

Forum:[Hack Forums](#)

Thread: [Pictures Not Sending Fully Over TCP](#)

true - January 26, 2017, 7:16 am

I have a server & client model. The client is supposed to take a screenshot, and then send it to the server. The problem I'm having, is part of the screenshot is missing. What I mean by this is like 3/4 of the screen is black when opened in Paint or another app. When I send the screenshot command a second time, the file doesn't open at all, it's corrupt.

Here is my client side

Code if (plainText.Contains("screenshot"))

```
{
    Bitmap bitmap = new Bitmap(Screen.PrimaryScreen.Bounds.Width,
Screen.PrimaryScreen.Bounds.Height);
    Graphics graphics = Graphics.FromImage(bitmap);
    graphics.CopyFromScreen(0, 0, 0, 0, bitmap.Size);
    bitmap.Save("test.bmp");
    writebuffer = File.ReadAllBytes("test.bmp");
    stream.Write(writebuffer, 0, writebuffer.Length);
}As you can see, it takes a screen shot, saves the image to a bitmap file, then reads the bytes
into a buffer and sends it.
```

Here is my server side

Code foreach (user c in clientList)

```
{
    if (c.RemoteIPAddress == currentRow)
    {
        NetworkStream stream = c.clientTCP.GetStream();
        writeBuffer = Encoding.ASCII.GetBytes("screenshot");
        stream.Write(writeBuffer, 0, writeBuffer.Length);

        byte[] readBuffer = new byte[0];
        readBuffer = new byte[c.clientTCP.ReceiveBufferSize];
        int data = stream.Read(readBuffer, 0, readBuffer.Length);

        string x = new Random().Next().ToString();

        FileStream f = new FileStream(new Random().Next() + ".bmp", FileMode.Create,
FileAccess.Write);
        f.Write(readBuffer, 0, data);
        f.Close();
        Process.Start(x + ".bmp");
    }
}Here, I send the command to tell the client to send a screenshot, then receive the screen, and
then write the said screenshot to a bitmap file.
```

I cannot seem to figure out what is causing the aforementioned issues in this code.

zeroexp - January 26, 2017, 8:42 am

Don't have time to test but try to debug and check if `writebuffer.Length` in the client is equal to the `readBuffer.Length` on the server.

Zilog80 - January 26, 2017, 11:20 am

Imagine the bitmap has 100000 Bytes.

The problem is that in your client, you are assuming that you received 100000 bytes from the server, and you're putting all 100000 bytes into the memory stream.

There is a great sample on the MSDN page for `TcpClient` which shows receiving from a `TcpClient` by using the underlying `NetworkStream`. Also, you will want to keep track of how many bytes you actually received (this is the return value from the `NetworkStream.Read` function). Finally, you'll want to keep reading until there is no more data to be read from the host. If your image is larger than your buffer, then you'll only have a partial image also. The sample on the linked page for `NetworkStream.Read` shows continually reading from the stream while there is data available.

Source: <http://stackoverflow.com/questions/6811760/sending-and-receiving-an-image-file-c-sharp>

TL;DR: what @Wops said

Zilog80 - January 26, 2017, 2:09 pm

(Jan 26, 2017 03:00 PM)Submerged™ Wrote: [\[CITE\]](#) So what you're saying is that my problem lies on the receiving end, not in the sender?

In your 'server' (where you are 'receiving' the bitmap), put a breakpoint in this line:

PHP Code

```
<?php
```

```
FileStream f = new FileStream(new Random().Next() + ".bmp", FileMode.Create, FileAccess.Write);
```

Launch your client and 'send' a bitmap.

Your server will be stop at the breakpoint and check the value for the `readBuffer` lenght.

You will see that the 'length' you received is < than bytes lenght your client send.

Hope it helps.

Quote:Hm, I just checked. The buffer lengths are different. One of them is set to 65536, which is one more than the max amount of bytes in a packet.

Check the links to MSDN from the SO thread I posted above and you will fix it ?

Zilog80 - January 26, 2017, 3:20 pm

No idea how your code works, but maybe this would help:

GrimEdittion TCP Library Framework: <https://hackforums.net/showthread.php?tid=5527262>

BahNahNah SyncIO: <https://hackforums.net/showthread.php?tid=5506453>

BahNahNah - January 26, 2017, 5:25 pm

Its most likely because its breaking up your packet.

If you want to receive a single packet, use UDP (but look up the differences between the TCP/UDP

because it probably dosent fit what you want to do)

Its because TCP is a stream protocol, so the data comes trough like water in a pipe (If the pipe is a bit dodgy, you are not going to get all the water you want as fast as a new pipe).

Where as UDP is like sending a postage box.

Here is my approach of receving full packets through TCP even if they are (bigger than the buffer)

<https://github.com/BahNahNah/SyncIO/blob/master/SyncIO/Transport/PacketDefragmenter.cs>

Here are the tests too:

<https://github.com/BahNahNah/SyncIO/blob/master/SyncIO/Tests/Transport/PacketDefragmenterTests.cs>

Ninja - January 26, 2017, 9:06 pm

(Jan 26, 2017 06:25 PM)BahNahNah Wrote: [\[CITE\]](#) Its most likely because its breaking up your packet.

If you want to receive a single packet, use UDP (but look up the differences between the TCP/UDP because it probably dosent fit what you want to do)

Its because TCP is a stream protocol, so the data comes trough like water in a pipe (If the pipe is a bit dodgy, you are not going to get all the water you want as fast as a new pipe).

Where as UDP is like sending a postage box.

Here is my approach of receving full packets through TCP even if they are (bigger than the buffer)

<https://github.com/BahNahNah/SyncIO/blob/master/SyncIO/Transport/PacketDefragmenter.cs>

Here are the tests too:

<https://github.com/BahNahNah/SyncIO/blob/master/SyncIO/Tests/Transport/PacketDefragmenterTests.cs>

<https://github.com/BahNahNah/SyncIO/blob/master/SyncIO/Tests/Transport/PacketDefragmenterTests.cs#L23>

This one typo is bugging me more than it should.