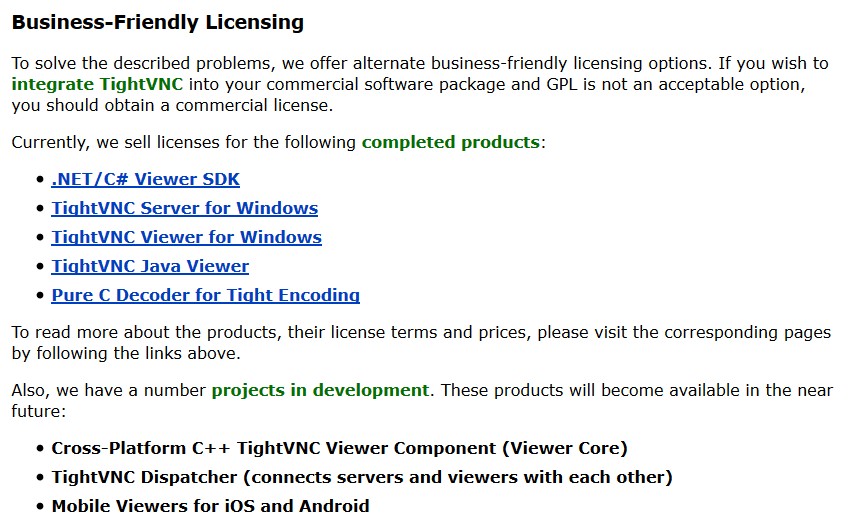
**远程桌面控制研究**

**一，介绍**

远程桌面原本是从Windows 2000 Server开始由微软公司提供的，它的功能是当某台计算机开启了远程桌面服务后我们就可以在网络的另一端控制这台计算机了，通过远程桌面功能我们可以实时的操作这台计算机，在上面安装软件，运行程序，所有的一切都好像是直接在该计算机上操作一样。这就是远程桌面的最大功能。

**二，技术方案**

# 远程桌面控制来源于微软，它可以有多种实现方式，网络上常见的一种就是Virtual Network Computing，也就是通常所说的VNC，VNC是采用RFB(remote frame buffer)协议图形化的桌面共享系统,它可以远程控制其他电脑。VNC在网络上有很多开源实现，比如RealVNC，TightVNC。二者都是开源的。其中TightVNC还提供远程桌面客户端(Viewer)的SDK,遗憾的是目前只提供C#的，还有一个Java的Viewer，遗憾的不是SDK，而是整个客户端Viewer。



**三，VNC基本组成架构**

VNC server-------共享被控机器的屏幕，以被动方式受VNC客户端控制

VNC client(viewer)-------可以观看服务端共享的屏幕，控制服务端

VNC 协议-------也是RFB协议，没有采用该协议的远控就不叫VNC。该协议的目的非常简单，就是一帧帧把像素矩阵（坐标系统是X,Y）和事件消息(鼠标消息，键盘消息)从服务端传送到客户端。

**四，基本原理**

VNC Server 运行在被控方的机器不需要物理显示器，一个默认的方式就是，客户端Viewer连接到Server端的端口上（默认5900），当然，浏览器其实也可以连接到Server端（这必须看实现，默认端口5800）。最后，Server也能以“侦听模式”连接客户端上的5500端口，这样的一个优势就是，Server端不需要配置防火墙就能允许客户端连接5900或5800端口，对于客户端来说，服务端配置的人员就可以不需要懂这些知识点，更多的是客户端的操作人员需要懂。

从远程屏幕帧数据流向来讲，Server端是把一帧的frame buffer分解成多个块矩阵发送给客户端，RFB协议可以使用多种带宽，所以这多种的方法，就是为了降低server端和client端的过多的通信核交成本。比如，RTB协议有多种编码类型（为了更加高效的传输这些多个矩阵块），RFB协议允许客户端和Server端在开始传输之前协商好即将使用的编码类型。最简单的编码类型所有客户端和Server端都支持，这种编码类型是将像素数据从左到友按照扫描行(scanline)顺序发送，等待整屏幕的像素数据传送完成时，之后就仅仅只发送屏幕中发生变化的像素部分了。但是这种编码类型有一个限制就是，仅仅只能在屏幕不大幅度更新像素的时候工作良好，一旦屏幕像素发生大幅度更新，所占用的带宽就会很大。（鼠标指针移动，打字就是小幅度的更新，但是看电影，滚动屏幕就是大幅度更新）

**五，RFB协议的限制**

1. 远程控制的粘贴板不支持复制粘贴Unicode文本，不能传送任何除Latin-1 character set以外的字符集编码。

2. 因为是基于像素传送的协议，所以从效率上来说，就没有那些采用了更加底层的图形系统的（Linux下的X11或windows下的RDP，RDP协议是Windows自带的远程桌面控制采用的协议）解决方案更高效。

3. 不是为安全设计的协议，传输密码有被嗅探到的可能

**六，参考资料**

VNC Html 5客户端（Web Sockets，Canvas）： <https://github.com/novnc/noVNC>

Libvncserver/client : https://github.com/LibVNC/libvncserver