New issue

Jump to bottom

Vulnerability Report： BinaryFormatter security vulnerability #1

Open ykytutou opened this issue on Dec 8, 2021 · 1 comment

ykytutou commented on Dec 8, 2021

Risk Class： SinGooCMSUtility/SinGooCMS.Utility/Net/SocketClient.cs

```
/// <summary>
/// 接收T类对象,反序列化
/// </summary>
/// <typeparam name="T">接收T类对象,T类必须是一个可序列化类</typeparam>
/// <param name="socket">客户端socket</param>
/// <returns>强类型对象</returns>
public static T ReceiveVarData<T>(this Socket socket)
{
    //先接收包头长度 固定8个字节
    byte[] lengthbyte = ReceiveFixData(socket, 8);
    //得到字节长度
    int length = GetPacketLength(lengthbyte);
    byte[] bytecoll = new byte[m_maxpacket];
    IFormatter format = new BinaryFormatter();
    MemoryStream stream = new MemoryStream();
    int offset = 0; //接收字节个数
    int lastdata = length; //还剩下多少没有接收,初始大小等于实际大小
    int receivedata = m_maxpacket; //每次接收大小
    //循环接收
    int mark = 0; //标记几次接收到的数据为0长度
    while (true)
    {
        //剩下的字节数是否小于缓存大小
        if (lastdata < m_maxpacket)
        {
            receivedata = lastdata; //就只接收剩下的字节数
        }

        int count = socket.Receive(bytecoll, 0, receivedata, 0);
        if (count > 0)
        {
            stream.Write(bytecoll, 0, count);
            offset += count;
            lastdata -= count;
            mark = 0;
        }
        else
        {
            mark++;
            if (mark == 10)
            {
                break;
            }
        }

        if (offset == length)
        {
            break;
        }
```

```
    }

    stream.Seek(0, SeekOrigin.Begin); //必须要这个 或者stream.Position = 0;
    T t = (T)format.Deserialize(stream);
    return t;
}
```

Set up socket communication

server :

```
```

client :

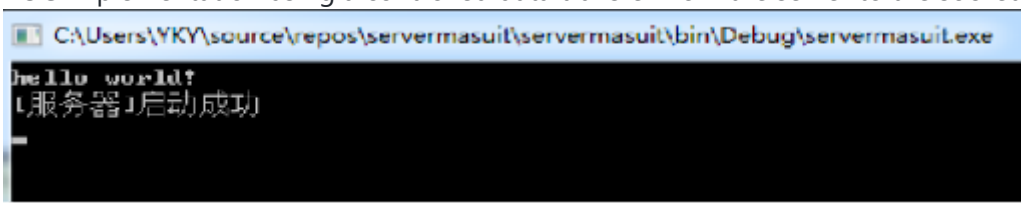
```
class Program
{
    static void Main(string[] args)
    {
        //const int BUFFER_SIZE = 3296;
        const int BUFFER_SIZE = 3118;
        byte[] readBuff = new byte[BUFFER_SIZE];

        //Socket
        Socket socket = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);
        //Connect
        string host = "127.0.0.1";
        int port = 1234;
        socket.Connect(host, port);
        //Recv
        //socket.Receive(readBuff);
        //Console.WriteLine(1);
        //Console.WriteLine(BitConverter.ToString(readBuff));
        //Console.WriteLine(2);

        //string str = System.Text.Encoding.UTF8.GetString(readBuff);
        //Console.WriteLine(str);
        SocketClient.ReceiveVarData<TestClass>(socket);

        ////本地序列化测试
        //Stream stream = new MemoryStream(readBuff);
        //stream.Position = 0;
        //IFormatter format = new BinaryFormatter();
        //var t = (TestClass)format.Deserialize(stream);


        //Close
        socket.Close();


        Console.ReadLine();
    }
}
```

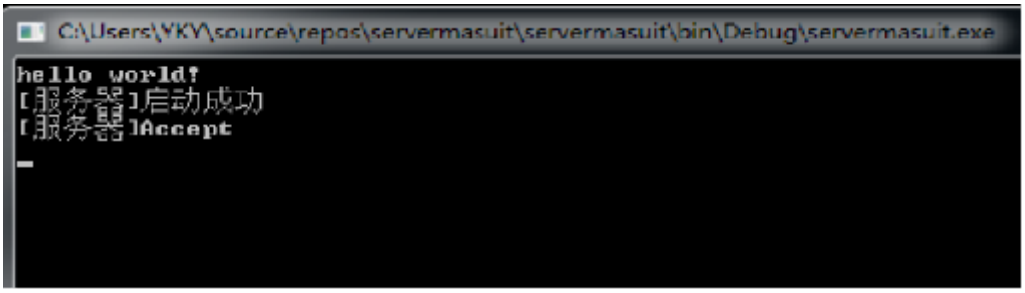
Constructing the payload

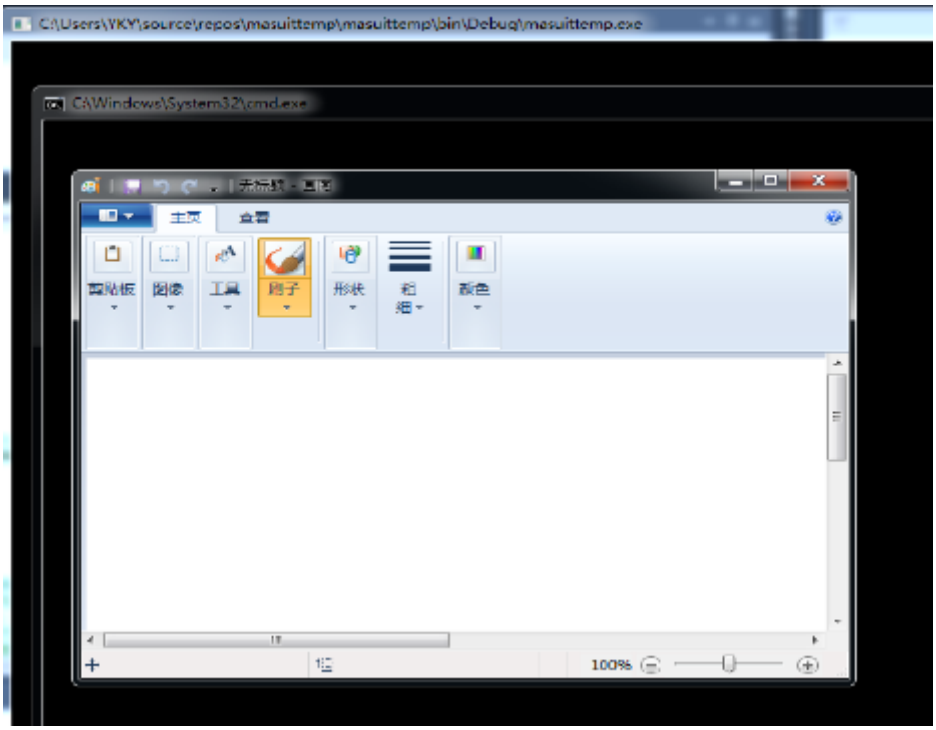
The ReceiveVarData() method internally first calls the ReceiveFixData() method to read the packet header (8 bytes) of the socket object information, and then calls the GetPacketLength() method to read the length of the bytes in the packet header (int type)
The ReceiveFixData() method will first intercept 8 bytes of information, so 8 bytes must be added before the original payload when constructing the POC.
The GetPacketLength() method reads the packet header information, i.e. the 8 bytes of information needs to contain the byte length (int type) of the original payload, while intercepting the int data type before the ' * ' ending.
Also the Payload source code is converted to a byte array length of 3110 bytes.
A local test of the GetPacketLength() method shows that the 8 bytes of information could be "3110 ****".
Int(3110) is the byte length of the original payload.

Simulating the transmission of messages to a socket client

POC implementation using a controlled data transfer from the server to the socket client, i.e. a set payload.







ykytutou commented on Dec 8, 2021 — Author

The payload here：https://github.com/ykytutou/Payload

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

---

1 participant