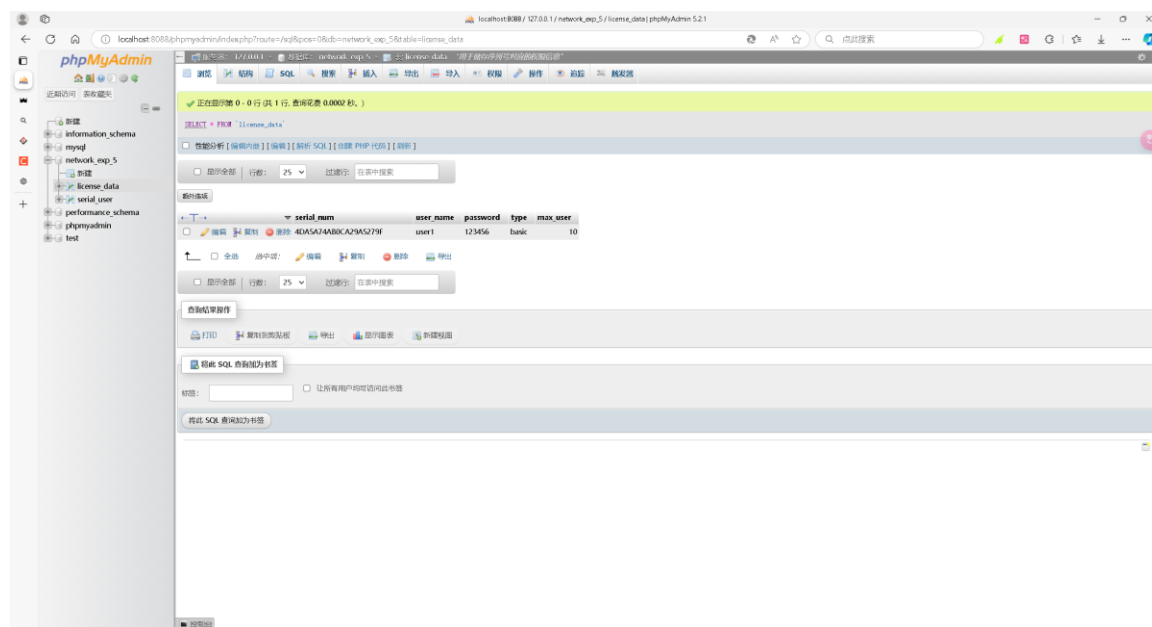
```
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: verify
1 授权成功
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
收到请求: heartbeat
PS E:\code\practice\Computer_Network\Experiment\cni-exp\E5_3790\bin\net8.0>
```

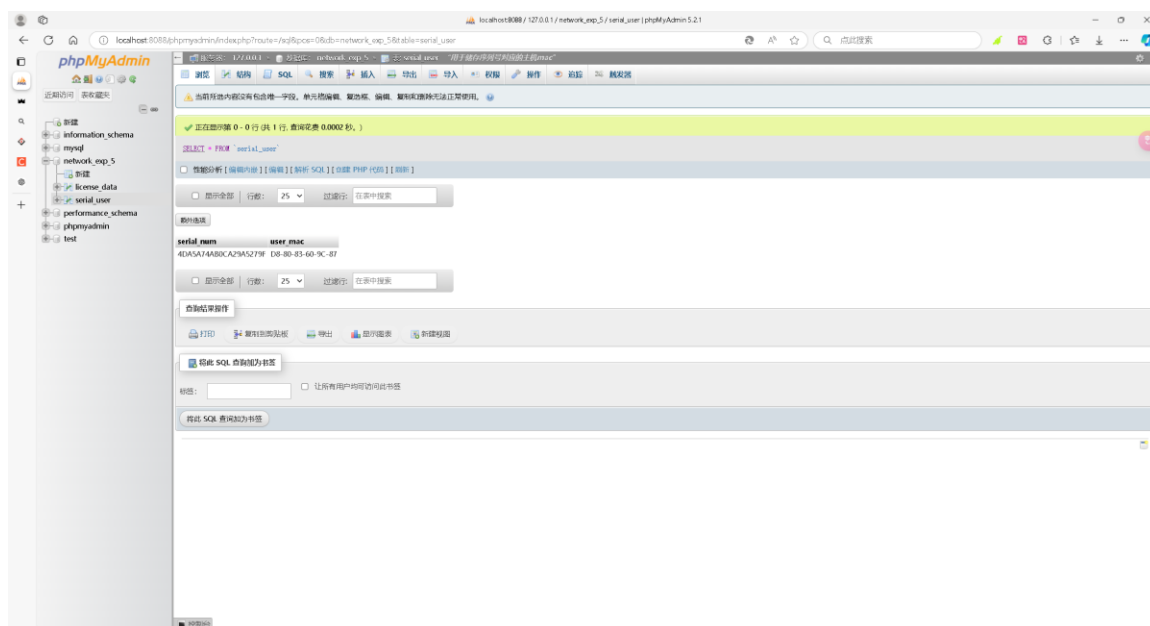


7. 将服务器重新启动后，发现软件 A 重连成功，服务器可以收到新的 heartbeat 报文



9. 数据库中也能查找到相对应的数据





4 实验代码

本次实验的代码已上传于以下代码仓库：[CNI-Exp: 厦门大学计算机网络课程实验项目集 \(gitee.com\)](#)

5 课后思考题

无

6 实验总结

本次实验实验目的是应用层文件传输的原理，但实际上过程中也夹杂了很多实用相关的知识。

首先，本次实验通过 `socket` 接口使用 `tcp` 报文实现服务器端和用户端的通信。

其次，本次实验还搭建了本地服务器以及搭载 `ssl` 证书以便通过 `https` 进行 `php` 文件接口的调用。

然后，还通过 `php` 文件实现了对数据库的交互。

总结来说，是收获颇丰。