NAME

vncviewer - an X viewer client for VNC

SYNOPSIS

vncviewer [*options*] [*host*][*:display*]
vncviewer [*options*] [*host*][*::port*]
vncviewer [*options*] *-listen* [*display*]
vncviewer *-help*

DESCRIPTION

vncviewer is an Xt-based client application for the VNC (Virtual Network Computing) system. It can connect to any VNC-compatible server such as **Xvnc** or WinVNC, allowing you to control desktop environment of a different machine.

You can use F8 to display a pop-up utility menu. Press F8 twice to pass single F8 to the remote side.

OPTIONS

-help
Prints a short usage notice to stderr.

-listen
Make the viewer listen on port 5500+*display* for reverse connections from a server. WinVNC supports reverse connections using the "Add New Client" menu option, or the -connect command line option. **Xvnc** requires the use of the helper program **vncconnect**.

-via *gateway*
Automatically create encrypted TCP tunnel to the *gateway* machine before connection, connect to the *host* through that tunnel (TightVNC-specific). By default, this option invokes SSH local port forwarding, assuming that SSH client binary can be accessed as /usr/bin/ssh. Note that when using the **-via** option, the host machine name should be specified as known to the gateway machine, e.g. "localhost" denotes the *gateway*, not the machine where vncviewer was launched. See the ENVIRONMENT section below for the information on configuring the **-via** option.

-shared
When connecting, specify that a shared connection is requested. In TightVNC, this is the default mode, allowing you to share the desktop with other clients already using it.

-noshared
When connecting, specify that the session may not be shared. This would either disconnect other connected clients or refuse your connection, depending on the server configuration.

-viewonly

Disable transfer of mouse and keyboard events from the client to the server.

-fullscreen

Start in full-screen mode. Please be aware that operating in full-screen mode may confuse X window managers. Typically, such conflicts cause incorrect handling of input focus or make the viewer window disappear mysteriously. See the grabKeyboard setting in the RESOURCES section below for a method to solve input focus problem.

-noraiseonbeep

By default, the viewer shows and raises its window on remote beep (bell) event. This option disables such behaviour (TightVNC-specific).

-passwd *passwd-file*

File from which to get the password (as generated by the **vncpasswd**(1) program). This option affects only the standard VNC authentication.

-encodings *encoding-list*

TightVNC supports several different compression methods to encode screen updates; this option specifies a set of them to use in order of preference. Encodings are specified separated with spaces, and must thus be enclosed in quotes if more than one is specified. Available encodings, in default order for a remote connection, are "copyrect tight hextile zlib corre rre raw". For a local connection (to the same machine), the default order to try is "raw copyrect tight hextile zlib corre rre". Raw encoding is always assumed as a last option if no other encoding can be used for some reason. For more information on encodings, see the section ENCODINGS below.

-bgr233

Always use the BGR233 format to encode pixel data. This reduces network traffic, but colors may be represented inaccurately. The bgr233 format is an 8-bit "true color" format, with 2 bits blue, 3 bits green, and 3 bits red.

-owncmap

Try to use a PseudoColor visual and a private colormap. This allows the VNC server to control the colormap.

-truecolour, **-truecolor**

Try to use a TrueColor visual.

-depth *depth*

On an X server which supports multiple TrueColor visuals of different depths, attempt to use the specified one (in bits per pixel); if successful, this depth will be requested from the VNC server.

-compresslevel *level*
Use specified compression *level* (0..9) for "tight" and "zlib" encodings (TightVNC-specific). Level 1 uses minimum of CPU time and achieves weak compression ratios, while level 9 offers best compression but is slow in terms of CPU time consumption on the server side. Use high levels with very slow network connections, and low levels when working over high-speed LANs. It's not recommended to use compression level 0, reasonable choices start from the level 1.

-quality *level*
Use the specified JPEG quality *level* (0..9) for the "tight" encoding (TightVNC-specific). Quality level 0 denotes bad image quality but very impressive compression ratios, while level 9 offers very good image quality at lower compression ratios. Note that the "tight" encoder uses JPEG to encode only those screen areas that look suitable for lossy compression, so quality level 0 does not always mean unacceptable image quality.

-nojpeg
Disable lossy JPEG compression in Tight encoding (TightVNC-specific). Disabling JPEG compression is not a good idea in typical cases, as that makes the Tight encoder less efficient. You might want to use this option if it's absolutely necessary to achieve perfect image quality (see also the **-quality** option).

-nocursorshape
Disable cursor shape updates, protocol extensions used to handle remote cursor movements locally on the client side (TightVNC-specific). Using cursor shape updates decreases delays with remote cursor movements, and can improve bandwidth usage dramatically.

-x11cursor
Use a real X11 cursor with X-style cursor shape updates, instead of drawing the remote cursor on the framebuffer. This option also disables the dot cursor, and disables cursor position updates in non-fullscreen mode.

-autopass
Read a plain-text password from stdin. This option affects only the standard VNC authentication.

ENCODINGS

The server supplies information in whatever format is desired by the client, in order to make the client as easy as possible to implement. If the client represents itself as able to use multiple formats, the server will choose one.

Pixel format refers to the representation of an individual pixel. The most common formats are 24 and 16 bit "true-color" values, and 8-bit "color map" representations, where an arbitrary map converts the color number to RGB values.

Encoding refers to how a rectangle of pixels are sent (all pixel information in VNC is sent as rectangles). All rectangles come with a header giving the location and size of the rectangle and an encoding type used by the data which follows. These types are listed below.

Raw

The raw encoding simply sends width*height pixel values. All clients are required to support this encoding type. Raw is also the fastest when the server and viewer are on the same machine, as the connection speed is essentially infinite and raw encoding minimizes processing time.

CopyRect

The Copy Rectangle encoding is efficient when something is being moved; the only data sent is the location of a rectangle from which data should be copied to the current location. Copyrect could also be used to efficiently transmit a repeated pattern.

RRE

The Rise-and-Run-length-Encoding is basically a 2D version of run-length encoding (RLE). In this encoding, a sequence of identical pixels are compressed to a single value and repeat count. In VNC, this is implemented with a background color, and then specifications of an arbitrary number of subrectangles and color for each. This is an efficient encoding for large blocks of constant color.

CoRRE

This is a minor variation on RRE, using a maximum of 255x255 pixel rectangles. This allows for single-byte values to be used, reducing packet size. This is in general more efficient, because the savings from sending 1-byte values generally outweighs the losses from the (relatively rare) cases where very large regions are painted the same color.

Hextile

Here, rectangles are split up in to 16x16 tiles, which are sent in a predetermined order. The data within the tiles is sent either raw or as a variant on RRE. Hextile encoding is usually the best choice for using in high-speed network environments (e.g. Ethernet local-area networks).

Zlib

Zlib is a very simple encoding that uses zlib library to compress raw pixel data. This encoding achieves good compression, but consumes a lot of CPU time. Support for this encoding is provided for compatibility with VNC servers that might not understand Tight encoding which is more efficient than Zlib in nearly all real-life situations.

Tight

Like Zlib encoding, Tight encoding uses zlib library to compress the pixel data, but it pre-processes data to maximize compression ratios, and to minimize CPU usage on compression. Also, JPEG compression may be used to encode color-rich screen areas (see the description of -quality and -nojpeg options above). Tight encoding is usually the best choice for low-bandwidth network environments (e.g. slow modem connections).

RESOURCES

X resources that **vncviewer** knows about, aside from the normal Xt resources, are as follows:

shareDesktop

Equivalent of **-shared**/**-noshared** options. Default true.

viewOnly

Equivalent of **-viewonly** option. Default false.

fullScreen

Equivalent of **-fullscreen** option. Default false.

grabKeyboard

Grab keyboard in full-screen mode. This can help to solve problems with losing keyboard focus. Default false.

raiseOnBeep

Equivalent of **-noraiseonbeep** option, when set to false. Default true.

passwordFile

Equivalent of **-passwd** option.

passwordDialog

Whether to use a dialog box to get the password (true) or get it from the tty (false). Irrelevant if **passwordFile** is set. Default false.

encodings

Equivalent of **-encodings** option.

compressLevel

Equivalent of **-compresslevel** option (TightVNC-specific).

qualityLevel

Equivalent of **-quality** option (TightVNC-specific).

enableJPEG
Equivalent of **-nojpeg** option, when set to false. Default true.

useRemoteCursor
Equivalent of **-nocursorshape** option, when set to false (TightVNC-specific). Default true.

useBGR233
Equivalent of **-bgr233** option. Default false.

nColours
When using BGR233, try to allocate this many "exact" colors from the BGR233 color cube. When using a shared colormap, setting this resource lower leaves more colors for other X clients. Irrelevant when using truecolor. Default is 256 (i.e. all of them).

useSharedColours
If the number of "exact" BGR233 colors successfully allocated is less than 256 then the rest are filled in using the "nearest" colors available. This resource says whether to only use the "exact" BGR233 colors for this purpose, or whether to use other clients' "shared" colors as well. Default true (i.e. use other clients' colors).

forceOwnCmap
Equivalent of **-owncmap** option. Default false.

forceTrueColour
Equivalent of **-truecolour** option. Default false.

requestedDepth
Equivalent of **-depth** option.

useSharedMemory
Use MIT shared memory extension if on the same machine as the X server. Default true.

wmDecorationWidth, wmDecorationHeight
The total width and height taken up by window manager decorations. This is used to calculate the maximum size of the VNC viewer window. Default is width 4, height 24.

bumpScrollTime, bumpScrollPixels
When in full screen mode and the VNC desktop is bigger than the X display, scrolling happens

whenever the mouse hits the edge of the screen. The maximum speed of scrolling is bumpScrollPixels pixels every bumpScrollTime milliseconds. The actual speed of scrolling will be slower than this, of course, depending on how fast your machine is. Default 20 pixels every 25 milliseconds.

popupButtonCount
The number of buttons in the popup window. See the README file for more information on how to customize the buttons.

debug
For debugging. Default false.

rawDelay, copyRectDelay
For debugging, see the README file for details. Default 0 (off).

ENVIRONMENT

When started with the **-via** option, vncviewer reads the **VNC_VIA_CMD** environment variable, expands patterns beginning with the "%" character, and executes result as a command assuming that it would create TCP tunnel that should be used for VNC connection. If not set, this environment variable defaults to "/usr/bin/ssh -f -L %L:%H:%R %G sleep 20".

The following patterns are recognized in the **VNC_VIA_CMD** (note that all the patterns %G, %H, %L and %R must be present in the command template):

%%
A literal "%";

%G
gateway host name;

%H
remote VNC host name, as known to the gateway;

%L local TCP port number;

%R remote TCP port number.

SEE ALSO

vncserver(1), **Xvnc**(1), **vncpasswd**(1), **vncconnect**(1), **ssh**(1)

AUTHORS

Original VNC was developed in AT&T Laboratories Cambridge. TightVNC additions were implemented by Constantin Kaplinsky. Many other people participated in development, testing and support.

Man page authors:
Marcus Brinkmann <Marcus.Brinkmann@ruhr-uni-bochum.de>,
Terran Melconian <terran@consistent.org>,
Tim Waugh <twaugh@redhat.com>,
Constantin Kaplinsky <const@tightvnc.com>