Virtual Network Computing

In computing, Virtual Network Computing (VNC) is a graphical desktop sharing system that uses the RFB protocol to remotely control another computer. It transmits the keyboard and mouse events from one computer to another, relaying the graphical screen updates back in the other direction, over a network.

VNC is platform-independent – a VNC viewer on one operating system may connect to a VNC server on the same or any other operating system. There are clients and servers for many GUI-based operating systems and for Java. Multiple clients may connect to a VNC server at the same time. Popular uses for this technology include remote technical support and accessing files on one's work computer from one's home computer, or vice versa.

VNC was originally developed at the Olivetti Research Laboratory in Cambridge, United Kingdom. The original VNC source code and many modern derivatives are open source under the GNU General Public License.

VNC in KDE.

There are a number of variants of VNC[1] which offer their own particular functionality; e.g., some optimised for Microsoft Windows, or offering file transfer (not part of VNC proper), etc. Many are compatible (without their added features) with VNC proper in the sense that a viewer of one flavour can connect with a server of another; others are based on VNC code but not compatible with standard VNC.

VNC and RFB are registered trademarks of RealVNC Ltd. in the U.S. and in other countries.

Etymology

The name 'Virtual Network Computer/Computing' originates from ORL's work on a thin client called the Videotile which also used the RFB protocol. This was essentially an LCD with a pen input and a fast ATM connection to the network. At the time, network computer was commonly used as a synonym for 'thin client'. VNC is essentially a software-only (i.e. virtual) version of this network computer.

Operation

A VNC system consists of a client, a server, and a communication protocol.

* The VNC server is the program on the machine that shares its screen. The server passively allows the client to take control of it.
* The VNC client (or viewer) is the program that watches, controls, and interacts with the server. The client controls the server.
* The VNC protocol (RFB) is very simple, based on one graphic primitive from server to client ("Put a rectangle of pixel data at the specified X,Y position") and event messages from client to server.

In the normal method of operation a viewer connects to a port on the server (default port 5900). Alternatively a browser can connect to the server (default port 5800). And a server can connect to a viewer in "listening mode" on port 5500. One advantage of listening mode is that the server site
does not have to configure its firewall to allow access on port 5900 (or 5800); the onus is on the viewer, which is useful if the server site has no computer expertise, while the viewer user would be expected to be more knowledgeable.

The server sends small rectangles of the framebuffer to the client. In its simplest form, the VNC protocol can use a lot of bandwidth, so various methods have been devised to reduce the communication overhead. For example, there are various encodings (methods to determine the most efficient way to transfer these rectangles). The VNC protocol allows the client and server to negotiate which encoding will be used. The simplest encoding, which is supported by all clients and servers, is the raw encoding where pixel data is sent in left-to-right scanline order, and after the original full screen has been transmitted, only transfers rectangles that change. This encoding works very well if only a small portion of the screen changes from one frame to the next (like a mouse pointer moving across a desktop, or text being written at the cursor), but bandwidth demands get very high if a lot of pixels change at the same time, such as when scrolling a window or viewing full-screen video.

VNC by default uses TCP port 5900+N,[4][5] where N is the display number (usually :0 for a physical display). Several implementations also start a basic HTTP server on port 5800+N to provide a VNC viewer as a Java applet, allowing easy connection through any Java-enabled web browser. Different port assignments can be used as long as both client and server are configured accordingly.

Using VNC over the Internet works well if the user has a broadband connection at both ends. However, it may require advanced NAT, firewall and router configuration such as port forwarding in order for the connection to go through. Some users may choose to use instant private networking applications such as Virtual Private Network (VPN) applications such as Hamachi to make usage over the Internet much easier. Alternatively, a VNC connection can be established as a LAN connection if VPN is utilized as a proxy.

Note that the machine the VNC server is running on does not need to have a physical display. Xvnc is the Unix VNC server, which is based on a standard X server. To applications Xvnc is an X "server" (ie displays client windows), and to remote VNC users it is a VNC server. Applications can display themselves on Xvnc as if it were a normal X display, but they will appear on any connected VNC viewers rather than on a physical screen.[6] Alternatively a machine (which may be a workstation or a network server) with screen, keyboard, and mouse can be set up to boot and run the VNC server as a service or daemon, then the screen, keyboard, and mouse can be removed and the machine stored in an out-of-the-way location.

In addition, the display that is served by VNC is not necessarily the same display seen by a user on the server. On Unix/Linux computers that support multiple simultaneous X11 sessions, VNC may be set to serve a particular existing X11 session, or to start one of its own. It is also possible to run multiple VNC sessions from the same computer. On Microsoft Windows the VNC session served is always the current user session.

VNC is commonly used as a cross-platform remote desktop system. For example, Apple Remote Desktop for Mac OS X (and more recently, "Back to My Mac" in 'Leopard' - Mac OS X 10.5) interoperates with VNC and will connect to a Linux user's current desktop if it is served with x11vnc, or to a separate X11 session if one is served with TightVNC. From Linux, TightVNC will connect to a Mac OS X session served by Apple Remote Desktop if the VNC option is enabled, or to a VNC server running on Microsoft Windows.

Security
By default, VNC is not a secure protocol. While passwords are not sent in plain-text (as in telnet), brute-force cracking could prove successful if both the encryption key and encoded password are sniffed from a network. For this reason it is recommended that a password of at least 8 characters be used. On the other hand, there is also an 8-character limit on some versions of VNC; if a password is sent exceeding 8 characters, the excess characters are removed and the truncated string is compared to the password.

However, VNC may be tunnelled over an SSH or VPN connection which would add an extra security layer with stronger encryption. SSH clients are available for all major platforms (and many smaller platforms as well); SSH tunnels can be created from UNIX clients, Microsoft Windows clients, Macintosh clients (including Mac OS X and System 7 and up) – and many others. There are freeware applications that create instant VPN tunnels between computers.

UltraVNC supports the use of an open-source encryption plugin which encrypts the entire VNC session including password authentication and data transfer. It also allows authentication to be performed based on NTLM and Active Directory user accounts. However, use of such encryption plugins make it incompatible with other VNC programs. RealVNC offers high-strength encryption as part of its commercial package. Workspot released AES encryption patches for VNC.

[edit] Limitations

Unicode is not supported in VNC versions 3.x and lower so it is impossible to transfer clipboard text outside the Latin-1 character set.

The VNC protocol is pixel based. Although this leads to great flexibility (i.e.- any type of desktop can be displayed), it is often less efficient than solutions that have a better understanding of the underlying graphic layout like X11. Those protocols send graphic primitives or high level commands in a simpler form (e.g., "open window"), whereas RFB just sends the raw pixel data.