

#### SecureCRT®

This help manual documents the VanDyke Software® SecureCRT®, version 7.1 terminal emulation package. If you do not find the information you need here, or if you have any comments or suggestions, please visit the following web page:

http://www.vandyke.com/feedback.php

For the latest information on SecureCRT and VanDyke Software, check out our home page:

http://www.vandyke.com

Make sure you don't miss important software releases and updates. Subscribe to our product mailing lists to receive email notifications when new releases of VanDyke products are announced.

https://secure.vandyke.com/cgi-bin/subscribe.php?PID=3

For in-depth tips, calls for testers, and overviews of the latest beta and official releases, visit the VanDyke Software *What's New* page:

http://whatsnew.vandyke.com

Thank you! The SecureCRT Product Team





### New in SecureCRT 7.1

SecureCRT 7.1 brings tiled sessions to Mac and Linux versions, adds an option to set up dependent sessions for connecting with jump hosts, and extends file transfer options with Kermit support. Productivity improvements include the ability to send a button bar command to all sessions and to optimize data transfer speed with a new send delay option.

Here are some of the new features in SecureCRT 7.1:

#### Tiling for Mac and Linux

Session tiling is now available on Windows, Mac, and Linux. Sessions can be tiled or cascaded to compare the contents of two or more sessions: use the chat window to send commands to all sessions and see the resulting output.

#### **Dependent session option**

Link a session to an SSH2 session that it depends on, which allows connection to a jump host before connecting to other sessions.

#### **Kermit support**

Support for the Kermit protocol provides an additional way to transfer files.

#### **Button bar enhancements**

Press SHIFT+<click> on a button to send a button bar command to all connected tabbed or tiled sessions.

#### New character send delay option

A character send delay option to wait for a text prompt allows data to be sent as fast as the system can handle, rather than using a fixed time delay.

#### **Support for SHA1 in RSA signatures**

Support for SHA1 in RSA signatures, in addition to MD5, allows X.509 certificates to be used in FIPS mode.

#### **Integration with SecureFX for Mac and Linux**

With the availability of SecureFX for Mac and Linux, integration between SecureCRT and SecureFX on Mac and Linux is now

possible.

Please see History.txt in the SecureCRT program group for additional information on other changes and bug fixes.

### What is SecureCRT?

SecureCRT protects your passwords, user accounts, and data, combining rock-solid terminal emulation with the strong encryption, broad authentication options, and data integrity of the Secure Shell protocol.

SecureCRT provides versatile solutions for a business, network management, information security, and development tasks, from accessing host-based applications and administering servers to securely accessing behind-the-firewall network resources like email, files, and printers.

SecureCRT supports <u>SSH1</u>, <u>SSH2</u>, <u>Telnet</u>, <u>Telnet/SSL</u>, RLogin, Serial, and <u>TAPI</u> protocols and authentication via password, public key, X.509 certificate, and Kerberos v5 via GSSAPI. Ciphers include AES, AES-CTR, Twofish, Blowfish, 3DES, and RC4.

Choose from a wide range of terminal emulations including VT100, VT102, VT220, ANSI, SCO ANSI, Wyse 50/60, Xterm, and Linux console – all with ANSI color. Unicode support includes the ability to display character sets from multiple languages, support for languages with right-to-left reading order, and a character encoding list that includes commonly-used encodings as well as all encodings supported by the local system.

SecureCRT features a multi-session tabbed or tiled interface with extensive session management and customization features. Customization options include toolbars, menus, keymaps, button bar, and login scripts, as well as fonts, cursors, and color schemes.

Named sessions and firewalls let you create individual configurations that can be used on a session-specific basis. Other features include auto logon, printing, Emacs mode, and SOCKS firewall support.

Powerful scripting capabilities, including ActiveX scripting support for VBScript and other scripting languages on Windows and embedded Python support, save you valuable time by automating routine configuration tasks. Secure file transfers can be performed using SFTP or by using Zmodem, Xmodem, Ymodem, or Kermit from an SSH1 or SSH2 session. Upload files easily by dragging and dropping them onto an SFTP tab or session window.

Save steps with the close integration of SecureCRT and the SecureFX® file transfer client, which share sessions and settings that let you run SFTP, FTP, SCP, and FTP/SSL file transfer sessions without reentering passwords.

For a list of new features in SecureCRT, see New in SecureCRT 7.1.

#### **Support for SSH1 and SSH2 Secure Shell protocols**

- SSH2 Protocol support:
  - AES-128, AES-192, AES-256, AES-128-CTR, AES-192-CTR, AES-256-CTR, Twofish, Blowfish, 3DES, and RC4 ciphers.
  - RSA and DSA host key support.
  - Multiple ordered authentication methods, ciphers and MACs.
  - Public Key Assistant makes it easier to upload public keys.
  - Local port forwarding, X11 forwarding, remote forwarding, and dynamic forwarding.
  - · OpenSSH Agent forwarding.
  - MD5, MD5-96, SHA1, and SHA1-96 MACs.
  - RSA, DSA, public-key, X.509 certificate, PKCS #11, PKCS #12, Kerberos v5 via GSSAPI, and password authentication methods.
  - SFTP tab creates an SFTP session to an existing SSH session.
  - · Passphrase and password caching options.
- SSH1 Protocol support:
  - Blowfish, DES, 3DES, and RC4 ciphers.
  - RSA, TIS, and password authentication.
  - Local port forwarding, X11 forwarding.

#### **Session Management**

- Named sessions store different preferences for different hosts.
- Tabbed sessions allow multiple sessions in the same window.
- Tabbed sessions can be moved in and out of windows.
- Launch multiple selected sessions in tabs with a single click.
- Tiling allows multiple sessions to be viewed at once.
- A session can be logged to a file, including options for logging custom data and an option for creating a new log file at midnight.

• Built in SSH agent allows keys to be explicitly added and removed.

#### **Configuration & Customization**

- Easy configuration of basic SSH, port forwarding, remote forwarding and other settings in Session Options dialog.
- Named firewalls.
- Dependent session option (jump host).
- Customizable toolbar and menus.
- User-defined number of savelines (scrollback) up to 128,000.
- User-configurable number of rows and columns.
- User-defined foreground, background, and bold colors.
- User-defined keymaps.
- User-defined button bar.
- User-defined word delimiter characters for double-click.
- Emacs mode maps ALT+<key> to send ESC+<key>.
- Real-time keyword highlighting.

#### **Advanced Terminal Emulation**

- Quality VT100, VT102, VT220, Linux console, SCO ANSI, Wyse 50/60, Xterm, and ANSI emulation:
  - · VT line drawing.
  - Support for bold, underline, and reverse attributes.
  - Double-width and double-height fonts.
  - 80/132 column switching.
  - VT100 and VT220 keyboard emulation.
  - Optional ANSI color.
  - 256-color Xterm.
  - Xterm extensions for mouse support and changing title bar.
  - Multi-byte character set support for Japanese, Korean, and Chinese.
  - Unicode support includes the ability to display character sets from multiple languages, support for multi-byte character sets, right-to-left reading order languages, and an extensive character encoding list.

#### **Other Features**

- FIPS 140-2 validated cryptographic library support.
- Variable compression increases performance on slow connections.
- Simple interface for automating logins.
- Support for Telnet, Telnet/SSL, and RLogin protocols:
  - Telnet supports Negotiate About Window Size (NAWS).
  - Telnet supports Local Flow control (LFLOW).
- Serial (COM) device support.
- Integration with SecureFX 3.x or higher.
- Scripting language support for VBScript, JScript, and Perlscript.
- · Script recorder.
- Embedded support for Python scripting.
- Zmodem, Xmodem, Ymodem, and Kermit file transfer (upload and download).
- Built-in TFTP server.
- Drag-and-drop file transfer (upload).
- Printing: auto print, selection, screen, and pass-through printing.
- Modem dialer support configure and save modem, country code, phone, and redial settings for TAPI sessions.
- SOCKS firewall support with password authentication.
- Unauthenticated and basic HTTP proxy support.
- Generic proxy firewall support.
- Copy and paste, including an "auto copy" option and paste on middle or right mouse click.
- Chat window option provides an editable type-ahead buffer with history support and the ability to send chat text to all tabs or to all tiled sessions.
- Support for use from the command line or web browsers.
- Support for standard insertion caret so that it can be tracked by screen access technology for the blind.



### System Requirements

### **Minimum System Requirements**

- Windows® Server® 2008 R2
- Windows 7
- · Windows Server 2008
- · Windows Vista®
- · Windows Server 2003 R2
- · Windows Server 2003
- · Windows XP
- Internet Explorer 5.5 or later

Note: 64-bit editions of the above platforms are supported.

### License Agreement

End-User License Agreement for SecureCRT 7.1 ("Software") Copyright (c) 1995-2013 VanDyke Software, Inc. All Rights Reserved.

AGREEMENT. After reading this agreement carefully, if you ("Customer") do not agree to all of the terms of this agreement, you may not use this Software. Unless you have a different license agreement signed by VanDyke Software, Inc. that covers this copy of the Software, your use of this Software indicates your acceptance of this license agreement and warranty. All updates to the Software shall be considered part of the Software and subject to the terms of this Agreement. Changes to this Agreement may accompany updates to the Software, in which case by installing such update, Customer accepts the terms of the Agreement as changed. The Agreement is not otherwise subject to addition, amendment, modification, or exception unless in writing signed by an officer of both Customer and VanDyke Software, Inc.

This Software is owned by VanDyke Software, Inc. and is protected by national copyright laws and international copyright treaties.

1. EXPORT LAW. This Software is subject to export control. The Software may be transmitted, exported, or re-exported only under applicable export laws and restrictions and regulations of the United States Bureau of Industry and Security or foreign agencies or authorities. By downloading or using the Software, you are agreeing to comply with export controls.

The Software may not be downloaded or otherwise exported or reexported to any country subject to U.S. trade sanctions governing the Software, sanctioned countries including Cuba, Iran, North Korea, Sudan, and Syria, or by citizens or residents of such countries except citizens who are lawful permanent residents of countries not subject to such sanctions, or by anyone on the U.S. Treasury Department's list of Specially Designated Nationals and Blocked Persons or the U.S. Commerce Department's Table of Denial Orders.

- 2. GRANT OF LICENSE AND PROHIBITIONS. This Software is licensed to you. You are not obtaining title to the Software or any copyrights. You may not sublicense, rent, lease, convey, modify, translate, convert to another programming language, decompile, or disassemble the Software for any purpose. The license may be transferred to another individual (not resold) if you keep no copies of the Software. Permission must be obtained before mirroring or redistributing the evaluation copies of the Software.
- 3. USE AND EVALUATION PERIOD. You may use one copy of this Software on one client computer. A copy of this Software is considered in use when loaded into temporary memory (i.e., RAM) and/or installed on a permanent storage device (i.e., hard disk, CD-ROM, etc.). You may also use a copy of the Software on a home or portable computer, provided only one copy of the Software is in use at a time. You may use an evaluation copy of the Software for only thirty (30) days in order to determine whether to purchase the Software.
- 4. MULTI-COMPUTER LICENSES. If this is a multi-computer license, you may make, install, and use additional copies of this Software up to the number of copies authorized in your registration documentation. Unless you have a terminal server license agreement signed by VanDyke Software, Inc., use of this Software is not allowed in environments in which the Software is located on a central server and is accessed by multiple computers. If this is an educational license, use is restricted to non-commercial use by students, faculty, and staff using campus facilities, and to off-campus non-commercial use by students, faculty, and staff primarily for school business.
- 5. LIMITED WARRANTY. THE SOFTWARE IS PROVIDED AS IS AND VANDYKE SOFTWARE DISCLAIMS ALL WARRANTIES RELATING TO THIS SOFTWARE, WHETHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
- 6. LIMITATION ON DAMAGES. NEITHER VANDYKE SOFTWARE NOR ANYONE INVOLVED IN THE CREATION, PRODUCTION, OR DELIVERY OF THIS SOFTWARE SHALL BE LIABLE FOR ANY INDIRECT, CONSEQUENTIAL, OR INCIDENTAL DAMAGES ARISING

OUT OF THE USE OR INABILITY TO USE SUCH SOFTWARE EVEN IF VANDYKE SOFTWARE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES OR CLAIMS. IN NO EVENT SHALL VANDYKE SOFTWARE'S LIABILITY FOR ANY DAMAGES EXCEED THE PRICE PAID FOR THE LICENSE TO USE THE SOFTWARE, REGARDLESS OF THE FORM OF CLAIM. THE PERSON USING THE SOFTWARE BEARS ALL RISK AS TO THE QUALITY AND PERFORMANCE OF THE SOFTWARE.

- 7. TERMINATION. This Agreement terminates on the date of the first occurrence of either of the following events: (1) The expiration of one (1) month from written notice of termination from Customer to VanDyke Software, Inc.; or (2) One party materially breaches any terms of this Agreement or any terms of any other agreement between Customer and VanDyke Software, Inc., that are either uncorrectable or that the breaching party fails to correct within one (1) month after written notification by the other party.
- 8. GOVERNING LAW. The agreement shall be governed by the laws of the State of New Mexico. Any action or proceeding brought by either party against the other arising out of or related to this agreement shall be brought only in a state or federal court of competent jurisdiction located in Bernalillo County, New Mexico. The parties hereby consent to the personal jurisdiction of such courts.
- 9. U.S. GOVERNMENT RESTRICTED RIGHTS. This Software is provided with RESTRICTED RIGHTS. Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (a) of the Rights in Commercial Computer Software clause at DFARS 227.7202-3 or subparagraphs (c)(1) and (2) of the Commercial Computer Software -- Restricted Rights clause at 48 CFR 52.227-19, as applicable. Manufacturer is:

VanDyke Software, Inc. 4848 Tramway Ridge Dr. NE, Suite 101 Albuquerque, NM 87111 USA Inquiry form: http://www.vandyke.com/feedback.php



The following information is for individual licenses.

For information on quantity pricing and site licenses, please visit:

http://www.vandyke.com/feedback.php

Upon receipt of payment, we will send you one license key for each copy of SecureCRT paid for. The license key(s) will be sent to you by email or postal mail. All software and software updates are available from <a href="https://www.vandyke.com">www.vandyke.com</a>. Because the software is obtained by Internet download, no disk media will be mailed.

For current pricing information, please visit our website:

http://www.vandyke.com/pricing/index.html

#### **Ordering Methods**

#### Order online with a credit card.

To order online using a secure web server, go to:

http://www.vandyke.com/purchase/index.html

#### Order by phone with a credit card.

You can order by phone, call VanDyke Software between 7:30 a.m. and 5:00 p.m. U.S. Mountain Time, Monday through Friday.

Phone: 505-332-5710

To order using a different method, such as fax or check, please contact us for ordering instructions.

Email: sales@vandyke.com

Phone: 505-332-5710

Visa, MasterCard, American Express, and Discover are accepted. Please include the credit card number, cardholder name, expiration date and

billing address.

Orders will be processed within one to two business days after they are received by VanDyke Software.

Any questions about the status of the shipment of the order, refunds, registration options, product details, technical support, volume discounts, dealer pricing, site licenses, or non-credit card orders, must be directed to VanDyke Software, Inc., 4848 Tramway Ridge Dr. NE, Suite 101, Albuquerque, NM 87111, USA or <a href="http://www.vandyke.com/feedback.php">http://www.vandyke.com/feedback.php</a>

If you are a registered user of CRT, please visit our  $\frac{\text{website}}{\text{pricing}}$  for current  $\frac{\text{upgrade}}{\text{pricing}}$ , or call 505-332-5700.



### Product Announcements

If you would like to receive email notification of new releases, please go

https://secure.vandyke.com/cgi-bin/subscribe.php?PID=3

Also, visit our web page:

http://www.vandyke.com



### **Copyright and Trademark Notices**

SecureCRT® 7.1 Documentation Copyright © 1995-2013 VanDyke Software®, Inc. All Rights Reserved.

VanDyke Software, SecureCRT, SecureFX, and VShell are trademarks or registered trademarks of VanDyke Software, Inc. in the United States and/or other countries.

All other products and services mentioned are trademarks or registered trademarks of their respective companies.

SecureCRT includes the VeriSign XCode (encode/decode) IDN Library. Copyright (c) 2003, VeriSign Inc. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- 3. Neither the name of the VeriSign Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF

USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER
IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE
USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY
OF SUCH DAMAGE.

This software is licensed under the BSD open source license. For more information visit www.opensource.org.



### **Reporting Problems**

You can submit any problems you are having with SecureCRT at our website:

http://www.vandyke.com/feedback.php

VanDyke Software will try to respond to all bug reports within one to two business days, and will try to resolve the problem as quickly as possible.



## Feature Requests and Questions

We are very interested in hearing from our users. If there is a feature that you would like to see added to SecureCRT, or if you have a general question about SecureCRT, please go to

http://www.vandyke.com/feedback.php

If you would like to receive email notification of new releases, please go to:

https://secure.vandyke.com/cgi-bin/subscribe.php?PID=3

# Support FAQ

The SecureCRT FAQ (Frequently Asked Questions) can be found on our website at:

http://www.vandyke.com/support



### Connection Overview

This section provides crucial information to those who are interested in getting connected to a remote machine as quickly as possible, putting aside any environmental customizations such as font color, etc.

When you start SecureCRT, you will see the Quick Connect dialog appear along with the main SecureCRT window.

There are two interfaces you can use to connect to a remote machine: the **Quick Connect** dialog and the **connect** dialog.

If this is your first time using SecureCRT, you may prefer to use the Quick Connect dialog.

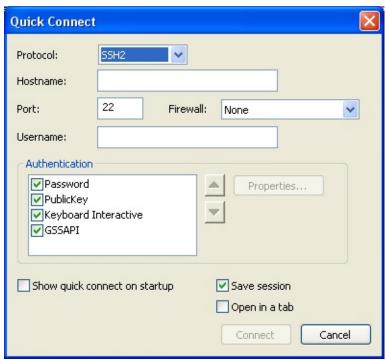
If you are a seasoned user of SecureCRT and have already defined one or more sessions, you may prefer to use the Connect dialog.

Regardless of the interface you choose, you will be able to establish a connection to a remote machine using the Serial, Telnet, SSH1, SSH2, Telnet/SSL, and RLogin protocols. If you need to establish a connection to a remote machine (such as a BBS) by dialing it directly with a modem, see the topic on Connecting with the TAPI dialer.



### **Creating a New Session with Quick Connect**

The **Quick Connect** dialog is the fastest way to create a new session and connect to the machine associated with it. By default, the **Quick Connect** dialog (see figure below) is shown when SecureCRT is started.



SecureCRT Quick Connect Dialog

The dialog can also be opened by clicking on the **Quick Connect** toolbar button 30 on either the SecureCRT session window or the connect dialog, or by opening the File menu and selecting the **Quick Connect** menu option.

In the **Quick Connect** dialog, specify the protocol you will be using, the hostname or IP address, and any other information necessary to make the  $\underline{\text{connection}}$ .

When you are finished entering the necessary settings for the connection, click on the **Connect** button and you will be connected to the specified host. If the **Save session** option is selected (this option is on by default), a session will be created and stored for future use under the name of the host or IP address you specified.

**Note:** You can customize SecureCRT to display the **Quick Connect** dialog on startup rather than the **Connect** dialog by selecting the

Show quick connect on start up option located in the  $\frac{Quick\ Connect}{Quick\ Connect}$  dialog.

Once you have created a session, you may wish to customize its behavior. For more information, see the topics under <u>Session Options</u>.



### **Connecting with the Connect Dialog**

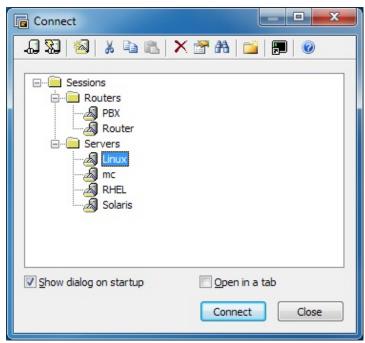
The <u>connect</u> dialog can be accessed with the **Connect** button  $\square$  on the toolbar or by selecting the **Connect** menu item from the <u>File</u> menu. In order to connect to a remote machine using the **Connect** dialog, you must select a previously created <u>session</u>.

You can quickly create a new session by clicking on the <u>New Session</u> button or by right-clicking on any folder in the **Connect** dialog and selecting **New Session** from the pop-up menu. If you would like to learn more about creating new sessions, see <u>Creating a New Session with Connect</u>.

To connect using a session you have already created, select the session and click on the **Connect** button. You can also connect to a session by double-clicking on the session name or by right-clicking on the session name and selecting **Connect** from the pop-up menu.

To display the **Connect** dialog automatically at startup, select the **Show** dialog on startup option found in the lower left corner of the **Connect** dialog.

To open the selected session as a tab in the SecureCRT window, select the **Open in a tab** option.



Connect Dialog



### Connecting with the TAPI Dialer

To connect to a remote machine (such as a BBS) by dialing it directly, you will need to create a new session for the connection, or use the quick **Connect** dialog with he following steps:

- 1. Specify TAPI as the connection protocol.
- 2. In the **Dial using** entry box, specify the modem you will use for dialing the remote machine.
- 3. If necessary, specify the country and/or area code for the number of the machine you will be dialing.
- 4. In the **Phone number** entry box, enter the phone number of the machine you will be dialing. If the remote machine or BBS you are attempting to dial provides you with alternate numbers, you can specify any alternate numbers by clicking on the TAPI/Advanced category in the Session Options dialog.
- 5. Click on the **Connect** button and SecureCRT will begin dialing the remote computer.



### 随 Starting a Session Automatically

SecureCRT can be configured to automatically connect to a specific session on startup. If you intend to use one particular session each time you run SecureCRT, you may want to enable the **Use auto session** option.

#### To enable **Use auto session**:

- 1. Select the General/Default Session Category in the Global Options dialog.
- 2. In the Auto session setup group, select the Use auto session option and specify a session name.

#### **Creating a Desktop Shortcut to a SecureCRT Session**

With SecureCRT, you can easily create a shortcut on the desktop that will start SecureCRT and automatically begin connecting with a certain session.

To create a desktop shortcut to a SecureCRT session:

- 1. In the connect dialog, right-click on the specific session that you want SecureCRT to begin connecting with once the shortcut is launched from the desktop.
- 2. Select the **Create Desktop Shortcut** menu item from the pop-up menu.



### Reconnecting Automatically

SecureCRT can be set up to automatically reconnect to a remote machine when a connection is terminated.

To enable this feature for all sessions:

- 1. Open the <u>Terminal</u> category in the <u>Global Options</u> dialog.
- 2. Select the **Auto reconnect** option.

To enable or disable this feature on a per-session basis:

- 1. In the <u>Session Options</u> dialog, open the <u>Terminal</u> category.
- 2. Select the **Auto reconnect** option.



### Running SecureFX from SecureCRT

The 5.0 (or newer) release of SecureCRT supports the integration of its configuration settings with those of VanDyke Software's secure file transfer client SecureFX® (version 3.0 or newer). This makes it simpler to create and maintain sessions; set your options once and they are used by both programs.

To integrate SecureCRT with SecureFX, install both applications on the same machine then start SecureCRT. From SecureCRT, you can start SecureFX in several different ways, including the following:

- Clicking on the **SecureFX** toolbar button.
- Right-clicking on a tab and selecting the **Start SecureFX** option.
- Selecting a SecureFX session from the **Connect** dialog.
- Right-clicking on the Activator system tray icon and opening a SecureFX session.

You can also start SecureCRT from SecureFX if you prefer.

If you open the SecureCRT Global Options or Session Options dialogs, you will see options for both SecureCRT and SecureFX. From here, you can configure sessions for both applications.

If you want to share your SecureCRT and SecureFX configuration settings and want to see both the terminal emulation and file transfer sessions in your SecureCRT Connect dialog, right-click on the Connect dialog and check the **Show Transfer Sessions** option.

### Menus

SecureCRT application window has a number of menus that provide access to program dialogs, controls, and features. The menus are listed below:

- System Menu
- File Menu
- Edit Menu
- · <u>View Menu</u>
- Options Menu
- Transfer Menu
- Script Menu
- Tools Menu
- Window Menu
- Help Menu



This topic lists the options available on the **System** menu. The **System** menu is accessed by clicking on the small SecureCRT icon located in the upper left-hand corner of the SecureCRT window. The **System** menu can also be accessed by right-clicking in the SecureCRT title bar.

The **System** menu allows you to accomplish basic tasks such as minimizing, maximizing, moving, and resizing the SecureCRT window on systems that do not have a mouse attached. To access the **System** menu without using the mouse, use the ALT+SPACE key combination. To select a menu item found in the **System** menu, press the key associated with the entry as designated with an underline.



SecureCRT System Menu

#### Restore

Restores the session window to its original size.

#### Move

Changes the mouse pointer into a handle to move the session window.

#### Size

Changes the mouse pointer into a handle to resize the session window.

#### **Minimize**

Minimizes the session window to the taskbar or, if you are using the Activator, to the system tray.

#### **Maximize**

Enlarges the session window to fill the entire screen.

#### Toggle Menu Bar

Toggles the menu bar on and off. This option is also available on the <u>view</u> menu.

#### **Always on Top**

Keeps the session window in front or "on top" of other windows on your screen. This option is also available on the view menu.

#### **Save Settings Now**

Saves your current global and session option settings to the initialization file. These settings will be the ones retrieved the next time you start SecureCRT. This option is also available on the options menu.

#### Close

Disconnects any sessions in progress and closes the session window.



This topic lists the options available on the **File** menu including connection, printing, and logging commands.

#### Connect...

Opens the <u>connect</u> dialog where you can connect to or edit existing sessions, or create new sessions. If you choose to open a new session and the current tab or window is disconnected, SecureCRT will display the new session in the current tab or window. If the current tab or window is connected, a new window will be opened.

#### **Quick Connect...**

Opens the <u>Quick Connect</u> dialog. This is the quickest way to define a <u>session</u> and is often used to make one-time connections.

#### Connect in Tab/Tile...

Opens the **Connect in Tab/Tile** dialog where you can open a session as a new tab or as a new tiled session if you are in tiled mode. If you choose to open a new session and the current tab/tiled session is disconnected, SecureCRT will display the new session in the current tab/tiled session. If the current tab/tiled session is connected, a new tab/tiled session will be opened in the current window.

#### Reconnect

Reconnects you to the session that most recently had focus.

#### Reconnect All

Re-establishes a connection on all currently disconnected sessions (honors the <u>connect to multiple sessions sequentially</u> option setting).

#### **Disconnect**

Disconnects you from the current session.

#### **Disconnect All**

Disconnects you from all currently connected sessions.

#### Clone Session

Creates a new session that is identical to the current session. For more information, see the <u>Session Tabs</u> topic.

#### **Connect SFTP Session**

Opens a new tab or tiled session containing an SFTP command line.

#### Lock/Unlock Session

Locks or unlocks the current session.

#### **Print**

#### **Auto Print**

Toggles on and off the Auto Print feature. When on, this feature directs terminal output to the printer; each line of output to the terminal screen is sent to the printer as soon as a CR, LF, FF, or VT character is received.

**Note:** This output will be buffered by the Windows spooler unless you have enabled the **Use raw mode** or **Direct pass-through printing** option in either the **Session Options** or **Global Options** dialog.

You can also turn this feature on and off by clicking the **Auto Print** button located on the SecureCRT toolbar. Selecting **Print/Cancel** from the **File** menu, also turns off the Auto Print feature.

**Note:** If Auto Print is on, the **Auto Print** button on the toolbar will appear depressed . If auto print is off, the **Auto Print** button will appear as normal.

#### Screen

Prints the contents displayed in the terminal screen. You can also print the contents of the terminal screen by clicking on the **Print Screen** button on the SecureCRT toolbar.

#### Selection

Prints the text selected in the terminal screen to the printer. You can also print the currently selected text by clicking on the **Print Selection** button so on the SecureCRT toolbar.

#### **Eject Page**

Sends a formfeed command to the printing system.

#### Cancel

Cancels all pending SecureCRT print jobs. This will not affect print

jobs that are already in the print buffer.

#### **Print Setup...**

Opens the Windows Print Setup dialog.

#### **Log Session**

Records the current session in a log file. When first logging a session, the **Save As** dialog will appear. Enter the desired log filename and click on the **Save** button. Also, see the <u>Terminal/Log File</u> category of the <u>Session Options</u> dialog. By default, logging is off.

#### **Raw Log Session**

Writes every character received by SecureCRT , including terminal escape sequences, to the session log file. By default, **Raw log** is off.

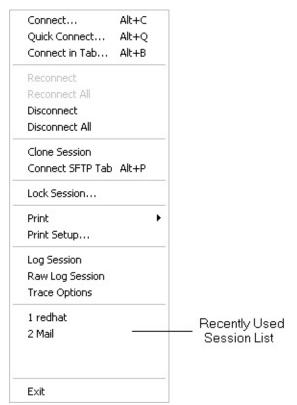
#### **Trace Options**

Displays normally hidden protocol negotiation information along with standard information shown in the SecureCRT window.

When SecureCRT attempts to connect to a target server, a great deal of information is passed between the two machines. Most of this information is visible in the SecureCRT window but some of it (the underlying "handshaking" commands) is generally unimportant to the user and is hidden from view. However, there are times when this handshaking information is useful, for example, it can be very valuable when trying to solve a connection problem. For these instances, SecureCRT provides the **Trace Options** feature which, when turned on, displays the handshaking information.

**Recently Used Session List** 

Displays the names of the most recently used sessions used to connect to remote machines. The number of session shown can be configured in the <u>Global Options/General</u> dialog.



SecureCRT File Menu

**Exit** 

Closes the SecureCRT application.



This topic lists the options available on the **Edit** menu.

#### Copy

<u>Copies</u> selected text from the SecureCRT <u>session</u> window to the Windows Clipboard.

#### **Paste**

<u>Pastes</u> the contents of the Windows clipboard to the session window at the point of the text cursor

#### **Copy and Paste**

<u>Copies</u> selected text from the SecureCRT session window and <u>pastes</u> it at the location of the text cursor.

#### **Paste as Quotation**

Pastes text from the Windows Clipboard prefaced by the quotation character (usually ">").

#### Select All

Selects all the text in the active session window including text in the scrollback buffer.

#### Find...

Opens the **Find** dialog which lets you search the text in a session window and <u>scrollback buffer</u>. for a specific character string.

#### **Print Selection**

Sends the text selected in the session window to the printer. You can also do this by clicking on the **Print Selection** button located on the SecureCRT toolbar.

#### Go to Session/Chat Window

This menu item reads **Go to Session Window** when the chat window is displayed and has focus. When selected it causes the focus to go to the session window.

When the chat window is displayed and focus is in the session window, this menu item reads **Go to Chat Window** and selecting it causes the

focus to go to the chat window.

#### Clear Scrollback

Clears the scrollback buffer.

#### **Clear Screen**

Clears the session window.

#### **Clear Screen and Scrollback**

Clears both the session window and the scrollback buffer.

You can also automatically clear the session window when you disconnect from a remote machine by checking the Clear on disconnect option in the Session Options/Terminal dialog

**Send Break** 

Sends a Telnet break to the session.

#### Reset

Resets SecureCRT the terminal emulator. This option is useful for clearing incidental display abnormalities (e.g., accidental display of binary data).



This topic lists the options available on the **View** menu.

#### Menu Bar

Toggles the menu bar on and off. The menu bar can also be toggled on and off from the <u>system</u> menu.

#### **Toolbar**

Toggles the toolbar on and off.

#### **Session Tabs**

Toggles the <u>session</u> tabs on and off. Tabs can also be enabled using the <u>Show session tabs</u> Option on the <u>Global Options/Terminal/Appearance</u> dialog.

#### **Chat Window**

Toggles the chat window on and off.

The SecureCRT chat window provides an interface to the remote command line that is unaffected by concurrent output to the terminal screen. The chat window is a great interface to use when communicating in the environment of a chat room. In addition, the chat window is useful for issuing subsequent commands. While waiting for the output of a previous command to complete, you can prepare the next command by typing it into the chat window. The contents of the chat window are sent to the command line as soon as the ENTER key is pressed in the chat window.

You can prepare multiple lines in the chat window with CTRL+ENTER.

SecureCRT will remember the number of recently used chat window commands that are specified in the **Chat window history** option of the **Global Options/Terminal/Appearance** dialog. You can cycle through these commands using your UP and DOWN arrow keys. Entering a 0 for the **Chat window history** will turn off this feature.

If you want to send commands to all your open tabbed sessions at once, right-click in the chat window and select the **Send Chat to All Tabs** option. Commands will be sent to all your tabbed sessions as long as this option is enabled,

#### **Connect Bar**

Toggles the connect bar on and off.

#### **Status Bar**

Toggles the status bar on and off.

#### **Button Bar**

Toggles the button bar on and off.

#### **Themes**

Allows you to select from the available Windows display themes.

## **Window Transparency**

Toggles the active window transparency between fully opaque and the transparency value set in the <u>Terminal/Appearance/Advanced</u> category of the <u>Global Options</u> dialog.

#### **Horizontal Scroll Bar**

Toggles the horizontal scroll bar on and off. Turning off the scroll bar does not turn off scrolling.

#### **Vertical Scroll Bar**

Toggles the vertical scroll bar on and off. Turning off the scroll bar does not turn off scrolling.

## **Always on Top**

Keeps the session window in front or "on top" of other windows on your screen.

#### **Full Screen**

Enlarges the session window to full screen size. The ALT+ENTER key combination returns the window to its standard size.



This topic lists the options available on the **Options** menu.

## **Session Options...**

Opens the **Session Options** dialog.

## **Global Options**

Opens the Global Options dialog.

When you start SecureCRT, the program retrieves your global and session option settings from an initialization file. While you are running SecureCRT, you can make changes to your settings, but these changes will be lost when you close SecureCRT unless you save your changes to the initialization file using the **Save Settings Now** command and/or the **Auto Save Options** command.

## **Auto Save Options**

Automatically saves changes that you make in the <u>Global Options</u> dialog or <u>Session Options</u> dialog to the initialization file when you click on the **OK** button on the dialog.

If this option is off (unchecked), changes made to your global or session options will be lost when you close SecureCRT unless you have saved the settings using the **Save Settings Now** command.

**Note:** When the **Auto Save Options** command is being used by more than one instance of SecureCRT, the settings of the last instance to be closed will be the settings saved to the initialization file.

## **Save Settings Now**

Saves your current global and session option settings to the initialization file. These settings will then be the ones retrieved the next time you start SecureCRT.



This topic lists the options available on the **Transfer** menu.

#### Send ASCII...

Opens the Select File to Send as ASCII dialog.

#### Receive ASCII...

Opens the <u>Specify Filename to Receive as ASCII</u> dialog.

## Send Binary...

Opens the Select File to Send as Binary dialog.

#### Send Kermit...

Opens the Select File to Send using Kermit dialog.

#### Receives Kermit...

Starts the transfer of the file(s) from the currently connected server.

#### Send Xmodem...

Opens the <u>Select File to Send using Xmodem</u> dialog. For more information on Xmodem, see Overview of Xmodem.

#### Receive Xmodem...

Opens the <u>Specify Filename to Receive using Xmodem</u> dialog. For more information on Xmodem, see Overview of Xmodem.

#### Send Ymodem...

Opens the <u>Select Files to Send using Ymodem</u> dialog. For more information on Ymodem, see <u>Overview of Ymodem</u>.

#### Receive Ymodem...

Starts the transfer of the file(s) from the currently connected <u>server</u>. For more information on Ymodem, see <u>Overview of Ymodem</u>.

## **Zmodem Upload List...**

Opens the <u>Select Files to Send using Zmodem</u> dialog. For more information on <u>Zmodem</u>, See <u>Overview of Zmodem</u>.

## **Start Zmodem Upload**

Starts the transfer of the file(s) selected in the <u>select Files to Send using Zmodem</u> dialog to the currently connected server. For more information on Zmodem, see <u>Overview of Zmodem</u>.



This topic lists the options available on the **Script** menu.

#### Run...

Opens the **Select Script to Run** dialog where you can choose and start a script. For more information on scripts, see <u>Creating ActiveX Scripts</u>.

#### Cancel

Cancels the currently running script.

## **Start Recording Script**

When this option is selected, all keystrokes and responses in the current session will be recorded until the **Stop Recording Script** or **Cancel Recording Script** option is selected. If a script is being recorded and a session is disconnected, you will be prompted to save the script.

## **Stop Recording Script...**

Stops the recording in progress and prompts you to save the recorded script as a Visual Basic (.vbs) or Python (.py) script file. If a script is being recorded and a session is disconnected, you will also be prompted to save the script.

## **Cancel Recording Script**

Stops the recording in progress and nothing is saved.

## **Recent Script**

Lists the most recently used scripts. For more information on scripts, see <a href="Creating ActiveX Scripts">Creating ActiveX Scripts</a> and <a href="Creating Python Scripts">Creating Python Scripts</a>.



This topic lists the options available on the **Tools** menu.

## **Keymap Editor...**

Opens the Keymap Editor.

## **Create Public Key...**

Opens the Key Generation wizard.

## Convert Private Key to OpenSSH Format...

Converts a VanDyke generated private key to OpenSSH format so that it can be used for authentication with other SSH2 clients. You will be prompted for the key to convert and where to save the converted key.

## **Export Public Key from Certificate...**

Exports the public key from an X.509 certificate. You will be prompted for the certificate to use and where to save the public key. After the public key has been exported, you will need to complete these instructions in order to use it.

## **Public-Key Assistant...**

Opens the <u>Public-Key Assistant</u> dialog, allowing you to manage your public keys on remote servers.

This option is only available when you are using the <u>SSH2</u> protocol. Also note that not all remote systems support the public-key assistant subsystem.

## **Manage Agent Keys**

Opens the **Manage Agent Keys** dialog which allows you to add keys to and remove keys from the list of accepted agent keys.

# 👼 Window Menu

This topic lists the options available on the **Window** menu. Of the following four options, only one can be selected at a time.

#### **Tabs**

Toggles the tabbed interface display setting.

## **Tile Vertically**

Resizes and displays all open session windows to be the height of the main window, equal-sized and horizontally positioned to fill the entire main window.

## **Tile Horizontally**

Resizes and displays all open session windows to be the width of the main window, equal-sized and vertically positioned to fill the entire main window.

#### Cascade

Resizes and displays all open session windows to be slightly overlapped and descending from the upper-left corner of the main window. The specified geometry settings are used.

#### **Active Session List**

Displays the names of the currently active sessions.



This topic lists the options available on the **Help** menu.

## **Help Topics**

Opens the SecureCRT Help system.

## **SecureCRT Web Page...**

Opens your default web browser to the SecureCRT product web page.

#### Order SecureCRT...

Opens your default web browser to the SecureCRT order web page.

## **Check for Updates...**

Opens your <u>default web browser</u> to a SecureCRT web page that will let you know if there are any versions available that are newer than the one you are currently running.

### VanDyke Software Forum...

Opens your <u>default web browser</u> to the VanDyke Software web forum where you can read and participate in discussions about SecureCRT and other VanDyke products.

#### **Enter License Data...**

Opens the License Wizard which will step you through entering your license data.

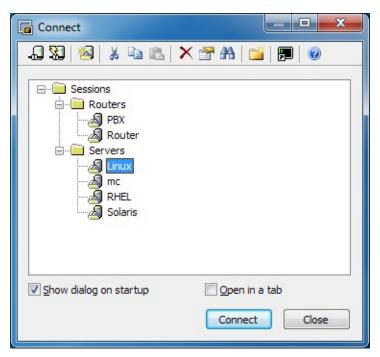
#### About SecureCRT...

Opens the **About** dialog which contains information about SecureCRT.



# **Connect Dialog**

The **Connect** dialog (shown below) allows you to connect to a defined session or to create and connect to a new session.



The **Connect** dialog can be accessed by clicking on **Connect...** in the **File** menu, by clicking on the **Connect** button on the SecureCRT toolbar, or by entering the keyboard shortcut ALT+C.

From the **Connect** dialog, you can connect to multiple sessions and folders. If multiple sessions or a folder is selected, the sessions will be opened in tabs. You can also select and edit multiple sessions from the **Connect** dialog; only the changed fields will be applied to the selected sessions, allowing specific settings to be changed for a group of sessions. When editing multiple sessions, SecureCRT will display the first session that was selected.

The <u>toolbar</u> on the **Connect** dialog provides quick access to the following actions:

Button	Description		
0.0	Connect (Connect dialog): Connects to selected session.		

<del>22</del>	Quick Connect: Brings up the Quick Connect dialog.
<u> </u>	New Session (Connect dialog): Opens the Session Options dialog for a new session.
*	<b>Cut</b> : Copies the selected session or folder to the clipboard. When it is pasted, the session or folder is moved to the new location.
	<b>Copy</b> : Copies the selected session or folder to the clipboard.
	Paste: Pastes a session or folder that was previously cut or copied into the specified folder.
×	<b>Delete</b> : Deletes the selected sessions and/or folders.
<b>*</b>	<b>Properties</b> : Opens the <u>session Options</u> dialog for the selected session.
<b>₽</b>	<b>Find</b> : Searches the contents of the session tree for a given string. The accelerator for this command is CTRL+F, and pressing the F3 key will result in a "find next" operation.
	<b>New Folder</b> : Creates a new folder in the folder tree list.
	Create Shortcut: Creates a desktop shortcut to the selected session.
0	<b>Help</b> : Invokes the online help manual.

Organized in a folder tree arrangement, the list window of the **Connect** 

dialog lists the sessions that have been defined.

The **Connect** dialog also contains the following options:

## **Show dialog on startup**

Check this box to have the **Connect** dialog displayed whenever SecureCRT starts.

## **Open in a tab** (if you are in tab mode)

Opens the selected session(s) in a tab in the SecureCRT window. If this option is checked and there is a session open, the selected item(s) will be opened in the current window. If this option is unchecked and there is a session open, the selected item(s) will be opened as tabs in a new window.

## **Open in a tiled window** (if you are in tiled mode)

Opens the selected session(s) in a tiled session in the SecureCRT window. If this option is checked and there is a session open, the selected item(s) will be opened in the current window. If this option is unchecked and there is a session open, the selected item(s) will be opened as tiled sessions in a new window.

#### Connect

Press this button to connect to the session selected in the session window.

#### Close

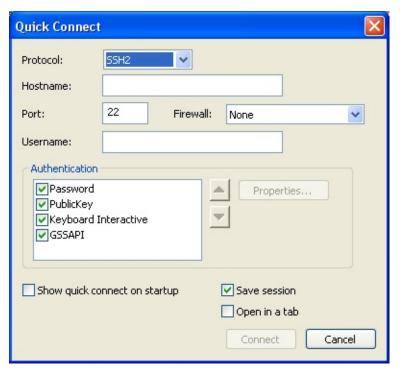
Press this button to close the **Connect** dialog.

If you want to share your SecureCRT and SecureFX configuration settings and want to see both the terminal emulation and file transfer sessions in your SecureCRT **Connect** dialog, right-click on the **Connect** dialog and check the **Show Transfer Sessions** option.



## **Quick Connect Dialog**

The **Quick Connect** dialog (shown below) allows you to quickly define and connect to a remote host.



The **Quick Connect** dialog can be accessed by clicking on **Quick Connect...** in the **File** menu, by clicking on the **Quick Connect** button 33 on the SecureCRT toolbar, or by entering the keyboard shortcut ALT+Q.

The **Quick Connect** dialog options are based on the choice of **Protocol** In other words, if SSH2 is selected as the protocol, the options displayed on the dialog will be different than if Telnet is selected. Below is a list of the protocols and options available in SecureCRT.

SSH2 Quick Connect Options

SSH1 Quick Connect Options

Telnet Quick Connect Options

Telnet/SSL Quick Connect Options

RLogin Quick Connect Options

Serial Quick Connect Options

TAPI Quick Connect Options

Raw Quick Connect Options

The **Quick Connect** dialog also contains the following options:

### Show quick connect on start up

Check this box to have the **Quick Connect** dialog displayed whenever SecureCRT starts.

#### Save session

Check this box to save the <u>session</u> settings that have been defined. The session will be stored for future use under the specified hostname or IP address. To customize the behavior of a saved session, see the topics in the "Customizing Session Behavior" section.

## Open in a tab

Opens the session as a tab in the session window.

#### Connect

Press this button to connect to the specified host. If the **Save session** option is selected, the session is saved at this time.

#### Cancel

Press this button to close the **Quick Connect** dialog without saving the specified settings.

# 8

# **Global Options Dialog**

The **Global Options** dialog is accessed by opening the **Options** menu and selecting **Global Options...** .

Global options apply to all sessions unless they are specifically overridden on a per <u>session</u> basis.

The available categories and subcategories are as follows:

- General
  - Default Session
- Terminal
  - Appearance
    - ANSI Color
    - Advanced
  - <u>Tabs/Tiling</u>
  - Printing
  - Web Browser
  - Advanced
- File Transfer
  - Options
    - Advanced
  - View
  - <u>FTP</u>
  - SSL
  - Sound
  - File Types
  - Filters
- Firewall
- · SSH2
- SSH Host Keys

# **General**

The **General** category of the **Global Options** dialog allows you to configure general SecureCRT options.

## **Dialogs group**

This group contain options that control the display of optional dialogs. These dialogs are displayed by default.

## Show connection closed dialog

Check this option to display a <u>connection</u> closed message when the connection is closed.

## Show confirm disconnect dialog

Check this option to display a message asking for confirmation before disconnecting .

## Show connect dialog on startup

Check this option to display the connect dialog on startup of SecureCRT.

## Save expanded folder state in connect dialog

Check this option to have SecureCRT remember the state (expanded or collapsed) of folders in the **connect** dialog.

#### New session uses wizard

Enabling this option causes the **New Session** toolbar button in the **Connect** dialog to start the New Session wizard. Disabling this option causes this button to open the <u>session Options</u> dialog instead. This option is enabled by default.

## View group

This group includes options that allow you to show or hide components of the SecureCRT application window. To turn off any of these options, clear the associated check box. The available options are as follows:

Show toolbar

Show status bar

Show menu bar

Show connect bar

**Sessions group** 

This group includes options that deal with SecureCRT sessions. The available options are as follows:

## **Connect dialog sort order**

Select the method by which you want to sort the entries listed in the session tree of the **Connect** dialog. Sessions can be sorted alphabetically (i.e., A1, A10, A2, B), logically (i.e., A1, A2, A10, B), or manually. This option can also be controlled from the **Connect** dialog's **Arrange Tree** context menu option. See also Arranging Entries in the Connect Dialog.

## Minimize to Activator in the system tray

Check this option to enable the SecureCRT Activator, which allows you to minimize specific sessions or all sessions to the system tray instead of the taskbar. This can be useful if you have a number of sessions running or if you only use SecureCRT to port forward your connections to another machine. See Using the Activator for more information about this feature. Also see Switching Between Multiple Sessions.

## **Recently used sessions list**

This option allows you to configure the number of sessions listed in the **Recently Used Sessions** section of the **File** menu. The **Recently Used Sessions** list displays a list of sessions that have recently been used to connect to a remote machine. The default value is 4.

**Note:** Entering a value of 0 (zero) in the **Recently used sessions list** entry box will disable the list in the **File** menu and clear the list of all entries.

## **Configuration folder**

This entry box displays and allows you to change the SecureCRT configuration folder. This folder is where the SecureCRT configuration files are stored. SecureCRT will look for the configuration settings in the application directory before using the folder specified in this entry box. Folders specified using the command-line option /F will take precedence over both these locations.

**Note:** To <u>change your configuration</u> information, you must have write privileges to this folder.



## General / Default Session

The **General/Default Session** category of the **Global Options** dialog allows you to configure SecureCRT to automatically connect to a session on startup and it also allows you to configure the SecureCRT default session which is used as the template for all new sessions.

## Auto session setup group

This group contains configuration options for automatically connecting to sessions on startup.

## Do not connect automatically on startup

Check this option to instruct SecureCRT not to automatically connect when the application starts.

#### Remember and connect to sessions from last use

Check this option to instruct SecureCRT to remember the sessions used during the last time SecureCRT was used and to connect to those sessions the next time the application is used.

#### Use auto session

Check this option to automatically connect to one or more sessions when SecureCRT starts. The window below this option displays the sessions configured to connect on startup. You can add or delete sessions using the Add... and Delete buttons.

## Edit default settings group

SecureCRT maintains an internal session that is used whenever a new session is being created or when a session has not been specified for a connection. This internal session is known as the "Default" session. For example, when using Quick Connect or when starting SecureCRT from the command line with no session argument, SecureCRT uses the settings specified in the Default session to make the connection requested.

## **Edit Default Settings**

Press this button to open the Session Options - Default dialog for the default session. In this dialog, you can customize the settings in any category. When you have finished making your changes, click on the OK button and the Apply default session changes dialog is displayed.

The Apply default session changes dialog allows you to choose between Change the Default session only (this is the same behavior provided by the No option in previous versions) and Change ALL sessions (no undo) (this is the same behavior provided by the Yes option in previous versions). The dialog also includes a WARNING that there is no UNDO for this change.

# Terminal

The **Terminal** category of the <u>Global Options</u> dialog allows you to configure SecureCRT terminal emulation-related options.

#### Save window state for each session

Check this option to save the window size and position of each session window.

#### Auto reconnect

Check this option to automatically reconnect to a remote machine when a connection is terminated. This feature can also be enabled or disabled on a per session basis.

### Mouse group

This group contains options that affect mouse behavior.

## **Copy on select**

This option allows you to speed up the process of copying text from the  $\underline{\text{terminal}}$  screen to the Clipboard. Check this option to allow text to be copied to the Clipboard as soon as it is selected.

## Paste on middle/right button

This option allows you to speed up the process of pasting content from the Clipboard to the terminal screen. Check this option to allow text to be pasted from the Clipboard to the terminal screen as soon as the middle or right mouse button is pressed. If your mouse does not have a middle mouse button, a middle button click can be simulated by pressing both the left and right mouse buttons simultaneously.

## **Hide mouse pointer on keypress**

Check this option to hide the mouse pointer when you start typing.

## Open URL with CTRL +click

Check this option to allow a URL in the terminal window to be opened by clicking on it using the CTRL+left-click key combination.



## 随 Terminal / Appearance

The Terminal/Appearance category of the Global Options dialog allows you to configure the appearance of your SecureCRT application window.

## **Options** group

This group includes options that deal with the SecureCRT application window's appearance. To turn off any of these options, clear the associated check box. The available options are as follows:

#### Show session tabs

This option enables or disables the session tabs. This can also be controlled from the Session Tabs Option on the View menu

#### Show chat window

#### **Show button bar**

By checking or unchecking this option, you can display or hide the button bar (this can also be controlled from the view menu). This option also allows you to display the button bar either at the bottom of the window or at the top.

#### Show horizontal scroll bar

#### Show vertical scroll bar

## Use ClearType to smooth edges of screen fonts

Check this box to use ClearType which will smooth edges of screen fonts and may improve readability on some displays. Depending on your platform, you may need to enable ClearType font smoothing at the system display level.

**Note:** This feature applies only to systems using ClearType fonts.

## Window left margin

Enter the left margin of the SecureCRT application window in pixels.

#### **Maximum columns**

Enter the maximum columns to be displayed in the SecureCRT application window.

## **Chat window history**

Enter the number of recently used chat window commands that you want SecureCRT to remember. In your chat window, you can cycle

through these commands using your UP and DOWN arrow keys.

**Note:** To disable the chat window history, enter a zero (0) in this entry box.

### Status bar group

This group includes options that deal with the <u>status bar</u>. To turn off any of these options, clear the associated check box. The available options are as follows:

**Show cursor position** 

Show window size

**Show elapsed time** 

**Never show seconds** 

## **Accessibility group**

This group includes options that deal with accessibility issues.

#### Use trackable insertion caret

For users with accessibility issues, SecureCRT supports the use of a standard insertion caret. Check this option to enable SecureCRT to work with screen access technology for the blind (e.g., screen reading and screen magnification programs) that track the position of the caret.

**Note:** Turning on the **Use trackable insertion caret** option will cause the loss of control over some <u>session</u> cursor options: the cursor color will always be the inversion of the screen color, and the user will not be able to turn off cursor blinking.



## Terminal / Appearance / ANSI Color

The Terminal/Appearance/ANSI Color category of the Global Options dialog allows you to customize the ANSI colors displayed in your SecureCRT application window.

Remote applications may send ANSI color escape sequences to specify that various portions of that application's display appears in one of 16 colors. Eight of the escape sequences correspond to what are called "normal" colors and the other eight typically correspond to "bold" versions of those colors. It is difficult to classify or label these colors beyond "normal and "bold" because an application may use any of the colors for whatever it wants.

Any application that you connect to with SecureCRT may use any ANSI color escape sequence (and its associated color) for its own purposes. While it is possible to configure SecureCRT globally to display 16 specific colors in response to ANSI color escape sequences, two different applications may specify different escape sequences for similar features. For example, when displaying filenames, one application may use the escape sequence that defaults to blue and another application may use the escape sequence that defaults to cyan.

Note: ANSI colors can be enabled or disabled for a specific session from the <u>Terminal/Emulation</u> category of the <u>Session Options</u> dialog.

## Normal colors group

This area displays the colors that are associated with the eight "normal" ANSI color escape seguences.

## **Bold colors group**

This area displays the colors that are associated with the eight "bold" ANSI color escape sequences.

## **Customizing ANSI Colors**

To customize ANSI color selections, click on the color block that you want to change and select a new color from the **Color** dialog. Pressing the **Default** button returns the color blocks to their original settings.



## 🗺 Terminal / Appearance / Advanced

The Terminal/Appearance/Advanced category of the Global Options dialog allows you to control the color scheme for your application window, and to control the transparency/opacity of the window.

## Color schemes group

This group allows you to create color schemes used in your SecureCRT application window. By using the **Edit...** and **New...** buttons, you can modify existing schemes or create new ones.

Tip: Using different color schemes for different session windows makes quick identification easier. You can change color schemes for individual sessions in the Terminal/Appearance Category of the Session Options dialog.

## Window transparency group

This group allows you to control the transparency/opacity of your application window.

#### **Active window**

This option determines how transparent or opaque the SecureCRT window is when it is the active window on the desktop. The minimum allowed value for the active window is 15 so that the active window can never be completely transparent.

#### **Inactive window**

This option determines how transparent or opaque the SecureCRT window is when it is NOT the active window on the desktop.

Note: The SecureCRT window will become completely transparent (invisible) if this option is set to 0 (all the way to the left).



## 随 Terminal / Tabs/Tiling

The Terminal/Tabs/Tiling category of the Global Options dialog allows you to configure the look and behavior of tabs and tiled session windows in SecureCRT. Some options only apply to tabs. See below for information about which options apply to both tabs and tiled sessions.

## **Appearance group**

This group allows you to set the appearance of the tabs in SecureCRT.

#### Connected

Clicking on this tab when the **Show status indicators** option is set to Background colors or Color rectangles, allows you to choose the color displayed for connected sessions.

#### Disconnected

Clicking on this tab when the **Show status indicators** option is set to Background colors or Color rectangles, allows you to choose the color displayed for disconnected sessions.

## **New Input**

Clicking on this tab when the **Show status indicators** option is set to Background colors or Color rectangles, allows you to choose the color displayed for sessions that have received new input.

## Inactivity

Clicking on this tab when the **Show status indicators** option is set to Background colors or Color rectangles, allows you to choose the color displayed for sessions that have been inactive for longer than the number of seconds specified below in the **Show inactivity status** after option.

#### Locked

Clicking on this tab when the **Show status indicators** option is set to Background colors or Color rectangles, allows you to choose the color displayed for sessions that have been locked.

#### Show status indicators

This option allows you to select how status indicators are displayed on the tabs: icons, background colors, or colored rectangles. Icons are always shown when sessions are tiled.

## Show inactivity status after *n* seconds

Enter the number of seconds after which the session will be considered inactive. The range is 1 to 999 seconds; the default is 5. This option is also used by tiled sessions.

#### Minimum tab width

Enter the minimum width of your tabs. Since SecureCRT will try to shrink multiple tabs to fit in one window, this option can be useful to keep your tabs readable. This option can be set from 20 to 300 pixels.

**Note:** If your tab name is too long for the space available, SecureCRT will truncate the name; however, you can view the full name in a tooltip by hovering your pointer over the tab name.

## **Options group**

This group allows you to set the behavior of tabs in your system.

#### **Double-click action**

This option allows you to choose the action taken when you doubleclick on a tab. The actions include the following:

- Close Tab Disconnects the tab's session and closes the tab.
- Clone Tab Creates a new session that is identical to the tab's session and opens it in a new tab.
- Clone Tab in New Window Creates a new session that is identical to the tab's session and opens it in a new window.
- Send Tab to New Window Removes the tab from the current window and moves it into a new window.
- Reconnect Tab Reconnects the tab's session if it is disconnected.

Note: For tiled sessions, double-clicking maximizes the session.

#### Tab location

This option allows you to choose the placement of the tabs, either at the top or the bottom of the window.

## **Confirm when closing multiple tabs**

Select this option to have SecureCRT alert you when you are closing more than one tab (for example, if you close a window containing more than one open session). If the option <a href="Show confirm disconnect dialog">Show confirm disconnect dialog</a> is checked, SecureCRT will always alert you if one or more sessions is open. This option is also used by tiled sessions.

## Only show tabs when there are more than one

Select this option to only have tabs displayed when there are multiple tabs.

## CTRL+TAB switches to most recently used tab

Instructs SecureCRT to apply this operation to the stated keystrokes. This option is also used by tiled sessions.

### **Reuse disconnected tabs**

When opening a new session, this option instructs SecureCRT to reuse the currently selected tab if it is disconnected. This option is enabled by default. This option is also used by tiled sessions.



## 随 Terminal / Printing

The Terminal/Printing category of the Global Options dialog allows you to set up your printer configuration.

## **Margins group**

This group allows you to set the margins on your printed pages. Margins are measured in inches. By default, all print margins are set at 0.5 inches.

**Note:** These settings may also be configured on a per session basis. For more information, see the Terminal/Printing Category of the Session Options dialog.

## **Printer fonts group**

This group displays the font currently being used for printed pages and also allows you to choose a different font.

#### Choose...

Press this button to open the **Font** dialog where you can choose the font you want to use for printed pages.

**Note:** This setting may also be configured on a per session basis. For more information, see the Terminal/Printing Category of the Session options dialog.

## **Print Setup...**

Press this button to open the Windows **Print Setup** dialog where you can modify your printer configuration.

## Pass-through printing group

Pass-through printing in SecureCRT allows the use of special print control escape sequences that turn on/off a printer control that sends characters directly to the printer. Depending on the sequence received, SecureCRT may bypass the terminal display and send the data directly to the printer.

## Disable pass-through printing

Check this option to disable pass-through printing and cause SecureCRT to ignore all printer control escape sequences it receives:

Note: This setting may also be configured on a per session basis. For more information, see the Terminal/Printing Category of the Session Options

dialog.

## **Buffer pass-through printing**

Check this option to buffer pass-through printing. This will avert the issue of some applications that frequently toggle pass-through printing mode on and off, resulting in printed pages with only one line or pages that are otherwise incomplete. This option buffers all pass-through printing data and does not send it to the printer until you instruct SecureCRT to print the contents of the pass-through printing buffer.

To print the contents of the pass-through printing buffer:

- 1. Open the **File** menu and select **Print**.
- 2. In the **Print** menu, select **Eject Page**.

#### Use raw mode

Check this option to allow the use of a network printer when using pass-through printing.

Raw mode pass-through printing allows you to send data to a named printer as defined in Windows. In addition, raw mode instructs Windows to allow the data to be sent to the named printer without being interpreted (i.e. by the printer driver). If your host or application sends embedded printer commands (e.g., change font, etc.), enabling raw mode pass-through printing will allow these commands to be sent directly to the printer. If raw mode pass-through printing is enabled, SecureCRT will not interpret any embedded printer commands.

**Note:** This setting may also be configured on a per session basis. For more information, see the <u>Terminal/Printing</u> category of the <u>Session</u> <u>Options</u> dialog.

## Direct pass-through printing to port n

Check this option to print directly to the COM or LPT port on the local machine to which a printer is attached.

Direct pass-through printing to a port essentially bypasses the Windows operating system, providing SecureCRT with direct communication to the local printer. If direct pass-through printing is enabled, print jobs are not spooled but are sent directly to the printer. The port to which the printer is attached will not be accessible to other applications until the current print job has been completed. For these reasons, it is suggested that you use raw mode pass-through printing rather than direct pass-through printing to port (see **Use raw mode** 

above).

**Note:** This setting may also be configured on a per session basis. For more information, see the <u>Terminal/Printing</u> category of the <u>Session</u> <u>Options</u> dialog.

## Settings...

Click on this button to open the <u>Direct Pass-Through Printing Settings</u> dialog, which allows you to configure the serial device settings for pass-through printing.

#### Other

#### Force black on white

Check this option to force your printer to print as though your session's foreground color is set to black and your background color is set to white, regardless of your session's actual color settings. This option is off by default.

## **Enable print screen (PrtScn) key**

When this option is unchecked, SecureCRT will ignore the print screen key being pressed. This option is off by default.

## Terminal / Web Browser

The **Terminal/Web Browser** category of the **Global Options** dialog allows you to choose your default web browser and **Telnet** application.

## Web browser group

With SecureCRT, you can open a URL by right-clicking on the URL text and selecting Open URL from the pop-up menu. This group allows you to specify which web browser SecureCRT should use for this operation.

## Use registry setting for web browser

By default, this option is checked and SecureCRT will open a URL using the default web browser for Windows as defined in the Windows registry. To specify a web browser other than the default browser for Windows, clear this option and enter the full path to the web browser that you want to use or use the **Browse** button to select the browser application...

## **Use Dynamic Data Exchange (DDE)**

Select this option if you want a new browser window created each time you open a URL with SecureCRT. By default, if the designated web browser is already running, the URL will not be opened in a new browser window. Instead, the running browser application will be brought into focus, and the URL will be opened in that window.

To open a URL with SecureCRT, right-click on the URL text and select Open URL from the pop-up menu. If you have trouble opening a URL or if you want to open a root folder of a URL follow these steps:

- 1. In the SecureCRT window, select all or part of the URL.
- 2. Right-click on the selected text.
- 3. Click on the **Open Selection as URL** item in the pop-up menu.

For example, suppose that your SecureCRT window displayed the following URL:

http://www.vandyke.com/products/securecrt/index.html You could open the URL www.vandyke.com by selecting just that text and using the **Open Selection as URL** command.

#### **Default URL handler**

## Make SecureCRT your default Telnet application

Click on this button to make SecureCRT the default Telnet application rather than using the Windows Telnet application when it opens a URL of the form:

telnet://...

## Make SecureCRT your default SSH1 application

Click on this button to make SecureCRT the default SSH1 application so that SecureCRT is used when a URL of the form below is opened.

ssh1://...

## Make SecureCRT your default SSH2 application

Click on this button to make SecureCRT the default SSH2 application so that SecureCRT is used when a URL of the form below is opened.

ssh2://...



## 随 Terminal / Advanced

The Terminal/Advanced category of the Global Options dialog allows you to configure SecureCRT advanced terminal emulation-related options.

## **Options group**

This group contains various SecureCRT options.

## **Enable** execute escape sequence

This option allows you to enable and disable the ability to send escape sequences.

#### Disable resize

Check this option to globally disallow the resizing of session windows by means of the resize handles. Note that even though you may change the setting of this option globally, it is possible to make exceptions for one or more sessions by setting this option on a per session basis. Resizing the SecureCRT window can affect the number of rows and columns (see the Size group of the Session Options dialog, Terminal/Emulation category, for more information). By default, the **Disable resize** option is not selected.

Note: Full screen mode is not allowed when the Disable resize feature is set.

## Connect to multiple sessions sequentially

When multiple sessions are connecting, check this option to have the sessions connect one after the other instead of all at once.

#### Custom menu and toolbar file

This entry box displays and allows you to change the SecureCRT menu and toolbar configuration file. This file defines the appearance of the SecureCRT main window menus and toolbar.

# File Transfer

This category only appears if VanDyke Software's SecureFX® 3.0 (or newer) is also installed.

The **File Transfer** category of the <u>Global Options</u> dialog allows you to control attributes of transfers and FTP connections.

# **Transfer options group**

This group contains options that define your file transfer parameters.

# **Default transfer type**

Choose the default file transfer type for undefined file types.

- ASCII Select this mode to transfer text files.
- **Binary** Select this mode to transfer program files, or other non-text files such as ZIP files.
- Prompt for type Select this option if you want to be prompted for the transfer type each time a file of the selected type is downloaded.

# **Action for existing files**

Choose the default overwrite action when files in a target directory already exist. The choices are:

- Confirm overwrite Presents a dialog confirming whether or not to overwrite files. The confirmation dialog allows you to confirm overwrites individually, or to choose the action for all files in the transfer with "Yes to All" or "No to All".
- Always overwrite Always overwrites existing files in the destination directory.
- Never overwrite Never overwrites existing files in the destination directory.
- Overwrite if older Overwrite existing files whose timestamp is older than the source file.

# **Preferences group**

This group contains options that define your general preferences.

# Prompt to resume interrupted transfer

Select whether or not to be prompted to resume interrupted transfers when starting a session. If this option is not selected, SecureFX will

not attempt to resume interrupted transfers at startup. This option is enabled by default.

#### Close window on disconnect

Choose whether or not to close session windows when a session is disconnected. This option is disabled by default.

# **Restore last session directory**

Choose if you want to restore the last remote directory that was visited when starting a session. This option is disabled by default.

# **Connection retry options group**

This group contains options that allow you to create a retry strategy.

# **Maximum number of attempts**

Choose the maximum number of logon retry attempts that will be performed after a failed logon attempts.

# **Delay between attempts**

Choose the number of seconds to delay between retry attempts.

# Remote edit options group

This group contains options that allow you to define actions to be taken after editing a file.

#### When the editor exits

Choose the action to be taken when the editor exits after a file has been modified.

- Prompt asks the user to enter an action.
- Always upload always uploads the edited file in the destination directory.
- Never upload never uploads a file whether or not it has been modified.

# Upload when the file is saved in the editor

Check this box to execute the action selected above when a file has been modified and saved. SecureFX determines when to take action by checking the timestamp on the file being edited to see if the file has been modified before the editor exits. If the file has been modified, SecureFX implements the action defined by the above option.

**Note:** If you are using an editor, such as MS Word, that locks files while they are open, SecureFX cannot upload the updated file while

the editor is running. You will need to close the file or exit the editor before the file can be uploaded.



# File Transfer / Options

This category only appears if VanDyke Software's SecureFX® 3.0 (or newer) is also installed.

The File Transfer/Options category of the Global Options dialog allows you to control attributes of transfers and of FTP and SFTP connections.

# **Local directories group**

#### **Local window**

Enter the path of a local directory that the local window should display on startup. Click on the browser button to select the directory.

# **Download directory**

Enter the path of a local directory that is to be used as the destination folder when the **Download** command is used and the **Download** directory field in the Session Options/File Transfer dialog is left blank. Click on the browser button to select the directory.

# **Cache directory**

Choose the directory where SecureFX will place its temporary files. Click on the browser button to select the directory.

# **Anonymous login group**

#### **Password**

Enter the anonymous password (usually an email address) that you want to use as the default when you are creating anonymous login sessions. Note that if you do not fill in this field, SecureFX will attempt to fill in anonymous password fields with a password of the form: username@.

**Note:** Anonymous login is only applicable to FTP sessions.

# **Directory tree options group**

# Disable directory tree detection

This option is unchecked by default. It sets the default behavior for the identical Session Options/File Transfer Option. Checking this option (in Session Options / File Transfer Category) will disable the directory tree for the current session.

# **Transfer options group**

# Verify file status on retrieve

This option is checked by default. It causes a LIST command to be sent prior to retrieving a file from the server in order to obtain the most current statistics (i.e., size and date) for the file.

# **Maximum number of parallel transfers**

This option allows you to set the maximum number of transfer operations that can be performed simultaneously, giving you control over your bandwidth usage. This value only applies to the transfer queue and the synchronize view (i.e., this does not apply to the transfer window that appears when the queue is not enabled). Note that specifying a large number could cause CPU and/or network bandwidth issues.

#### **Bandwidth limit**

This option allows you to control the maximum amount of data that can be sent during an individual transfer operation thus controlling your bandwidth usage. Specifying 0 for this field means that there is no limit (i.e., SecureFX will use as much bandwidth as is available).

# **Directory listing options group**

# Resolve symbolic links

Checking this box causes SecureFX to include the "L" flag when listing files and directories.



# File Transfer /Options / Advanced

This category only appears if VanDyke Software's SecureFX® 3.0 (or newer) is also installed.

The File Transfer/Options/Advanced category of the Global Options dialog allows you to access advanced features in SecureFX.

# **Options** group

## Move deleted local files to the Recycle Bin

This option is enabled by default. Disabling this option causes deleted local files to be deleted from the file system.

# **Keep connections alive**

This option is off by default. Enabling this option causes SecureFX to send SFTP/FTP NO-OP commands to the SFTP/FTP server whenever 1 minute or more has passed without sending a normal file operation command. This will usually keep the SFTP/FTP server from detecting that the connection is idle and, consequently, disconnecting the session.

It is not recommended that you leave this feature enabled.

# Never use cached file for remote editing

Enabling this option instructs SecureFX never to use the locally cached copy of a file to perform remote editing.

# Log group

# **Enable log output to file**

Enabling this option causes the SecureFX output to be logged to the specified file.

# Log filename

This is the log file's full path.

# **Permissions group**

This group contains options that allow you to specify file and directory permissions. The permissions should be specified in the appropriate octal representation. For FTP connections, these options only work with UNIX systems.

# **File permissions**

This option allows you to specify the read/write/execute permissions on uploaded files. For example, entering 660 in the entry box gives read and write access permissions to the file owner and the group. If this option is left blank/empty, no attempt will be made to set the permissions and the permissions used on the files will be set according to the user's UMASK settings on the remote system.

This option only affects new files that are uploaded (i.e., it will not change the permissions on files that already exist). The exception to this rule is that when the **Always overwrite** option is chosen for existing files, the permissions will be overwritten even for existing files.

# **Directory permissions**

This option allows you to specify the read/write/execute permissions on created directories. For example, entering 775 in the entry box gives read and write access permissions to the directory owner and the group, and read access to the world. If this option is left blank/empty, no attempt will be made to set the permissions and the permissions used on the directories will be set according to the user's UMASK settings on the remote system.

## **Transfer queue group**

# Suspend transfer of queued items

Check this option to postpone the transfer of queued files and folders. You can start the transfer of one or all items in the transfer queue by right-clicking in the transfer queue window and selecting the desired option.



# 🛅 File Transfer / View

This category only appears if VanDyke Software's SecureFX® 3.0 (or newer) is also installed.

The File Transfer/View category of the Global Options dialog allows you to control the attributes of the local window or of session windows.

# General group

This group contains options that control your general view options.

#### Show file attributes in details view

Select this option to include an **Attributes** column in windows that use the **Details** icon view. In the local window this column will display Windows file attributes. In remote session windows this column displays UNIX-style permissions, owner and group ownership of files. This information is usually only available on UNIX-based FTP servers.

# Display the folder tree

Select this option to display the folder tree view. This default behavior can be temporarily superseded with the use of the SecureFX Folder Tree command on the View menu or with the SecureFX Folder Tree toolbar button.

# **Local window group**

This group contains options that control the local window.

# **Display filenames using Explorer attributes**

Enabling this option forces SecureFX to display filenames with the same alphabetic case as the Windows Explorer. For example, an uppercase filename stored on disk as AUTOEXEC.BAT will normally display in Windows Explorer as Autoexec.bat. By default, this option is disabled, which causes filenames to be displayed using the same alphabetic case in which the name is actually stored on disk.

**Note:** When a file is uploaded to a remote site, the real filename case is used regardless of this setting.

# Do not show hidden or system files

This option affects the local window only and is disabled by default.

Position on bottom/right when tabbed horizontally/vertically This option controls the location of the local tab and the remote tabs. When this option is enabled, the local tab appears at the right or bottom of the window. This option is off by default.

# **Remote window group**

This group contains options that control the remote window.

#### Do not show dot files

Enabling this option prevents filenames beginning with a dot (.) in remote directories from displaying. By default, it is enabled. This option may be temporarily superseded using the **Dot Files** command on the **View** menu. However, the **Do not show dot files** option will take precedence over the **All entries** option in the <u>connection//FTP</u> category of the <u>Session Options</u> dialog.

# Display file times in local time zone

Enabling this option causes the remote directory file times to be displayed in the time zone being used by the local machine rather than the time zone defined for the session. This option may be helpful when visually comparing the file times displayed on the remote machine to the file times displayed in the local window. By default, this option is disabled.

# Display the log view

Select this option to display the session window's log view. This default behavior can be temporarily superseded with the use of the SecureFX **Log View** toolbar button.

# Save session window position

Checking this option instructs SecureFX to store your session window positions when you close the windows and reopen them in the same place.

# Log window buffer size (lines)

This option sets the maximum number of lines that each log window can display. The default value is 500.

# **Transfer window group**

This group contains options that affect your transfer window.

# **Number style**

This option sets the way that numbers are displayed in the transfer window. The default selection is Abbreviated. The available options are:

- Abbreviated Selecting this option will cause numbers to be displayed using an appropriate abbreviated size quantity (i.e., KB, MB, GB). Numbers less than 1 kilobyte are not abbreviated.
- Punctuated Selecting this option will cause numbers to be punctuated using the Regional parameters defined in the system settings (e.g., numbers would be displayed with commas after every three digits on U.S. systems).



This category only appears if VanDyke Software's SecureFX® 3.0 (or newer) is also installed.

The **File Transfer/FTP** category of the <u>Global Options</u> dialog allows you to control the attributes of FTP transfers.

# Receive buffer group

# Use system default size

By default the system default buffer size is used. It is highly recommended that you use the system default buffer size.

#### Size

If you decide not to use the system default buffer size, enter the buffer size here.

# Send buffer group

# Use system default size

By default the system default buffer size is used. It is highly recommended that you use the system default buffer size.

#### Size

If you decide not to use the system default buffer size, enter the buffer size here.

# **Data connection options group**

# **Use outgoing data connections (PASV)**

This option will instruct the remote server to assign a port and listen to it for a data connection being initiated by SecureFX.

**Note:** This feature can be useful if a firewall exists between your client and the server.

This option is on by default. This is used as the default setting for the identical <u>Session Options/File Transfer</u> option.

## Disable direct server-to-server transfers

This option is unchecked by default. Checking this option will force all server-to-server transfers to be handled by establishing indirect data connections through the local host. This option is sometimes needed where a server behaves poorly after receiving a request to establish a

server-to-server connection.

# **Limit local ports used for PORT connections**

This option, when enabled, allows you to specify a range of ports for SecureFX to use when making an FTP or FTP/SSL data connection.



# File Transfer / SSL

This category only appears if VanDyke Software's SecureFX® 3.0 (or newer) is also installed.

The File Transfer/SSL category of the Global Options dialog allows you to choose or create authentication certificates for SSL transfers.

#### **Certificate**

SecureFX supports generating and using self-signed certificates, which allows mutual authentication.

#### **Path**

Enter the path of the certificate you want SecureFX to use when authenticating.

#### Create...

Press this button to configure and generate a new certificate.



# 😼 File Transfer / Sound

This category only appears if VanDyke Software's SecureFX® 3.0 (or newer) is also installed.

The File Transfer/Sound category of the Global Options dialog allows you to choose the sounds used by SecureFX for specified events. SecureFX supports .wav sound files.

You can assign sounds for the following events:

- Connect
- **Unexpected disconnect**
- **Transfer complete**
- **Error**

Use the **Browse** button to locate and select a sound file, and the **Play** button button to test the selected file.



# 👼 File Transfer / File Types

This category only appears if VanDyke Software's SecureFX® 3.0 (or newer) is also installed.

The File Transfer/File Types category of the Global Options dialog allows you to configure the registered file types and their associated actions for remote sessions.

File types can be registered with SecureFX so that specific actions are taken when files are opened in a remote session. For example, you can configure SecureFX to automatically download, then open remote text documents with Notepad when you double-click on them in a remote session. The initial file types used by SecureFX are imported from the settings used by Explorer when SecureFX is installed. You may add, edit, or remove any of the registered file types.

Note: The actions and applications for the registered file types described here apply only to files opened in remote session windows. Each file type also includes information that applies only when opening a remote file, such as transfer type and save-to-disk options. The file types used when opening files in the SecureFX local window are the same file types used by Windows Explorer. (Refer to "File Types" in Windows Explorer Help for more information.)

# **Registered file types**

This window displays the file associations that SecureFX currently uses.

#### New

Opens a blank Edit File Type dialog where you can add and configure a new file type.

#### **Delete**

Deletes the currently selected file type.

#### **Edit**

Opens the **Edit File Type** dialog for the selected file type.

#### Search

You can search for a file type that is associated with a file extension by entering the file extension in the Ext. edit box. Pressing return in this edit box or clicking the **Search** button will cause the file type that is associated with the entered file extension to be displayed and highlighted in the file types list.

# **Reset All**

Click this button to cause all of the file types known by Windows Explorer to be reloaded into SecureFX.

# **Details**

This window displays the detail for the file type selected in the **Registered file types** window.



# 👼 File Transfer / Filters

This category only appears if VanDyke Software's SecureFX® 3.0 (or newer) is also installed.

The File Transfer/Filters category of the Global Options dialog allows you to configure filters to be used in file transfer operations.

#### **Filters**

This window displays the filters that SecureFX can use in synchronize operations.

To create a new filter, click on the Add... button to open a blank Filter **Properties** dialog. To modify and existing filter, select that filter and click on the **Edit...** button to open that filter's **Filter Properties** dialog. To remove an existing filter, select that filter and click on the **Delete** button.

# 5 Firewall

The **Firewall** category of the <u>Global Options</u> dialog allows you to configure one or more named firewalls. SecureCRT supports connections through generic proxy firewalls as well as SOCKS versions 4 and 5 firewall types.

#### **Firewalls**

This window lists the firewalls that you have configured. These firewalls can be applied to specific <u>ssh1</u>, <u>ssh2</u>, <u>Telnet/ssl</u>, and <u>Telnet</u> sessions using the applicable protocol category of the <u>Session Options</u> dialog.

#### Name

Displays the unique identifying name for the firewall. This name will be used in the <u>Session Options</u> dialog to specify a specific firewall for a given session.

## **Type**

Displays the type of firewall to be used.

#### Hostname

Displays the hostname or IP address used by the firewall.

#### **Port**

Displays the port used by the firewall.

To create a new firewall, click on the **Add...** button to open a blank <u>Firewall</u> <u>Properties</u> dialog. To modify an existing firewall, select that firewall and click on the **Edit...** button to open that firewall's **Firewall Properties** dialog. To remove an existing firewall, select that firewall and click on the **Delete** button.



The **SSH2** category of the <u>Global Options</u> dialog allows you to configure your <u>SSH2</u> public-key and Agent settings.

# **Public Key**

# Use identity or certificate file

Selecting this option instructs SecureCRT to use the <u>identity file</u> or <u>PKCS #12</u> <u>file</u> stored on the local system as your method of <u>authentication</u>. For more information on configuring your system to use identity files, see <u>Public-Key Authentication for SSH2</u>.

SecureCRT also supports accessing X.509 certificates through PKCS #11. The locator prefix (e.g., "pkcs11::") and suffix (e.g., "::standard") specify which public-key algorithm to use. To use this feature, enter a string similar to one of the following examples but pointing to your PKCS #11.dll file in the text box:

```
pkcs11::prov=c:\windows\system32\opensc-pkcs11.dll
pkcs11::prov=c:\windows\system32\pkcs11.dll::cert
```

The above examples use the x509v3-sign-rsa algorithm.

```
pkcs11::prov=c:\windows\system32\opensc-
pkcs11.dll::standard
```

The above example uses the x509v3-sign-rsa-sha1 algorithm.

```
pkcs11key::prov=c:\windows\system32\pkcs11.dll
```

The above example uses the ssh-rsa algorithm, which can be used to send the certificate as a raw key.

# **Fingerprint**

This entry box will be filled in automatically when an identity file is entered above.

# **Create Identity File**

Press this button to start the Generate Key wizard and create identity files which contain your public/private key pair.

Note: SecureCRT supports both DSA and RSA key types.

# **Change Passphrase**

Press this button to change the passphrase for your global SSH2

identity file.

**Note:** Passphrases for specific session identity files can be changed using that session's Public Key Properties dialog which is opened by pressing the **Properties** button on the Connection/SSH2 category of the Session Options dialog.

# **Use personal store certificate (CAPI)**

Selecting this option instructs SecureCRT to use X.509 certificates from your Microsoft CAPI personal store as your method of authentication. For more information on X.509 certificates, see <u>using X.509 Certificates</u>.

# **Advanced group**

# Add keys to agent

Check this option to enable the SSH2 Agent. Agents are programs that work in the background gathering information or performing small processing tasks. In SecureCRT, the implemented agent temporarily holds private keys for use with <u>public-key authentication</u> to multiple remote hosts. For more information on agents, see <u>Using the Agent</u>.

# **Enable OpenSSH agent forwarding**

Check this option to use the agent to connect to a remote machine through another remote machine (see <u>Using the Agent</u>). This option can be overridden on a per-session basis from the <u>Session</u>

Options/Connection/SSH2/Advanced Category.

# **Enable deprecated GSSAPI**

Check this option to have SecureCRT first attempt to connect using GSSAPI with MIC and then, if that is not successful, try regular GSSAPI. If this box is not checked, SecureCRT will only try to connect using GSSAPI with MIC.

**Note:** When using Kerberos host and user authentication via GSSAPI, the connection could be vulnerable to a man-in-the-middle attack. Using GSSAPI with MIC eliminates this risk. Although the GSSAPI method has been deprecated, GSSAPI with MIC is not yet widely supported. SecureCRT allows you to attempt to connect using GSSAPI with MIC if it is available on the server.

# Cache session password

When this option is set, passwords will be cached while SecureCRT, SecureFX®, or the Activator is running so that when re-connecting to

that session or connecting in one of the other applications, the password does not have to be re-entered.



The **SSH Host Keys** category of the <u>Global Options</u> dialog allows you to manage your host keys. SecureCRT provides support for viewing, importing, exporting and deleting host keys.

## **Accepted host keys**

This list displays your host keys by **Host** (hostname and IP address) and **Port**. You can also select one of the existing host keys from the list and view, export, or delete it by pressing the appropriate button.

# View Key...

Press this button to view the selected host key in textual format.

# Import Key...

Press this button to import a new host key file.

## **Export Key...**

Press this button to create a text file of the selected host key that is suitable for sending to <u>server</u> administrators and others.

#### **Delete**

Press this button to remove the selected host key from the list and delete the host key file from your system.

# **Host key database location**

This entry box displays and allows you to change the location of the host key database.

# X.509 Host Keys

SecureCRT supports automatic acceptance of X.509 host keys. This means that if the following conditions are met, the host key will be accepted without prompting the user.

Before accepting a certificate, SecureCRT checks to ensure the following conditions:

- 1. The hostname to which you are connecting matches the common name part of the subject name of the certificate.
- 2. The current date is between the start and end validity dates of the certificate.
- 3. The certificate chains to a trusted root certificate (anchor) installed in the Microsoft CAPI store.
- 4. If the Check certificate revocation lists option below has been

checked, SecureCRT checks to ensure that the certificate (and any in its chain) have not been listed as revoked in a certificate revocation list (CRL) installed in the Microsoft CAPI store.

# **Digital certificates group**

# **Check certificate revocation lists**

Check this option to have SecureCRT check the certificate revocation lists before accepting a host key.

# 随 Session Options Dialog

The **Session Options** dialog allows you to create a new session or to modify an existing session. The dialog can be accessed from the main SecureCRT window by clicking on Session Options... in the options menu or by clicking on the **Session Options** button on the toolbar. The Session Options dialog can also be accessed from the Connect dialog by clicking on either the **New Session** button or **Properties** button.

The **Session Options** dialog lists the available categories and displays the options for the category selected. The options change with the selection in the Category window.

IMPORTANT: If you edit a session to which you are currently connected, the settings will revert back to your old settings when you close the session. To permanently change session options, close the session, make the edits, and then reopen the session.

The session option categories displayed will change depending on the **Protocol** selected in the connection category. The available categories and subcategories are as follows:

- Connection
  - **Logon Actions**
  - SSH1 (when SSH1 is the selected protocol)
    - **Advanced**
  - SSH2 (when SSH2 or SFTP is the selected protocol)
    - **SFTP Session**
    - Advanced
  - Telnet (when Telnet is the selected protocol)
  - Telnet/SSL (when Telnet/SSL is the selected protocol)
  - **RLogin** (when RLogin is the selected protocol)
  - **FTP** (when FTP is the selected protocol)
    - SSL Options (when FTP/SSL is the selected protocol)
  - Serial (when Serial is the selected protocol)
  - TAPI (when TAPI is the selected protocol)
    - Advanced

- · Raw (when Raw is the selected protocol)
- · Port Forwarding (when SSH1 or SSH2 is the selected protocol)
  - · Remote/X11 (when SSH2 is the selected protocol)
- <u>Terminal</u>
  - Emulation
    - Modes
    - Emacs
    - Mapped Keys
    - Advanced
  - Appearance
    - ANSI Color
    - Window
  - Log File
  - <u>Printing</u>
    - Advanced
  - X/Y/Zmodem
- File Transfer
  - · FTP/SFTP (when FTP or SFTP is the selected transfer protocol)

# **Connection**

The **Connections** category of the  $\frac{Session\ Options}{Session\ Configure\ a\ Session\ Connection}$  .

The options of the **Connection** category of the **Session Options** dialog are based on the choice of **Protocol**. In other words, if Telnet is selected as the protocol, the options displayed on the dialog will be different than if SSH2 is selected. Below is a list of the protocols and options available in SecureCRT.

The following options are displayed for all protocols.

#### Name

Enter a uniquely identifying name for this session.

**Protocol** (This option appears if only SecureCRT is installed) Select the protocol that you want to use for this session. SecureCRT supports the following protocols: <u>SSH1</u>, <u>SSH2</u>, <u>Telnet</u>, <u>Telnet</u>/SSL, RLogin, Serial (COM), <u>TAPI</u>, and <u>Raw</u>.

**Protocol group** (This group appears if SecureCRT and SecureFX are both installed)

Select the protocol that you want to use for the stated type of session.

**Note:** If FIPS mode is enabled, the following protocols cannot be used: SSH1, Telnet/SSL, and FTP/SSL.

#### **Terminal**

Select the terminal emulation protocol that you want to use for this session. SecureCRT supports the following protocols:  $\underline{SSH1}$ ,  $\underline{SSH2}$ ,  $\underline{Telnet}$ ,  $\underline{Telnet}$ , RLogin, Serial (COM),  $\underline{TAPI}$ , and  $\underline{Raw}$ .

#### File transfer

Select the file transfer protocol that you want to use for this session. SecureFX supports the following protocols: SFTP, FTP, and FTP/SSL.

# **Description group**

Enter a session description to be displayed in the Connect window.



# Connection / Logon Actions

The **Logon Actions** category of the **Session Options** dialog allows you to run scripts to automate logging on to a remote machine and perform other functions.

# Automate logon

Check this option to create a simple script to automate logging on to a remote machine. Use the **Expect** and **Send** fields to enter the character strings that SecureCRT should expect to receive from the remote machine and should send as its response. If you want SecureCRT to send a carriage return before running a script, check the **Send initial** carriage return option.

## Send initial carriage return

Check this option if you want SecureCRT to send a carriage return to the remote machine before starting the automated logon script. Some systems will not display a logon prompt until receiving a carriage return.

# Expect / Send

In the **Expect** entry boxes, enter the character strings that SecureCRT should expect to receive from the remote machine in the order that they should be received. In the associated **Send** entry boxes, enter the strings that you want SecureCRT to send in response when it receives the **Expect** string. For a list of function key commands that can be included in **Send** strings, see <u>Send String and Function Key Commands</u>.

**Note:** SecureCRT works through the **Expect** strings from top to bottom; in other words, if SecureCRT does not receive the first expected string, it will not send the associated **Send** string or listen for the next expected string.

# **Example script:**

The following is an example script that will instruct SecureCRT to listen for the logon prompt "Logon>" and respond with the username "johnsmith" when the prompt is received. SecureCRT will then listen for "Password>" and respond to it with "way2easy". Finally, SecureCRT will listen for the string "Welcome" and respond with the directory request "dir".

**Expect:** Logon> **Send:** johnsmith **Expect:** Password> **Send:** way2easy

Expect: Welcome Send: dir

#### Hide

Check this box to have the **Send** string shown as asterisks. This option lets you avoid displaying confidential information such as a password.

# **Logon script**

Check this option to select a script file that you want SecureCRT to run when it makes a  $_{\text{connection}}$  .

**Note:** If the **Automate logon** option is also selected, it will be executed first.

#### Remote command

Enter the command that you want SecureCRT to instruct the remote machine to run when a <u>connection</u> is made.

**Note:** The remote command is the only command that the session will run. Once the command exits, the session will disconnect.

# Display logon prompts in terminal window

Check this option to have SecureCRT display logon prompts in the terminal window.



# Connection / SSH1

The SSH1 category of the Session Options dialog allows you to configure your SSH1 connection . The SSH1 category only appears when you have selected SSH1 as your <u>Protocol</u>.

## SSH1 Overview

Port forwarding is another feature based on SSH security. See Port Forwarding with SSH to learn more about encrypting connections for other applications (such as IMAP) that are not secure by default.

SSH1 connection settings include hostname, port, username, authentication, and key exchange.

#### Hostname

The hostname or IP address of the remote machine that provides the SSH1 service.

#### **Port**

The port number of the SSH1 service on the remote machine. For SSH1, the default port is 22.

#### **Firewall**

If your connection involves a firewall, select your firewall from the list of firewalls that have been configured in the Global Options/Firewall dialog.

**Note:** You can also select an SSH2 session to be used as a firewall. When a session is specified as a firewall, the firewall session will be connected first.

#### Username

The username used to log on to the remote machine.

#### Authentication

SecureCRT supports three types of authentication for connecting to SSH1 servers: password, RSA, and TIS.

**Password** authentication transmits the user's password to the server to authenticate the connection. The transmitted password is protected from network eavesdropping, due to the cipher encryption of the data channel. For this reason, some SSH1 servers reject the use of password authentication if the cipher is set to None.

RSA authentication uses a public/private key pair to authenticate the connection. The general mechanism behind RSA authentication is that the SSH1 server "challenges" the client to decrypt a message encoded using the user's public key stored on the server. Upon connecting, the SSH1 server generates a random value, encrypts the value using the user's public key and sends the encrypted challenge to the client. The client authenticates the connection by successfully decrypting the challenge using the user's private key. The security of the mechanism requires that no one but the owner have access to the private key. The private key is stored locally in an identity file. The first time you connect to an SSH1 server using RSA authentication, SecureCRT will prompt you for the location of this file. Also, prior to using RSA authentication, the public key must be made available to the SSH1 server.

**TIS** firewall authentication uses the TIS firewall server to provide a challenge phrase / response combination. SSH1 servers must be configured to offer TIS authentication.

# Key exchange group

Key exchange is part of establishing trust between a client and a Secure Shell server. SecureCRT supports several algorithms for doing key exchange and will attempt to use them in the order that you specify.

**Diffie-Hellman** key exchange algorithms are common cryptographic protocols which are supported by Secure Shell servers.

**GSSAPI** key exchange algorithms can be used to connect to SSH2 servers that support GSSAPI. When a GSSAPI key exchange algorithm is specified and the server supports it, further authentication is not needed if you already have GSSAPI credentials (e.g., by logging onto a Windows machine that is part of an Active Directory domain).

If a GSSAPI key exchange algorithm is specified and is not supported by the server, there could be a delay during connection because the server is waiting to time out. To prevent this delay, uncheck the **GSSAPI** key exchange algorithms.



# Connection / SSH1 / Advanced

The **SSH1/Advanced** category of the <u>Session Options</u> dialog allows you to further configure your <u>SSH1</u> <u>connection</u> . The **SSH1** category only appears when you have selected SSH1 as your Protocol.

# Cipher

The encryption algorithm to use for data privacy. The default cipher is 3DES. DES, RC4 and Blowfish ciphers are also available for use with the SSH1 protocol. You may also set the cipher to None, which will effectively render your connection insecure. No encryption will occur if the cipher is set to None, and all data will be transmitted in plain text.

# WARNING: Setting cipher to None causes the data channel to be left unencrypted and offers no security.

Cipher Speed Information -- The DES and 3DES ciphers are very CPU intensive (slow). The RC4 and Blowfish ciphers are considerably less CPU intensive (faster).

# **MAC (Message Authentication Code) (SSH2 Only)**

The SSH2 protocol provides increased security over SSH1 by means of a MAC (Message Authentication Codes) which ensures data integrity. Although specifying a MAC is optional, it is highly recommended that a MAC be specified in order to ensure data integrity.

# WARNING: Data integrity cannot be ensured if MAC is set to None.

SecureCRT will attempt to connect using the first selected MAC in the MAC list and then, if not successful, work down the list trying each selected MAC. To reorder the list, select the MAC that you want to reposition and use the buttons to the right of the list to move the MAC up or down.

# **Options** group

# Compression

Enable this option to have SecureCRT use compression when transferring data.

# **Enable OpenSSH agent forwarding**

Check this option to use the agent to connect to a remote machine through another remote machine (see Using the Agent).

# Level

This option controls the level of compression. The compression level ranges from 0 (minimal compression) to 9 (maximum compression). A higher level means better compression but slower performance. The default value is 5.



# Connection / SSH2

The SSH2 category of the Session Options dialog allows you to configure your SSH2 connection . The SSH2 category only appears when you have selected SSH2 or SFTP as your Protocol.

#### SSH2 Overview

SSH2 provides secure communication over an unsecure channel by encrypting the data channel using the cipher algorithm selected for the session by the user. The cipher selected must also be supported by the destination SSH2 server (an error will be reported during a connection attempt if the chosen cipher is not supported by the server). A cipher is used to encrypt network traffic between the local machine and the SSH2 server, thus providing data privacy.

Port forwarding is another feature based on SSH security. See Port Forwarding with **SSH** to learn more about encrypting connections for other applications (such as IMAP) that are not secure by default.

SSH2 connection settings include hostname, port, username, authentication, and key exchange.

#### Hostname

The hostname or IP address of the remote machine that provides the SSH2 service.

#### **Port**

The port number of the SSH2 service on the remote machine. For SSH2, the default port is 22.

#### **Firewall**

If your connection involves a firewall, select your firewall from the list of firewalls that have been configured in the Global Options/Firewall dialog.

**Note:** You can also select an SSH2 session to be used as a firewall. When a session is specified as a firewall, the firewall session will be connected first.

#### Username

The username used to log on to the remote machine.

# **Authentication group**

SecureCRT supports several authentication methods for connecting to SSH2 servers, and will attempt to connect using them in the order that you specify.

**Password** authentication transmits the user's password to the server to authenticate the connection. The transmitted password is protected from network eavesdropping, due to the cipher encryption of the data channel.

**Note**: If the remote machine supports both the SSH2 protocol and changing passwords at the protocol level, the password for an SSH2 session can be changed from the **Password Properties** dialog, which is accessed by selecting **Password** in the **Authentication** group and clicking on the **Properties** button.

**PublicKey** authentication uses a public/private key pair to authenticate the connection. During the authentication process, the client and the server negotiate a public key to use for the connection. Once a public key has been determined, the client uses the corresponding private key to perform a signature operation over a unique connection identifier. This signature is then sent to the server for verification. If verification is successful, the client is given permission to connect to the server. The security of the mechanism requires that no one but the owner have access to the private key. The private key is stored locally in an identity file. Also, prior to using public-key authentication, the public key must be made available to the SSH2 server. For more information on generating private-public key pairs, See Public-Key Authentication for SSH2.

**Keyboard Interactive** authentication allows you to use the keyboard to respond to challenges put forth by the server.

**GSSAPI** (Generic Security Services Application Program Interface) is a generic API for performing client/server authentication. GSSAPI allows SecureCRT to authenticate with a server without knowing anything about the specific authentication mechanism in use. For more information about using GSSAPI, see the **GSSAPI Properties** dialog.

SecureCRT also supports the use of  $\underline{x.509}$  certificates.

# Key exchange group

Key exchange is part of establishing trust between a client and a Secure

Shell server. SecureCRT supports several algorithms for doing key exchange and will attempt to use them in the order that you specify.

**Diffie-Hellman** key exchange algorithms are common cryptographic protocols which are supported by Secure Shell servers.

**GSSAPI** key exchange algorithms can be used to connect to SSH2 servers that support GSSAPI. When a GSSAPI key exchange algorithm is specified and the server supports it, further authentication is not needed if you already have GSSAPI credentials (e.g., by logging onto a Windows machine that is part of an Active Directory domain).

If a GSSAPI key exchange algorithm is specified and is not supported by the server, there could be a delay during connection because the server is waiting to time out. To prevent this delay, uncheck the **GSSAPI** key exchange algorithms.



# Connection / SSH2 / Advanced

The **SSH2/Advanced** category of the <u>Session Options</u> dialog allows you to further configure your <u>SSH2</u> <u>connection</u> . The **SSH2** category only appears when you have selected SSH2 or SFTP as your Protocol.

### **Cipher**

The encryption algorithm to use for data privacy. The ciphers provided for use with the SSH2 protocol in SecureCRT are AES, Twofish, Blowfish, 3DES, and RC4. You may also set the cipher to None, which will effectively render your connection insecure. No encryption will occur if the cipher is set to None, and all data will be transmitted in plain text.

# WARNING: Setting cipher to None causes the data channel to be left unencrypted and offers no security.

SecureCRT will attempt to connect using the first selected cipher in the Cipher list and then, if not successful, work down the list trying each selected cipher. To reorder the list, select the cipher that you want to reposition and use the buttons to the right of the list to move the cipher up or down.

Cipher Speed Information -- The 3DES cipher is very CPU intensive (slow). The AES and Twofish ciphers are considerably less CPU intensive (faster) than 3DES.

# **MAC (Message Authentication Code)**

The SSH2 protocol provides increased security over SSH1 by means of a MAC (Message Authentication Codes) which ensures data integrity. Although specifying a MAC is optional, it is highly recommended that a MAC be specified in order to ensure data integrity. The MACs provided for use with the SSH2 protocol in SecureCRT are SHA1, MD5, and UMAC.

# WARNING: Data integrity cannot be ensured if MAC is set to None.

SecureCRT will attempt to connect using the first selected MAC in the MAC list and then, if not successful, work down the list trying each selected MAC. To reorder the list, select the MAC that you want to reposition and use the buttons to the right of the list to move the MAC up or down.

# **Options group**

#### Compression

Select the type of compression that you want for this SSH2 session .

#### Level

This option controls the level of compression. The compression level ranges from 0 (minimal compression) to 9 (maximum compression). A higher level means better compression but slower performance. The default value is 5.

### **Enable OpenSSH agent forwarding**

Check this option to use the agent to connect to a remote machine through another remote machine (see <u>Using the Agent</u>). This is a tri-state option; If this option is set to the tri-state value (square), the setting from the <u>Global Options/SSH2</u> category will be used.



# Connection / SSH2 / SFTP Session

The SSH2/SFTP Session category of the <u>Session Options</u> dialog allows you to specify the directories you want to connect to upon starting the session.

#### **Initial Directories**

### **Local directory**

Enter the path of the local directory that you want the local window to display on startup of the session. This is the default folder from which you want to transfer files to the remote machine. Click on the browser button to use a directory browser to select the directory.

#### **Remote directory**

Enter the full path of the directory on the remote host that you want to be the current directory upon connecting to the remote host. If the this is not specified, the default or home directory for the username on the remote system will be used.



# Connection / Telnet

The **Telnet** category of the <u>Session Options</u> dialog allows you to further configure your Telnet connection. The **Telnet** category only appears when you have selected Telnet as your **Protocol**.

#### Hostname

The hostname or IP address of the remote host.

#### **Port**

The port number of a service on a remote machine. For Telnet, the default port is 23.

#### **Firewall**

If your connection involves a firewall, select your firewall from the list of firewalls that have been configured in the Global Options/Firewall dialog.

**Note:** You can also select an SSH2 session to be used as a firewall. When a session is specified as a firewall, the firewall session will be connected first.

# **Advanced group**

#### Will LFLOW

Select this option to enable local flow control when connecting to Telnet sites that support LFLOW. This option is selected by default.

#### **Enable NAWS**

Select this option to enable the Negotiate About Window Size (NAWS) feature. NAWS allows a Telnet client and a Telnet server to negotiate resizing the terminal window.

**Note:** NAWS may not be applicable to all systems.

#### Force character at a time mode

Select this option to force SecureCRT to be in "character at a time" Telnet mode. This option may be required in rare cases where Telnet negotiation between the client and server does not occur correctly.

WARNING: For most users, this option should <u>not</u> be selected.

# Send SGA (port 23 only)

Select this option to initiate Telnet negotiation by sending the Telnet SGA command. Most Telnet servers expect this from the Telnet client.

# **WARNING:** For most users, this option <u>should</u> be selected.

### Send SGA

Select this option to send the DO SGA command immediately after connecting.

# **Enable NTLM authentication**

Select this option to enable  $\underline{\text{NT LAN Manager (NTLM)}}$  authentication. This option is enabled by default.



# Connection / Telnet/SSL

The **Telnet/SSL** category of the <u>Session Options</u> dialog allows you to further configure your **Telnet** connection over the cryptographic communications protocol SSL . The **Telnet/SSL** category only appears when you have selected Telnet/SSL as your Protocol.

#### Hostname

The hostname or IP address of the remote host.

#### **Port**

The port number of a service on a remote machine. For Telnet/SSL, the default port is 992.

#### **Firewall**

If your connection involves a firewall, select your firewall from the list of firewalls that have been configured in the Global Options/Firewall dialog.

**Note:** You can also select an SSH2 session to be used as a firewall. When a session is specified as a firewall, the firewall session will be connected first.

### **Advanced group**

#### Will LFLOW

Select this option to enable local flow control when connecting to Telnet sites that support LFLOW. This option is selected by default.

#### **Enable NAWS**

Select this option to enable the Negotiate About Window Size (NAWS) feature. NAWS allows a Telnet client and a Telnet server to negotiate resizing the terminal window.

**Note:** NAWS may not be applicable to all systems.

#### Force character at a time mode

Select this option to force SecureCRT to be in "character at a time" Telnet mode. This option may be required in rare cases where Telnet negotiation between the <u>client</u> and <u>server</u> does not occur correctly.

WARNING: For most users, this option should <u>not</u> be selected.

# Send SGA (port 992 only)

Select this option to initiate Telnet negotiation by sending the Telnet

SGA command. Most Telnet servers expect this from the Telnet client.

**WARNING:** For most users, this option <u>should</u> be selected.

#### Send SGA

Select this option to send the DO SGA command immediately after connecting.

#### **Enable NTLM authentication**

Select this option to enable NT LAN Manager (NTLM) authentication. This option is enabled by default.

### **SSL** group

### **Disable certificate validation**

Select this option to discontinue checking the validity of X.509 certificates.

WARNING: While selecting this option may speed up your connection time, it may leave you vulnerable to invalid certificates.



# Connection / RLogin

The RLogin category of the Session Options dialog allows you to further configure your RLogin connection . The **RLogin** category only appears when you have selected RLogin as your Protocol.

RLogin connection settings include hostname or IP address, and username.

#### Hostname

Enter the hostname or IP address of the remote host.

#### Username

Enter the username used to log onto the remote machine.



# Connection / Serial

The **Serial** category of the <u>session Options</u> dialog allows you to further configure your Serial connection . The **Serial** category only appears when you have selected Serial as your Protocol.

The Serial protocol is used to connect to another computer with your computer's COM port or with a modem attached to your computer's COM port.

#### **Port**

Enter the Serial port (COM1, COM2, etc.) used by the session.

#### **Baud rate**

Select the baud rate for the device. The default value is 38400. The baud rate can be changed while a session is connected.

#### **Data bits**

Select the number of communication data bits. The default value is 8.

### **Parity**

Choose the parity setting supported by the connected device. The default setting is None.

### Stop bits

Choose number of stop bits sent after each character. The default value is 1.

# Flow control group

#### DTR/DSR

Select this option to use the DTR/DSR (data-terminal-ready/data-setready) signals with the connected device. By default, this is not selected.

#### RTS/CTS

Select this option to use RTS/CTS (request-to-send/clear-to-send) hardware flow control. By default, this option is selected.

#### XON/XOFF

Select this option to use XON/XOFF software flow control. By default, this option is not selected.

# Serial break length

Select the length of breaks in Serial data traffic. For more information about breaks, please see <u>Sending a Serial Break</u>.

# Connection / TAPI

The **TAPI** category of the <u>session Options</u> dialog allows you to configure your TAPI connections and redial strategy for busy numbers. The **TAPI** category only appears when you have selected TAPI as your <u>Protocol</u>.

You can use the TAPI protocol to "share" your computer's modem with other applications. For example, if you have a fax application that is TAPI compliant, you will be able to have both SecureCRT and your fax software open at the same time. Although both programs cannot use the modem at exactly the same time, TAPI compliant programs can be running at the same time, using the modem to dial out or receive calls on an "as needed" basis.

When using TAPI, the following information must be entered:

### Dial using

Select the modem you would like to use from the **Dial using** list of modems. If you have a modem installed, you can also configure it while in the <u>session Options</u> dialog by selecting the modem and clicking on the **Configure** button.

### **Country code**

Enter the country code associated with the number you are dialing.

#### Area code

Enter the area code for the number you are dialing.

#### Phone number

Enter the phone number that you are dialing. SecureCRT also allows you to specify any alternate phone numbers you wish to dial using the same session .

### **Auto redial group**

# Number of redial attempts

Select the number of redial attempts that SecureCRT will make before giving up on the  $_{\hbox{\scriptsize connection}}$  .

### **Delay between attempts**

Select the number of seconds that SecureCRT will wait between redial attempts.



# Connection / TAPI / Advanced

The **TAPI/Advanced** category of the <u>session Options</u> dialog allows you to enter alternate phone numbers if the remote machine or BBS you are attempting to dial provides you with alternate numbers.

The TAPI/Advanced category only appears when you have selected TAPI as your **Protocol**.

### New phone number

Enter the alternate phone number. Click on the **Add** button to add the number to the **Phone numbers** list below, or click on the **Replace** button to replace a number selected in the list.

#### **Phone numbers**

This list displays the configured phone numbers in the order in which they will be tried. You can change this order by selecting a number and using the **Up** or **Down** buttons to move it in the list.

Move successful number to top of list on connection Check this option to move a number that has made a successful connection to the top of the list.



The **Raw** category of the <u>Session Options</u> dialog allows you to further configure your Raw <u>connection</u>. The **Raw** category only appears when you have selected Raw as your <u>Protocol</u>.

#### Hostname

The hostname or IP address of the remote host.

#### **Port**

The port number of a service on a remote machine.

#### **Firewall**

If your connection involves a firewall, select your firewall from the list of firewalls that have been configured in the Global Options/Firewall dialog.

**Note:** You can also select an SSH2 session to be used as a firewall. When a session is specified as a firewall, the firewall session will be connected first.

# Connection / FTP

This category only appears if VanDyke Software's SecureFX® 3.0 (or newer) is also installed.

The **FTP** category of the <u>session Options</u> dialog allows you to control attributes of FTP connections. The **FTP** category only appears when you have selected FTP as your <u>Protocol</u>.

### **FTP logon group**

#### Hostname

The hostname or IP address of the remote machine that provides the FTP service.

#### **Port**

The port number of the FTP service on the remote machine. For FTP and FTP/SSL (explicit), the default port is 21. For FTP/SSL (implicit), the default port is 990.

#### **Firewall**

If your connection involves a firewall, select your firewall from the list of firewalls that have been configured in the Global Options/Firewall dialog.

**Note:** You can also select an SFTP or SCP session to be used as a firewall. If SecureFX is integrated with SecureCRT, an SSH2 session can also be used as a firewall. When a session is specified as a firewall, the firewall session will be connected first.

# **Anonymous login**

When this option is selected, SecureFX automatically selects anonymous as the username and fills in the password field using the password you supplied in the Global Options/File Transfer/Options dialog. Unselecting this option makes the username and password fields available.

#### Username

The username used to log on to the remote machine.

#### **Password**

The password used to log on to the remote machine.

#### Account

The account name to be used on the remote machine.

### **Data connection options group**

### **Use outgoing data connection (PASV)**

Checking this box forces all data connections to be originated by SecureFX (using the FTP PASV command). This setting is often needed where a firewall will not allow incoming connections from external FTP servers. When this check box is unchecked, SecureFX will have the FTP server create an incoming data connection instead (using the FTP PORT command).

This check box can also be set to a third, indeterminate state. When the check box is in this state, it appears grayed out and will cause SecureFX to use the **Use outgoing data connection setting** in the **Global Options/File Transfer/FTP** dialog.

### **Directory listing options group**

### Resolve symbolic links

Checking this box causes SecureFX to include the "L" flag when listing files and directories.

### **Long format**

Checking this box causes SecureFX to include the "-I" flag when listing files and directories. This requests that the FTP server include all available columns of informational output when performing directory listings. This option is usually only useful when the server is a UNIX system. This option is off by default.

#### All entries

Checking this box causes SecureFX to include the "-a" flag when listing files and directories. This requests that the FTP server include hidden files (including dot files) when performing directory listings. This option is usually only useful when the server is a UNIX system. This option is off by default.

Note: The **Do not show dot files** option from the <u>Global Options/File</u> <u>Transfer/View</u> dialog will take precedence over the **All entries** option.



# Connection / FTP / SSL Options

This category only appears if VanDyke Software's SecureFX® 3.0 (or newer) is also installed.

The FTP/SSL Options category of the <u>Session Options</u> dialog allows you to control attributes of FTP/SSL connections. The FTP/SSL Options category only appears when you have selected FTP/SSL as your **Protocol**.

### **Certificate group**

SecureFX supports generating and using self-signed certificates, which allows mutual authentication.

### Use global certificate

Checking this option instructs SecureFX to use the certificate specified in the Global Options / File Transfer / SSL dialog for this session.

#### Path

If you choose not to use the global certificate for this session, enter the path of the certificate you want SecureFX to use when authenticating.

#### Create...

Press this button to configure and generate a new certificate.

# **Advanced group**

#### Disable certificate validation

Select this option to discontinue checking the validity of X.509 certificates.

WARNING: While selecting this option may speed up your connection time, it may leave you vulnerable to invalid certificates.

# **Use CCC (Clear Command Channel) after authentication**

This option tells SecureFX to transmit data over the control channel in an unencrypted form (i.e., clear text) after authentication. It may be necessary to enable this option when the local client machine is behind a firewall.

### Use unencrypted data channel

This option tells SecureFX to transmit data over the data channel in an unencrypted form (i.e., clear text). The control channel, however, will remain encrypted even when this option has been turned on.

WARNING: The above options should only be used when absolutely necessary because selecting either of the above options will result in data being transmitted as clear text.



# Connection / Port Forwarding

The **Port Forwarding** category of the <u>Session Options</u> dialog allows you to configure secure connections to an SSH1 or SSH2 Server .

# **Overview of Port Forwarding**

Port forwarding is a powerful tool that allows you to secure TCP/IP traffic using SecureCRT's SSH1 and SSH2 protocol support. This means that you can encrypt application data using protocols such as IMAP, POP3 and SMTP. For example, if you receive your email from an Internet Service Provider (ISP), you could encrypt the communication between your workstation running the email client and the ISP's SSH server.

In general, with any port forwarded by SecureCRT for an application, the application needs to be reconfigured to use the localhost or loopback address 127.0.0.1 as its application server address.

### **Locally forwarded connections**

#### Name

Displays the unique identifying name for the connection.

#### **Local Address**

Displays the local port number for the connection.

#### **Remote Host**

Displays the remote machines hostname or IP address and the remote port.

# **Application**

Displays the application to be run over this connection.

To create a new port-forwarded connection, click on the **Add...** button to open a blank Local Port Forwarding Properties dialog. To modify an existing connection, select that connection and click on the Edit... button to open that connection's **Local Port Forwarding Properties** dialog. To remove an existing connection, select that connection and click on the **Delete** button.

# Do not request a shell

Select this option to have SecureCRT not request a command shell when attempting to port forward. This ensures that port forwards will work even

if shell access is denied.

Port Forwarding with SSH
Session Options/Connection
Session Options/Connection Port Forwarding/Remote



# Connection / Port Forwarding / Remote/X11

The **Port Forwarding/Remote/X11** category of the <u>session Options</u> dialog allows you to configure secure connections through an SSH Server to an application that resides on the SSH client -side, and to enable the forwarding of X11 packets.

### **Remotely forwarded connections**

#### Name

Displays the unique identifying name for the connection.

#### **Remote Address**

Displays the port number on the remote machine that is used for this connection.

#### **Local Host**

Displays the hostname or IP address of the local machine and the local port number.

To create a new port-forwarded connection, click on the **Add...** button to open a blank Remote Port Forwarding Properties dialog. To modify an existing connection, select that connection and click on the **Edit...** button to open that connection's **Remote Port Forwarding Properties** dialog. To remove an existing connection, select that connection and click on the **Delete** button.

### X11 forwarding

### Forward X11 packets

Check this option to allow the forwarding of X Windows packets through the SSH session, which makes possible the encryption of the data between the client and server

Note: SecureCRT is not an X Server. The Forward X11 packets option allows SecureCRT to accept X11 data from the remote machine and forwards it to the X server running on the local machine. The local X Server must be running before any X11 sessions can be displayed. If you are using Xhost authority access on the local X11 server, you will need to add the localhost or loopback address 127.0.0.1 to your server's Xhost list.

#### **Enforce X11 authentication**

Check this box to instruct SecureCRT to enforce X11 authentication.

**Note:** It may be helpful to disable this option when switching to a different user before launching an X application. If this option is enabled and the secondary user's .Xauthority file does not match the initial user's file, access to the X server will be denied.



The **Terminal** category of the <u>session Options</u> dialog allows you to configure general <u>session</u> options for the selected session.

#### Audio bell

Check this option to have SecureCRT emit an audible sound when a bell character is received. This option is on by default.

#### Visual bell

Check this button to have SecureCRT flash the session window when a bell character is received. This option is off by default.

#### Close on disconnect

Check this button to have SecureCRT automatically close the session window after disconnecting from the remote machine.

#### Clear on disconnect

Check this button to have SecureCRT automatically clear the session window after disconnecting from the remote machine.

#### **Auto reconnect**

Check this option to automatically reconnect to a remote machine when a connection is terminated.

# **Jump scroll**

When this option is off, each line is scrolled off the screen one at a time. When **Jump Scroll** is on, up to a full screen can be scrolled off the screen at once. When viewing a large volume of data from the remote host, turning on **Jump Scroll** can significantly enhance performance.

# **Minimize drawing (faster)**

Checking this option speeds the scrolling process by not refreshing the screen as often. .

#### Scroll to clear screen

Many applications send an escape sequence to clear the screen. When this option is turned on, the current screen's data is scrolled into the scrollback buffer prior to clearing the screen.

# Scroll to bottom on output

Check this button to have SecureCRT automatically scroll to the bottom of the incoming data when new data is received, as long as the session is

scrolled back less than a screen length. If this option is turned off, you must scroll back at least one page in order to not be scrolled back to the bottom on output.

### Scroll to bottom on keypress

Check this button to have SecureCRT automatically scroll to the bottom of the incoming data when a key is pressed.

### Mouse group

This group contains options that affect mouse behavior.

### Shift forces local mouse operations

This option causes mouse operations to be local rather than remote when the SHIFT key is pressed. This option is on, by default.

#### Send scroll wheel events to remote

This option allows SecureCRT to scroll remote applications. This only works when you run a remote application that starts an alternate screen buffer. This applies to most applications such as Emacs, Vim, and Midnight Commander.

### **Anti-idle group**

This group contains controls designed to prevent your <u>connection</u> from being closed due to a lack of network traffic between SecureCRT and the remote machine.

# Send string n every n seconds

Check this box to send a string to the remote machine on a regular basis.

- Enter the string that you want to send in the **Send string** entry box. Strings can contain commands or other information. For a listing of special octal and escape codes that can be included as part of the sent string, see the table of ASCII Control Codes and the table of Send String and Function Key Commands.
- Enter how often you want the string sent in the associated every n seconds entry box. The default value for this option is 300 seconds; the minimum is 10 seconds.

### Send protocol NO-OP every n seconds

Check this box to send a NO-OP instruction to the remote machine on a regular basis.

Enter how often you want the NO-OP sent in the associated

**every** *n* **seconds** entry box. The default value for this option is 60 seconds; the minimum is 10 seconds.

**Note:** Sending NO-OP instructions is only available for  $\underline{SSH2}$  and  $\underline{Telnet}$  sessions.

### **Selection group**

#### Set word delimiter characters

By default, SecureCRT uses all non-alphanumeric characters (except underscore) as word delimiters when selecting an entire "word" with a double-click action as described in Selecting Text in the Terminal Window. You can modify the word delimiter characters by entering your personal word delimiter characters in this entry box. Do not separate the entries with spaces or any other character.



# Terminal / Emulation

The **Emulation** category of the <u>session Options</u> dialog allows you to customize your terminal emulation for a session. Terminal keyboard emulation settings are defined by a keymap file. SecureCRT includes several builtin keymap files: VT100, VT220, Linux, SCO ANSI, Custom, and Default. The Default keymap file defines an emulation most similar to the PC keyboard, but also provides mappings for the keys F1-F4. SecureCRT will automatically choose a keyboard emulation based on the **Terminal** emulation that you select but you can change this selection using the Select an alternate keyboard emulation option described below.

#### **Emulation group**

This group contains general emulation controls.

#### **Terminal**

Select the terminal type that you want to emulate. SecureCRT supports ANSI, Linux, SCO ANSI, VShell, VT100, VT102, VT220, WYSE50, WYSE60, and Xterm terminal emulation.

### **ANSI Color**

Check this option to use the ANSI color palette defined in the Terminal/Appearance/ANSI Color Category of the Global Options dialog.

#### Use color scheme

When ANSI color is selected, check this option to honor the color scheme defined in the Global Options/Terminal/Appearance/Color Scheme dialog. Otherwise, the default ANSI foreground and background colors will be used.

# Select an alternate keyboard emulation

SecureCRT automatically chooses a keyboard emulation (or "keymap") to use based on the **Terminal** emulation described above. To select a different keymap, check this option and select from the list of keymaps, or use the Browse button to display the Open dialog and select the keymap you want.

For more information on customizing the keyboard for your needs, see Overview of Keymaps.

### Size group

The number of rows and columns that are defined in the parameters of a

connection between the client and the remote machine are known as the number of logical rows and columns. This group lets you specify an explicit number of logical rows and columns that appear in your session window. Any changes in the number of logical rows and columns will be reflected in the size of the session window.

#### **Logical rows**

Enter the integer value for the number of rows in the session window.

### **Logical columns**

Enter the integer value for the number of columns in the session window.

**Note:** SecureCRT allows you to set a maximum for the number of columns in the SecureCRT window (the upper limit allowed is 1024 columns). The **Maximum columns** setting in the <u>Terminal/Appearance</u> category of the <u>Global Options</u> dialog controls the maximum number of columns that can be specified under <u>Session Options</u>.

### On resize group

This group contains controls that allow you to specify how a session window behaves when manually resized.

### Synchronize view to size

Check this option to allow the number of rows and columns to match the size of the session window. If this option is not checked, the number of rows and columns will be maintained when resizing the window.

#### Scale the font

Check this option to increase or decrease the font size displayed in the session window as the window is resized.

#### Retain size and font

Check this option to maintain the font size displayed in the session window as the window is resized.

# Scrollback group

#### Scrollback buffer

Enter the number of lines to be saved in the scrollback buffer. This must be an integer value between 0 and 128000.

To clear the scrollback buffer:

- 1. Open the **Edit** menu.
- 2. Select the Clear Scrollback menu item.



# 隀 Terminal / Emulation / Modes

The **Emulation/Modes** category of the <u>Session Options</u> dialog allows you to customize the terminal emulation mode settings for a session, thereby configuring how the terminal operates.

You can configure SecureCRT to start with a customized selection of these modes on a per session basis. This is done by making mode selections within the **Initial modes** group. In addition, if a session is open, you can modify the current terminal modes by making mode selections within the **Current modes** group.

### **Cursor key mode**

Check this option to enable the cursor keys to send ANSI cursor control sequences to the remote host. If this option is not selected, the cursor keys send application control sequences for the remote host. By default, the **Cursor key mode** option is not selected.

#### Line wrap

Check this option to have any characters received when the cursor is at the right margin be displayed on the next line. The right margin is defined by the number of logical columns. If **Line wrap** is not selected, any characters received when the cursor is at the right margin will be displayed just to left of the right margin, replacing the current character displayed there. By default, **Line wrap** mode is selected.

#### New line mode

Check this option to have the ENTER key generate both CR and LF characters. In addition, any LF, FF, or VT character will cause the text cursor to be moved to the first column of the next line. When **New line mode** is not selected, the ENTER key generates only the CR character. By default, **New line mode** is not selected.

# **Numeric keypad**

Select this option to have the keypad send ASCII character codes that correspond to the numeric characters shown on the keys.

**Note:** The **Application keypad** mode and **Numeric keypad** mode are mutually exclusive.

# **Application keypad**

Select this option to have the keypad send escape sequences to the

application on the remote host.

# **Mode switching group**

Some applications send special escape code sequences that toggle the cursor key, line wrap, new line, and numeric/application keypad modes. By default, these mode switching sequences are interpreted and mode switching is allowed. You can configure SecureCRT to ignore some or all of these mode switching sequences by checking or clearing the following options:

Enable 80/132 column switching Enable cursor key mode switching Enable keypad mode switching Enable line wrap mode switching



# Terminal / Emulation / Emacs

The **Emulation/Emacs** category of the <u>Session Options</u> dialog allows you to configure your session to use the ALT key as the meta key. This may be helpful if you commonly use Emacs.

### Use ALT as meta key

Check this option to enable Emacs compatibility and use the ALT key as the meta key.

**WARNING:** The Use ALT as meta key option will affect your ability to use ALT key combinations to access application menus. You will still be able to access the application menus by pressing and releasing the ALT key and then pressing the menu hot key. For example, to access the File menu when Use ALT as meta key is enabled, press and release the ALT key and then press the F key. However, holding down the ALT key and pressing F will be interpreted by Emacs and will not bring up the **File** menu. The pattern shown in this example applies to all ALT key combinations.

### Preserve ALTGR international key

Many international keyboards have an ALTGR key that uses the same keyboard code as CTRL+ALT to create an alternate shift state for accessing additional characters on certain keys. To enable the use of this key and prevent Emacs from interpreting it as a CTRL+ALT sequence, select the **Preserve ALTGR international key** option. With this option enabled, Emacs mode is ignored for all ALTGR combinations.

### **Use 8th bit for meta key (requires termcap support)**

Select this option to use the eighth bit for meta key combinations instead of sending the ESC key character. This option requires termcap support. By default, the ESC character is sent for meta key combinations when Emacs compatibility is enabled.

### Send escape for meta key

Select this option to have SecureCRT send an ESC character for meta key combinations. By default, the ESC character is sent for meta key combinations when Emacs compatibility is enabled.



# 阿 Terminal / Emulation / Mapped Keys

The Emulation/Mapped Keys category of the Session Options dialog allows you to create or edit keymaps for a session .

**Note:** Mapped keys apply only to the current session and will not carry over to other sessions. If you want to make a key assignment that works in multiple sessions, you may want to use the Keymap Editor.

#### Mapped keys group

This group lists the Key Combination, Function, and Argument for keys mapped in this session.

### Map a key...

This button allows you to map a key from the keyboard. When the **Map Key** dialog appears, press the key combination you want to map. In the Map Selected Key dialog, specify the action you want the key to perform, and any applicable arguments.

#### Edit...

This button allows you to edit the mapping for the key selected in the window above.

#### **Delete**

This button deletes the mapping for the key selected in the window above.

### Other mappings group

You may find it necessary to change the behavior of the BACKSPACE key by sending the DELETE sequence and vice versa. This group contains options controlling the BACKSPACE and DELETE key behaviors.

### Backspace sends delete

Check this option to have the BACKSPACE key send the DELETE character. The DELETE character is generally ASCII 127 (decimal).

### Delete sends backspace

Check this option to have the DELETE key send the BACKSPACE character. The BACKSPACE character is generally CTRL+H or ASCII 8 (decimal), 0x8 (hexadecimal), or 10 (octal).

# **CUA** settings group

# **Use Windows copy and paste hotkeys**

Check this option to enable the CTRL+C, CTRL+V, CTRL+A, and CTRL+F accelerator keys for copy, paste, select all, and find, respectively.

**WARNING:** Turning on **Use Windows copy and paste hotkeys** will disable your ability to send CTRL+C, CTRL+V, CTRL+A, and CTRL+F to the remote host. Most UNIX users should leave this option off.



# Terminal / Emulation / Advanced

The **Emulation/Advanced** category of the <u>session Options</u> dialog allows you to modify standard terminal emulation settings for the selected session in order to communicate with special remote machines. Most users will not need to modify any of these settings.

### Advanced terminal options group

This group contains advanced emulation controls.

#### **Answerback**

By default, SecureCRT does not respond to the ENQ character with an answerback message. To enable the answerback feature for this session, check this option and enter the message to be sent.

With the Answerback option selected, SecureCRT will respond to the ENQ character with the specified answerback message. You can include substitution variables to be sent to the remote server when the server sends the ENQ (control byte 5) command.

### **Terminal type**

By default, SecureCRT sends the terminal type to the remote machine as designated by the terminal emulation type. To send a string other than the terminal type emulated by SecureCRT to the remote machine, check this option and enter the terminal type string.

# Display tab as

Some applications produce data in a tab-separated columnar format. SecureCRT allows you to change how the tab character is displayed by substituting a specific character or string in place of the tab character. For example, substituting a comma in place of the tab character would produce comma-separated data which could be imported into a spreadsheet or other application that uses commaseparated data.

To change the text displayed when a tab character is received by SecureCRT, check this option and enter the tab replacement string or character.

### Other group

This group contains miscellaneous emulation controls.

#### Local echo

Most systems that accept remote terminal connections receive characters from the terminal keyboard and echo them to the terminal screen. However, there are some legacy systems that do not echo characters received from the terminal keyboard to the terminal screen. On such machines, you will not be able to see what you have typed unless you enable local echo. By default, local echo of user input is disabled.

To enable local echo of user input, check this option.

### Strip 8th bit

ASCII uses seven bits to represent characters. Some seven-bit systems do not clear the eighth bit, causing some characters to be displayed incorrectly. If some characters are not being displayed correctly, try enabling the **Strip 8th bit** option.

To cause SecureCRT to strip (ignore) the high-order bit of each byte received from the remote host, check this option.

### Ignore window title change requests

This option causes SecureCRT to ignore escape sequences that would change the title bar.

### **Copy translates ANSI line-drawing characters**

This option causes SecureCRT to translate ANSI line-drawing characters to the three characters "+", "-", and "|" when they are copied to the clipboard.

### Copy to clipboard as RTF and plain text

This option allows SecureCRT to copy ANSI color and font data to the clipboard in RTF format so that it can be pasted into document editors (e.g., Microsoft Word).

### Translate incoming CR to CR/LF

This option causes SecureCRT to translate an incoming carriage return character (CR) to a carriage return/line feed character (CR/LF).

# Send delay options group

This group contains line and character send delay controls.

### Line send delay

This option allows you to set the number of milliseconds that SecureCRT pauses after sending a carriage return. This only applies in the following circumstances:

- When text is pasted into the SecureCRT window
- When text is sent by selecting the Send ASCII command from the Transfer menu
- When data is sent from an ActiveX script using SecureCRT Send() function

## **Character send delay**

This option allows you to set the number of milliseconds that SecureCRT pauses after sending each individual character. This only applies in the following circumstances:

- When text is pasted into the SecureCRT window
- When text is sent by selecting the Send ASCII command from the Transfer menu
- When data is sent from an ActiveX script using SecureCRT Send() function

### **Prompt**

This option allows you to specify a string to wait for. This only applies in the following circumstances:

- When text is pasted into the SecureCRT window
- When text is sent by selecting the **Send ASCII** command from the **Transfer** menu
- When data is sent from an ActiveX script using SecureCRT Send() function

### Max wait

This option is used in conjunction with the **Prompt** option to specify the maximum number of milliseconds to wait. A value of 0 indicates that SecureCRT will pause until the prompt is received.



# Terminal / Appearance

The **Appearance** category of the <u>session Options</u> dialog allows you to configure the appearance of the screen for the selected session.

### **Current color scheme group**

This list box displays the current color scheme for this session's window and allows you to select the color scheme for your session windows from the existing schemes. By using the **Edit...** and **New** buttons, you can also modify existing schemes or create new ones.

**Note:** If the settings for this option differ from the color scheme settings in the Global Options/Terminal/Appearance/Color Scheme dialog, these settings will override the global settings.

For more information on color schemes, see Overview of Color Schemes and Creating or Modifying a Color Scheme.

### Fonts group

This group allows you to select the font displayed in your session window.

### **Normal font**

This box displays the current font used in this session's window. By using the **Font...** button, you can select a different font to display.

#### Narrow font

Check this option to display the narrow font currently being used and to enable the **Font...** button. By using the **Font...** button, you can select a different font to display.

## Character encoding

Different computer systems can use different character encodings for the same language. For example, UNIX variants often use a character encoding called Extended UNIX Coding (EUC) to represent Japanese characters. Microsoft Windows uses an encoding called Shift JIS. SecureCRT also supports UTF-8 character encoding for those systems that require it. This option allows you to select the encoding method you need to display your language. Select the type of encoding used by the remote system. If the remote system uses the same character encoding as used by Microsoft Windows systems, select Default. This option uses facilities built into the Microsoft

operating system to decode and display the data. If the remote system uses a different encoding, select from the available encoding methods. If the encoding method for your language is not available, try setting this option to Default (be sure that your font and script settings are correct). If you still cannot display your language, please let us know.

## **Use Unicode line-drawing characters**

Check this box to use Unicode line-drawing characters.

### **Cursor group**

This group allows you to configure the cursor used in this session's window.

### **Cursor style**

This option allows you to configure the style of cursor displayed.

### **Use color**

This option allows you to configure the color of the cursor displayed. By using the **Color...** button, you can select a different color for display.

## **Blinking**

This option turns on and off the blinking of the cursor for this session's window. To stop the blinking, clear the check box.

## Highlight keywords group

This group allows you to create lists of keywords that can be highlighted in the SecureCRT terminal window.

#### Name

This option allows you to select (or create) a list of keywords that you want to be highlighted in the terminal window (the entire keyword will be highlighted). To create a new list of keywords, select the New... entry from the drop-down list. This opens the **Keyword List Properties** dialog where you can assign a name to the new list and add words to it. To edit an existing list, select the name of the list you want to modify and press the **Edit** button.

The word delimiters for keyword highlighting are the following:

Word delimiters specified in the <u>Terminal</u> category of the <u>Session Options</u> dialog only affect word selection by double-clicking and do not affect

keyword highlighting.

To disable the highlight feature, select <None> from the drop-down list.

## Style

This option allows you to select the style in which keywords are highlighted.

**Note:** Using this option reduces output display performance. If the host's shell supports highlighting, that is the recommended way to highlight keywords.



## Terminal / Appearance / ANSI Color

The Terminal/Appearance/ANSI Color category of the Session Options dialog allows you to customize the ANSI colors displayed in your SecureCRT application window.

Remote applications may send ANSI color escape sequences to specify that various portions of that application's display appears in one of 16 colors. Eight of the escape sequences correspond to what are called "normal" colors and the other eight typically correspond to "bold" versions of those colors. It is difficult to classify or label these colors beyond "normal and "bold" because an application may use any of the colors for whatever it wants.

Any application that you connect to with SecureCRT may use any ANSI color escape sequence (and its associated color) for its own purposes. While it is possible to configure SecureCRT globally to display 16 specific colors in response to ANSI color escape sequences, two different applications may specify different escape sequences for similar features. For example, when displaying filenames, one application may use the escape sequence that defaults to blue and another application may use the escape sequence that defaults to cyan.

Note: ANSI colors can be enabled or disabled for a specific session from the Terminal/Emulation Category of the Session Options dialog.

## Use global ANSI color settings

Check this option to use the ANSI color settings specified in the Global Options / Terminal / Appearance / ANSI Color Category instead of configuring sessionspecific settings.

## **Normal colors group**

This area displays the colors that are associated with the eight "normal" ANSI color escape sequences.

## **Bold colors group**

This area displays the colors that are associated with the eight "bold" ANSI color escape sequences.

## **Customizing ANSI Colors**

To customize ANSI color selections, click on the color block that you want

to change and select a new color from the **Color** dialog. Pressing the **Default** button returns the color blocks to their original settings.



## 🗺 Terminal / Appearance / Window

The **Appearance/Window** category of the <u>session Options</u> dialog allows you to configure the appearance of the window for the selected session(s). To turn off any of these options, clear the associated check box.

**Note:** The check boxes for the features in this category can be set to one of three states.

- No check mark -- Do not show the feature for this session.
- Check mark -- Show the feature for this session.
- Solid square -- Accept the global setting for this feature.

### Show chat window

Check this option if you want to show the chat window.

The SecureCRT chat window provides an interface to the remote command line that is unaffected by concurrent output to the terminal screen. The chat window is a great interface to use when communicating in the environment of a chat room. In addition, the chat window is useful for issuing subsequent commands. While waiting for the output of a previous command to complete, you can prepare the next command by typing it into the chat window. The contents of the chat window are sent to the command line as soon as the ENTER key is pressed in the chat window.

You can prepare multiple lines in the chat window with CTRL+ENTER.

SecureCRT will remember the number of recently used chat window commands that are specified in the **Chat window history** option of the Global Options/Terminal/Appearance dialog. You can cycle through these commands using your UP and DOWN arrow keys. Entering a 0 for the **Chat window history** will turn off this feature.

If you want to send commands to all your open tabbed sessions at once, right-click in the chat window and select the Send Chat to All **Tabs** option. Commands will be sent to all your tabbed sessions as long as this option is enabled,

### Show horizontal scroll bar

Check this option if you want to show the horizontal scroll bar.

### Show vertical scroll bar

Check this option if you want to show the vertical scroll bar.

### Show button bar

Check this option if you want to show the button bar in the current session. You can also select which button bar you want to display from the drop-down list.

### Disable resize

Check this option if you want to disable manual resizing of the session window.

### Session window always on top

Check this option to keep the session window in front or "on top" of other windows on your screen.

### Title bar

Check this option if you want to change the title displayed in the title bar of your session (if you are using tabs, this option will change the name at the top of your session tab). Enter the alternate title in the associated entry box. You can also use the following variables which will be expanded when SecureCRT writes to the title bar.

%H - hostname

%S - session name

%U - username

%I - IP address

%P - port

**Note:** It is possible for some remote systems to change the title bar text by sending an escape sequence. If you want these escape sequences to be ignored, set the **Ignore window title change requests** option found in the <u>Terminal / Emulation / Advanced</u> Category.

You can also use the commands listed in the <u>Send String & Function Commands</u> topic.

## **Initial position**

If you want this session window to always open in the same screen position, enter that position's x,y pixel location coordinates.

**Note:** If negative values are used for x or y, the initial position will be the offset from the right or bottom, respectively. In a dual monitor configuration, negative initial position values will be

honored.

You can also change the window's initial position by following these steps:

- 1. In the <u>session Options</u> dialog, open the **Terminal/Appearance/Window** category.
- 2. Be sure that the **Initial position** option is <u>not</u> selected.
- 3. Click on the **OK** button at the bottom of the **Session Options** dialog.
- 4. Move the session window to the preferred default position.
- 5. Open the **Options** menu and select **Session Options...**
- 6. In the **Session Options** dialog, click on the **Terminal/Appearance/Window** category.
- 7. Check the **Initial position** option check box.
- 8. Click on the **OK** button at the bottom of the **Session Options** dialog.

After completing these steps, the SecureCRT window will now be placed in the location specified each time you connect with the session.



## Terminal / Log File

The **Log File** category of the <u>session Options</u> dialog allows you to customize the logging options for the selected session.

### Log file name group

This section allows you to specify a default filename for your log file. You can enter a path or filename, or you can use the **Browse** button — to open the **Save As** dialog and select the path or filename. You can also include any of the variables listed in the **Substitution** section below. These variables will be expanded when SecureCRT writes to the log file.

## **Options** group

This section lets you specify log file behaviors.

### Prompt for filename

Check this option to prompt the user for the session log filename when the log is started.

## Start log upon connect

Check this option to start writing to the session log file whenever a connection is made. By default, this option is not selected.

### Overwrite file

If this option is selected and the session log file already exists, the current log file will be overwritten by the new session log file.

## Append to file

If this option is selected and the session log file already exists, the new session log will be appended to the existing log file. If the log file does not already exist, a new file is created.

## Raw log

Check this option to write every character received by SecureCRT, including terminal escape sequences, to the session log file. By default, this option is off.

## Start new log file at midnight

Check this option to have SecureCRT start a new log file each night at midnight. Using this feature can aid in automatic log rotation.

## **Custom log data group**

This section lets you specify custom messages to be written to the log

file. You can include any of the variables listed in the **Substitution** section below. These variables will be expanded when SecureCRT writes to the log file.

### **Upon connect**

Entries in this box will be written to the log file when the session connects.

### **Upon disconnect**

Entries in this box will be written to the log file when the session disconnects.

### On each line

Entries in this box will be written to each line of the log file.

## Log only custom data

If this option is selected, SecureCRT will log only the custom data from the three fields above. All normal log messages will be suppressed.

## **Substitutions group**

In the dialog, this section lists some of the variables that SecureCRT can expand when writing to the log file. A complete list is shown below.

%H - hostname

%S - session name

%Y - four-digit year

%M - two-digit month

%D - two-digit day of the month

%P - port

%h - two-digit hour

%m - two-digit minute

%s - two-digit seconds

%t - three-digit milliseconds

%% - percent (%)

%envvar% - environment variable

Note: The environment variable substitution occurs first.



# 👼 Terminal / Printing

The **Printing** category of the <u>Session Options</u> dialog allows you to set up your printer configuration for the selected session.

### Use global print settings

Check this option to use the global options instead of configuring session-specific options. By default, all print settings for SecureCRT sessions are derived from global print settings.

**Note:** If this option is not checked, the settings specified here will override the setting in the Terminal/Printing Category of the Global **Options** dialog for the selected session.

## **Margins group**

This group allows you to set the margins on your printed pages. Margins are measured in inches. By default, all print margins are set at 0.5 inches.

### Printer fonts group

This group displays the font currently being used for printed pages and also allows you to choose a different font.

### Choose...

Press this button to open the **Font** dialog where you can choose the font you want to use for printed pages.

## **Print Setup...**

Press this button to open the Windows **Print Setup** dialog where you can modify your printer configuration.

## **Pass-through printing group**

Pass-through printing in SecureCRT allows the use of special print control escape sequences that turn on/off a printer control that sends characters directly to the printer. Depending on the sequence received, SecureCRT may bypass the terminal display and send the data directly to the printer.

## Disable pass-through printing

Check this option to disable pass-through printing and cause SecureCRT to ignore all printer control escape sequences it receives:

## **Buffer pass-through printing**

Check this option to buffer pass-through printing. This will avert the issue of some applications that frequently toggle pass-through printing mode on and off, resulting in printed pages with only one line or pages that are otherwise incomplete. This option buffers all pass-through printing data and does not send it to the printer until you instruct SecureCRT to print the contents of the pass-through printing buffer.

To print the contents of the pass-through printing buffer:

- 1. Open the **File** menu and select **Print**.
- 2. In the **Print** menu, select **Eject Page**.

### Use raw mode

Check this option to allow the use of a network printer when using pass-through printing.

Raw mode pass-through printing allows you to send data to a named printer as defined in Windows. In addition, raw mode instructs Windows to allow the data to be sent to the named printer without being interpreted (i.e. by the printer driver). If your host or application sends embedded printer commands (e.g., change font, etc.), enabling raw mode pass-through printing will allow these commands to be sent directly to the printer. If raw mode pass-through printing is enabled, SecureCRT will not interpret any embedded printer commands.

## Direct pass-through printing to port *n*

Check this option to print directly to the COM or LPT port on the local machine to which a printer is attached.

Direct pass-through printing to a port essentially bypasses the Windows operating system, providing SecureCRT with direct communication to the local printer. If direct pass-through printing is enabled, print jobs are not spooled but are sent directly to the printer. The port to which the printer is attached will not be accessible to other applications until the current print job has been completed. For these reasons, it is suggested that you use raw mode pass-through printing rather than direct pass-through printing to port (see **Use raw mode** above).

## Settings...

Click on this button to open the <u>Direct Pass-Through Printing Settings</u> dialog, which allows you to configure the serial device settings for pass-through printing.

## Other

## Force black on white

Check this option to force your printer to print as though your session's foreground color is set to black and your background color is set to white, regardless of your session's actual color settings. This option is off by default.



## Terminal / X/Y/Zmodem

The **Terminal/X/Y/Zmodem** category of the <u>Session Options</u> dialog allows you to configure your Xmodem, Ymodem, and Zmodem settings.

### **Directories group**

### **Upload**

Enter the path of a local directory that is to be used as the source folder when the **Upload** command is used. Click on the browser button to select the directory.

### **Download**

Enter the path of a local directory that is to be used as the destination folder when the **Download** command is used. Click on the browser button to select the directory.

## X/Ymodem send packet size group

This group allows you to select the size of the Xmodem and Ymodem packets you send. Your options are as follows:

128 bytes 1024 bytes (Xmodem-1k/Ymodem-1k)

### **Zmodem**

### Disable Zmodem

Check this option to turn off the ability to make Zmodem transfers.

## **TFTP server group**

### Start TFTP server when session connects

Check this option to launch a TFTP server when the session connects. The TFTP server will use the upload and download directories specified in the **Directories** group in this category. Note that the upload and download directories are used opposite of how they are normally used in SecureCRT because the TFTP server is a server instead of a client.



The **File Transfer** category of the <u>session Options</u> dialog allows you to control attributes of the directory locations for a given <u>session</u>.

## **Local directories group**

### **Local window**

Enter the path of the local directory that you want the local window to display on startup of the session. This is the default folder from which you want to transfer files to the remote machine. Click on the browser button to use a directory browser to select the directory.

## **Download directory**

Enter the path of the local directory to be use as the destination folder when the **Download** command is used. This is the default folder in which you want to receive files from a remote machine. Click on the browser button to use a directory browser to select the directory.

## Remote directories group

## **Initial directory**

Enter the full path of the directory on the remote host that you want to be the current directory upon connecting to the remote host. If the initial directory is not specified, the default or home directory for the username on the remote system will be the initial directory.

## **Directory tree options group**

## Disable directory tree detection

This option is unchecked by default leaving the directory tree enabled. Checking this option will disable the directory tree for the current session.

This check box can also be set to a third, indeterminate state. When the check box is in this state, it appears grayed out and will cause SecureFX to use the **Disable directory tree detection** setting in the Global Options/File Transfer/Options dialog.

## **Transfer options group**

## Verify file status on retrieve

This option is checked by default. It causes a LIST command to be

sent prior to retrieving a file from the FTP server in order to obtain the most current statistics (i.e., size and date) for the file.

This check box can also be set to a third, indeterminate state. When the check box is in this state, it appears grayed out and will cause SecureFX to use the **Verify file status on retrieve** setting in the Global Options/File Transfer/Options dialog.



## File Transfer / FTP/SFTP

This category only appears if VanDyke Software's SecureFX® 3.0 (or newer) is also installed.

The File Transfer/FTP/SFTP category of the Session Options dialog allows you to control server and upload attributes for a given session.

### **Server parameters group**

### Operating system

Select the operating system used on the server. By default, this option is set to automatically detect the operating system on the server when connecting. This option allows you to override the automatic detection and specify a supported type.

### Time zone

Select the time zone of the server. This information is used when translating file times from the server so that a more accurate comparison can be performed when comparing local file time and remote file times.

By default, this option is set to automatically detect the time zone of the server, if possible, using the server's connection reply. Many servers put time and date information in the initial reply text sent from the server, and SecureFX will attempt to determine time zone of the server from this text. If it fails to detect an appropriate time zone, the local time zone (i.e., the time zone in use on the client machine) is used by default.

## **Upload options group**

### Filename conversion

Choose the filename conversion rule that will be applied to uploaded files. The choices for this option are as follows:

- **Preserve case** causes SecureFX to create the file on the remote machine using the filename exactly as it is stored on the local machine.
- **Uppercase** causes the alphabetic characters in a filename to be converted to all uppercase.
- Lowercase causes the alphabetic characters in a filename to be

converted to all lowercase.

### File creation mask (not available when using SFTP)

Enter a file creation mask (also called the UMASK) that will be used when a new file is uploaded to the FTP server. By default, this option is blank, which means that the UMASK command will not be sent to the FTP server, and the server will use the default UMASK setting for current logon.

The file creation mask is a value that is used to determine the default permissions settings for new files. This feature is usually only available on UNIX systems.

The value specified for the file creation mask is given in octal and is XORed on the remote system with the permissions of 666 for files and 777 for directories. For example, the value 002 would give complete access to the group and only read (and directory search) access to others. The value 022 would give only read (and directory search) access to the group and others.



# 👼 Customize Dialog

The **Customize** dialog allows you to customize the user interface menu bar and toolbars to suit your needs and preferences. You can show or hide toolbars, rearrange menu commands and toolbar buttons, add or remove commands and buttons, and create new menus and toolbars.

**Note:** You can also customize your toolbars by using the ALT+drag method. To do this, place the mouse pointer over a menu title or toolbar button, hold down the ALT key, and drag the menu or button to a new location, or totally off the bar if you want to remove it.

The **Customize Toolbar** dialog can be accessed in several ways:

- Right-click on any of the bars and select **Customize...** from the context menu.
- Right-click on the **Toolbar Options** button on the main toolbar and select **Customize...** from the context menu.

### Commands tab

### Categories window

Displays the categories of commands available for use in menus and buttons. Selecting a category displays the associated commands in the **Commands** window.

### Commands window

Displays the commands associated with the selected category. These commands can be added to menus and used as toolbar buttons.

You add commands or buttons to menus and toolbars by clicking on that command and dragging it to the desired menu or toolbar. If a command that you want to add as a toolbar button does not have a button icon, SecureCRT will create a button on the toolbar displaying the command's name.

### **Toolbars tab**

### **Toolbars window**

Displays the Menu bar, Main toolbar, and any other toolbars that have been created. To hide a toolbar in the interface, uncheck the box to the left of that toolbar's name. The **Menu bar** cannot be hidden.

When the **Customize** dialog is open, SecureCRT allows you to add

commands to and remove commands from menus using drag and drop. To add a command to a menu, drag that command from the **Commands** tab to the menu where you want it. To remove a command from a menu, open that menu and drag the command off of that menu.

Clicking on the **Reset** button on this tab will reset the selected toolbar to its default values. Clicking on the **Reset All** button resets all the toolbars.

### Menu tab

## **Application Frame Menus group**

### **Show Menus for**

Select the menu bar to use.

### Reset

Resets selected menu bar to default values.

### **Menu animations**

Select your preferred method of animating the opening and closing of SecureCRT menus.

### Menu shadows

Select this option to have an open menu display a shadow.

## **Options tab**

## **Toolbar group**

## **Show ScreenTips on toolbars**

Check this option to have SecureCRT display ScreenTips when you hover your pointer over a toolbar button.

## Show shortcut keys in ScreenTips

Check this option to have SecureCRT display alternative keyboard shortcut options with ScreenTips when you hover your pointer over a toolbar button.



# Direct Pass-Through Printing Settings

The **Direct Pass-Through Printing Settings** dialog allows you to set the serial device settings for pass-through printing.

### **Baud rate**

Select the baud rate for the device. The default value is 38400.

### **Data bits**

Select the number of communication data bits. The default value is 8.

### **Parity**

Choose the parity setting supported by the connected device. The default setting is None.

### **Stop bits**

Choose number of stop bits sent after each character. The default value is 1.

### Flow control

### DTR/DSR

Select this option to use the DTR/DSR (data-terminal-ready/data-setready) signals with the connected device. By default, this is not selected.

### RTS/CTS

Select this option to use RTS/CTS (request-to-send/clear-to-send) hardware flow control. By default, this option is selected.

### XON/XOFF

Select this option to use XON/XOFF software flow control. By default, this option is not selected.



# Firewall Properties Dialog

The Firewall Properties dialog can be accessed by clicking on either the Add... or Edit... button on the Firewall category of the Global Options dialog.

### Name

Enter unique name for this firewall. This name will be used in the session options dialog to specify the firewall for a given session.

### Type group

Select the type of firewall that you are using from the list of firewalls that are supported by SecureCRT.

### **Parameters** group

This group allows you to configure the parameters for the firewall **Type** that you selected above. The options in this group vary based on the **Type** selection. The supported firewalls are listed below:

- **Generic/Telnet -** Choose this option if you connect through a proxy server that uses a simple send/expect interface. Selecting this type will enable the **Proxy prompt** and **Proxy command** options described below.
- **SOCKS Version 4**
- SOCKS Version 5 (no authentication )
- SOCKS Version 5 (username/password) If your firewall server is running either SOCKS version 4 or SOCKS version 5, select one of these types for the Type field. Enter the hostname or IP address of your SOCKS firewall server in the "Hostname or IP" field, and enter the SOCKS firewall port number in the "Port" field. The conventional port number for SOCKS is 1080.
  - The "username/password" version of the SOCKS firewall choices allows for firewall user authentication. The only method of authentication currently supported is the "Username/Password Authentication" protocol, which requires the username and password to be sent as clear text. Therefore, this method is not recommended for non-secure environments
- **HTTP** (no authentication) This type allows your server to connect through an HTTP proxy server.
- HTTP (Basic username/password) This type allows your server

to connect through an HTTP proxy server that supports "Basic" authentication (this is an HTTP standard). "Basic" authentication sends usernames and passwords in the clear, therefore, this method is not recommended for nonsecure environments.

- OPEN host Select this type if your firewall server requires the OPEN command. This type may be selected for the CSM proxy server. Also, enter the hostname and port of the firewall server in the appropriate fields.
- USER user@host:port Select this type if your server is running the WinGate proxy server. Enter the hostname or IP address and port of the firewall server in the appropriate fields. This type may also be selected for use with the CSM proxy server. This type should also be used for the WinProxy proxy server. For this particular proxy server you may also need to have the Use outgoing data connections (PASV) option unchecked in the Connection/FTP category of the Session Options dialog.
- USER user@host port
- USER user@host port (with login) Select these types if your server is running the TIS Internet Firewall Toolkit. Choose the first form if the firewall server does not require authentication. If your firewall server requires authentication, choose the second form. Also, enter the hostname or IP address and port of the firewall server in the appropriate fields.
- USER user@host login This type will send the remote username and the server address followed by the firewall user ID. It then expects to send the password for the remote user and the password for the firewall user. This type should be used if your server is running the Raptor Firewall.
  For this firewall type you will need to enter the username and password that is required by the firewall server to authenticate your connection. Also, enter the hostname or IP address and port of the firewall server in the appropriate fields.

#### Hostname or IP

Enter the hostname or IP address of the machine to which you want to connect.

#### **Port**

Enter the port to which you want to connect. For a SOCKS firewall,

this is usually 1080.

### Username

Enter the username for this connection.

### Password

Enter the password for this connection.

### **Proxy prompt**

This option applies only if you have selected Generic/Telnet proxy as your firewall **Type**.

Enter the prompt that SecureCRT can expect to receive from the generic/Telnet proxy. For example, to use the TIS Firewall toolkit, the prompt would be:

tn-gw>

For WinGate, the prompt would be:

WinGate>

### **Proxy command**

This option applies only if you have selected Generic/Telnet proxy as your firewall **Type**.

Enter the connect or open command that SecureCRT needs to send to the generic/Telnet proxy. For example, to use the TIS Firewall toolkit, the command would be:

c %h % p\r

For WinGate, the command would be:

The <u>SSH</u> protocol imposes some special constraints on what firewall software can be used. SSH does work with SOCKS firewalls. SSH does not work with generic firewalls that are not 8-bit clean or that modify the contents of the data stream. SSH cannot be used through the TIS tn-gw generic proxy.



## GSSAPI Properties Dialog

The GSSAPI Properties dialog can be accessed by clicking on the Properties button in the Authentication group of the Connection SSH1, SSH2, or Telnet category of the Session Options dialog when GSSAPI is the specified authentication method.

GSSAPI (Generic Security Services Application Program Interface) is a generic API for performing client/server authentication. GSSAPI allows SecureCRT to authenticate with a server without knowing anything about the specific authentication mechanism in use.

### Method

SecureCRT supports the following types of GSSAPI provider:

- MS Kerberos In order to use this provider, SecureCRT must be running on Windows 2000 or newer. The Windows 2000 computer must have been configured as part of an Active Directory domain or been configured to participate in a Kerberos realm.
- GSSAPI In order to use this provider, you must have a Gssapi32.dll file provided by a third party (e.g., the MIT Kerberos distribution). This third-party application must be configured correctly for your environment.
- Auto Detect This setting instructs SecureCRT to attempt to automatically determine which of the above two methods will work with the server that you are connecting to. This is the recommended setting.

## **Delegation**

When SecureCRT authenticates with GSSAPI, it can control whether or not the server is allowed to access other secured resources (such as network file servers) without further prompting for credentials. SecureCRT supports the following delegation settings:

- Full If this delegation is selected and the GSSAPI mechanism both supports delegation and is configured to allow delegation, the server may be able to access other secured resources without prompting for credentials.
- None If this delegation is selected, the server may have to prompt for further authentication in order to access secured

resources such as network files, printers, or to log on to a different server.

Limited - This delegation is the same as Full delegation for the MS Kerberos method. If the GSSAPI method is in use, it's meaning is determined by the Gssapi32.dll in use.

### << Advanced

Pressing this button expands (or contracts) the GSSAPI Properties dialog to display (or hide) the following options.

### **SPN (Server Principal Name)**

When authenticating with GSSAPI, SecureCRT must determine the canonical name of a server. The server has exactly one canonical name, which no other server can share. The server may have other names, for example, the server 192.168.20.1 may be known as mail.mydomain.com, mydomain.com and mail, but it has only one canonical name, mail.mydomain.com.

SecureCRT uses this canonical name to form a Server Principal Name (SPN), which the GSSAPI provider uses to identify the server with which it should authenticate.

SecureCRT usually uses the host variable (HOST) to determine the server SPN. However, this depends on hostname lookups working correctly. If this does not work correctly, this behavior can be overridden by manually specifying the SPN.

## Manually specify the SPN (default is host@\$(HOST))

Checking this box will enable the SPN text box below and manually specify the SPN.

#### SPN

Enter the SPN string. The string is almost always of the form host@<server canonical name>. An example of a valid string is "host@mail.mydomain.com". SecureCRT will make the following variable substitutions in the specified SPN name:

- \$(HOST) the hostname as specified in the <u>session</u>
   Options/Connection/SSH2 Category.
- \$(PORT) the port as specified in the <u>session Options/Connection/SSH2</u> category

If the server is in a different Kerberos realm, the realm name may need

to be appended (e.g., host@mail.mydomain.com@KRBS.MYDOMAIN.COM.



## **Keymap Editor Dialog**

The **Keymap Editor** dialog is accessed from the **Keymap Editor** toolbar button or from the **Keymap Editor...** command on the <u>Tools</u> menu. The **Keymap Editor** allows you to modify keymaps. You can make changes to the keymap currently selected in the <u>Terminal/Emulation</u> category of the <u>Session Options</u> dialog, or you can load and modify other keymaps.

SecureCRT provides built-in keymaps for the ANSI, Linux, SCO ANSI, VShell, VT100, VT102, VT220, WYSE50, and WYSE60 terminal emulations. You can also create custom keymap files by saving modifications made with the **Keymap Editor** as a new file. Keymap files are saved with the extension .key.

Unlike the Mapped Keys function, which only changes key associations for the Session in which the keys were mapped, built-in and custom keymaps can be used by multiple sessions by setting the Select an alternate keyboard emulation to the same keymap.



Keymap Editor Dialog

## Keyboard

The top part of the **Keymap Editor** dialog is a representation of a physical keyboard. By clicking on a key in the keyboard area, you can see if the key is mapped to an action (displayed in the **Selected key maps to** field) or select it for mapping. This area also provides check

boxes for **Alt**, **Ctrl**, and **Shift** modifiers that you can add to key combination selections.

**Note:** The ALT+F6 key combination is reserved in Windows and cannot be mapped.

### Selected key maps to

This field displays any action to which the selected key is mapped.

### Map Selected Key...

After selecting a key in the keyboard area above, click on this button to open the Map Selected Key dialog. In that dialog you can map the selected key to one of the listed actions.

## Map a Key...

Click on this button to map a key using your physical keyboard. After you click on this button and press the key (or key combination) that you want to map, the Map Selected Key dialog will open allowing you to select the action that you want mapped to the specified key.

### **Unmap Selected Key**

Click on this button to remove the mapping of the selected key from the keymap.

### Save

Click on this button to save your modifications to the keymap currently in use.

### Save As...

Click on this button to save your modification to a keymap other than the one currently in use.

### Load...

Click on this button to load an existing keymap for editing. To edit the keymap being used by the current session, you do not have to load the keymap.

### Close

Click on this key to close the **Keymap Editor**.



# Local Port Forwarding Properties Dialog

The **Local Port Forwarding Properties** dialog can be accessed by clicking on either the Add... or Edit... button on the Connection/Port Forwarding category of the **Session Options** dialog.

### Name

### Name

Enter a unique name for this connection.

### Local

Manually select local IP address on which to allow connections Check this option to bind the local end of a port forward to an address other than the default localhost loopback adaptor (e.g., 127.0.0.2).

### **Port**

Enter the number or name of the port on the local machine from which you want to connect.

### Remote

### Destination host is different from the SSH server

Check this option if the remote destination host is different from the SSH server .

#### Port

Enter the number or name of the port on the remote machine to which you want to connect.

## Dynamic forwarding using SOCKS 4 or 5

Check this option to have SecureCRT act as a SOCKS proxy server on the specified port. This will allow any client that can connect using a SOCKS firewall to use the "dynamic port forward" to set up a port forward.

For example, if you have a dynamic port forward set up to use 127.0.0.3 on port 1080, you could then use that one dynamic port forward to connect to any host behind the firewall.

## **Application**

## **Application**

SecureCRT allows you to optionally start an application on the local machine once you have established a connection. Enter the full pathname of the application.

## **Arguments**

Enter any arguments that you want to use with the specified application.



## **Map Button Dialog**

The **Map Button** dialog can be accessed by right-clicking anywhere in the button bar and select the **New Button...** option.

The **Map Button** dialog allows you to associate an action (function and associated argument) with the button selected when accessing the dialog.

### **Action**

This group contains the functions and arguments that make up a button action. A list and descriptions of the available functions is in the <u>Table of Keymap Functions</u>.

### **Function**

Select the type of function that you want to assign to the specified button. You also need select the specific function details that the **Function** requires.

### <Function details>

This entry box will change depending on the selected **Function** option. Select or enter the specific function that you want to assign to the specified button.

#### Label

Enter the text you want to appear as the label on the button.

## **Argument**

This entry box appears if you select Run Script in the **Function** box above. Enter argument (or arguments) needed for your selected script. These script arguments function in the same manner as if they were passed in using the command line /ARG option.



# **Map Selected Key Dialog**

The Map Selected Key dialog can be accessed by several methods:

- Open the <u>Keymap Editor</u>, select a key to be mapped from the keyboard area, and click on the **Map Selected Key...** button.
- Open the Keymap Editor, click on the **Map a Key...** button, and enter a key or key combination to be mapped.
- Open the <u>Terminal/Emulation/Mapped Keys</u> category of the <u>Session</u>

  <u>Options</u> dialog, click on the **Map a Key...** button, and enter a key or key combination to be mapped.
- Open the <u>Terminal/Emulation/Mapped Keys</u> category of the <u>Session</u>

  <u>options</u> dialog, select a mapped key from the list, and click on the **Edit** button.

The **Map Selected Key** dialog allows you to associate an action (function and associated argument) with the key (or keys) selected when accessing the dialog.

### **Key Combination**

This field displays the key or key combination specified when accessing this dialog.

### **Action**

This group contains the functions and arguments that make up a key action. A list and descriptions of the available functions is in the  $\frac{\text{Table of }}{\text{Keymap Functions}}$ .

#### **Function**

Select the type of function that you want to assign to the specified key. You also need to select the specific argument that the Function requires.

#### <Function details>

This entry box will change depending on the selected **Function** option. Select or enter the specific function that you want to assign to the specified key.

### **Argument**

This entry box appears if you select Run Script in the **Function** box above. Enter argument (or arguments) needed for your

selected script. These script arguments function in the same manner as if they were passed in using the command line  $_{/ARG}$  option.



# Public-Key Assistant Dialog

The Public-Key Assistant dialog is accessed from the Public-Key Assistant toolbar button or from the Public-Key Assistant... option on the **Tools** menu. This dialog allows you to view and manage the public keys that are installed for the current user on the remote server that is configured for the session.

#### User

Displays the username of the current user.

### **Status**

Displays the status of the <u>connection</u> to the server.

The list window in this dialog displays the public keys that are installed for the current user on the remote server. For each public key, the list also displays the **Type** of algorithm used to generate the key (either DSA or RSA), the key's **Comment**, and the key's **Fingerprint**.

### Upload...

Click this button to choose another public key to upload to the remote server using the public-key assistant...

#### Create...

Click this button to open the Key Generation wizard where you can create and upload a new public key file.

#### Delete

Click this button to delete the selected public key from the server.

Note: If the computer on which SecureCRT is running does not support the key type for a key in the list, the **Fingerprint** will read "unsupported key type" and the **Delete** button will be disabled for that key.



# Public Key Properties Dialog

The **Public Key Properties** dialog can be accessed by clicking on the Properties button in the Authentication group of the Connection/SSH1 Or SSH2 category of the Session Options dialog. This dialog is not available when the session is connected.

Use global public key setting/Use session public key setting Select whether SecureCRT should use the global identity file or a specified session identity file as your method of authentication for this session. Your choice will determine the options available in the group below.

### Global/Session settings group

### Use identity or certificate file

Selecting this option instructs SecureCRT to use the specified identity or PKCS #12 file as your method of authentication. For more information on configuring your system to use identity files, see Public-Key Authentication for SSH1 Of Public-Key Authentication for SSH2.

SecureCRT also supports accessing X.509 certificates through PKCS #11 . The locator prefix (e.g., "pkcs11::") and suffix (e.g., "::standard") specify which public-key algorithm to use. To use this feature, enter a string similar to one of the following examples but pointing to your PKCS #11 .dll file in the text box:

```
pkcs11::prov=c:\windows\system32\opensc-pkcs11.dll
pkcs11::prov=c:\windows\system32\pkcs11.dll::cert
```

The above examples use the x509v3-sign-rsa algorithm.

```
pkcs11::prov=c:\windows\system32\opensc-
pkcs11.dll::standard
```

The above example uses the x509v3-sign-rsa-sha1 algorithm.

```
pkcs11key::prov=c:\windows\system32\pkcs11.dll
```

The above example uses the ssh-rsa algorithm, which can be used to send the certificate as a raw key.

To use an Amazon EC2 "key pair" with SecureCRT, specify the private key file of the key pair generated by Amazon as the identity or certificate file. This file typically has a .pem extension.

### **Fingerprint**

This entry box will be filled in automatically when an identity file is entered above.

### **Create Identity File**

Press this button to start the Key Generation wizard and create identity files which contain your public-private key pair.

**Note:** SecureCRT to supports both DSA and RSA key types.

### **Upload**

Press this button to open the **Upload Public Key** dialog and upload the specified identity file to the <u>server</u> configured for this session.

Public keys are uploaded using the public-key assistant. Uploads are only available on servers that support the public-key assistant (such as VanDyke Software's VShell® server 2.1 or newer).

### **Change Passphrase**

Press this button to change the passphrase for your identity file.

### **Use personal store certificate (CAPI)**

Selecting this option instructs SecureCRT to use X.509 certificates from your Microsoft CAPI personal store as your method of authentication. For more information on X.509 certificates, see <u>Using X.509 Certificates</u>.



# Remote Port Forwarding Properties Dialog

The **Remote Port Forwarding Properties** dialog can be accessed by clicking on either the Add... or Edit... button on the connection/Port Forwarding/Remote/X11 Category of the Session Options dialog.

### Name

#### Name

Enter a unique name for this connection.

#### Remote

Manually select remote IP address on which to allow connections Check this option to bind the remote end of a port forward to an address other than the default localhost loopback adaptor (e.g., an internal LAN interface).

### **Port**

Enter the number or name of the port on the remote machine from which you want to connect.

### Local

### **Destination host is different from localhost**

Check this option if the destination host is different from the localhost.

#### **Port**

Enter the number or name of the port on the local machine to which you want to connect.



# Select Files to Send...Dialogs

The Select Files to Send... dialogs consist of the Select Files to Send as ASCII dialog, the Select Files to Send as Binary dialog, the Select Files to Send using Xmodem dialog, the Select Files to Send using Ymodem dialog, and the Select Files to Send using Zmodem dialog. These dialogs have similar options and can be accessed from the Transfer menu using the Send ASCII..., Send Binary..., Send Xmodem..., Send Ymodem..., and Zmodem Upload List... options, respectively.

The following options appear on all four dialogs:

### Look in

Select the folder in which you want to look for the file that you want to send. The list below will display the contents of the specified folder. You can also use the buttons to the right of the entry box to help you navigate through and sort the items in the list.

### File name

Enter the name of the file that you want to send, or click on the file in the list above.

# Files of type

Select the type of files that you want displayed in the list above.

On the Select Files to Send as ASCII dialog, the Select Files to Send as Binary dialog, and the Select Files to Send using Xmodem dialog, you begin transferring the selected file by clicking on the **Send** button.

Ymodem and **Zmodem** support multiple file transfers and, therefore, the Select File to Send using Ymodem and Select File to Send using **Zmodem** dialogs have a few more options:

#### Add

Click on this button to add the file shown in the File name entry box to the **Files to send** list.

#### Files to send

This list displays the files that are queued to be sent using Zmodem. To send the specified files, follow these steps:

Click on the **OK** button to save the files and close the dialog 1.

- 2. Open the **Transfer** menu
- 3. Select the **Start Zmodem Upload** option.

#### Remove

Click this button to remove selected files from the **Files to send** window.

### Start Upload (Zmodem only)

Click this button to start the transfer immediately.

### **Upload files as ASCII**

Check this box to instruct SecureCRT to send the files in the **Files to send** window in ASCII format. If left unchecked, files will be sent in binary format.

**Note:** For this option to work properly, the installation of Zmodem on the remote machine must be able to convert ASCII to use the local convention for line termination.



# **Specify Filename to Receive... Dialogs**

The Specify Filename to Receive... dialogs consist of the **Specify Filename to Receive as ASCII** dialog and the **Specify Filename to Receive using Xmodem** dialog. These dialogs have the same options and can be accessed from the **Transfer** menu using the **Receive ASCII...** and **Receive Xmodem...** options, respectively.

The following options appear on both dialogs:

### Save in

Select the folder in which you want to save the file that you are to receive. The list below will display the contents of the specified folder. You can also use the buttons to the right of the entry box to help you navigate through and sort the items in the list.

### File name

Enter the name of the filename that you are to receive, or click on the filename in the list above.

### Save as type

Select the type of file that you want to save the received file as.

To begin transferring the specified file, click on the **Receive** button.



# **Upload Public Key Dialog**

The **Upload Public Key** dialog can be accessed by clicking on the **Upload** button on the <u>Public Key Properties</u> dialog or from the Key Generation wizard after you have created a public key file. This dialog allows you to view the status of public key file uploads to a <u>server</u>.

Using password <u>authentication</u>, SecureCRT attempts to connect to the server that is configured for the <u>session</u>. SecureCRT then uploads the public key file by way of the public-key assistant. Uploads are only possible on servers that support the public-key assistant (such as VanDyke Software's VShell® 2.1 or newer server).

Once a public key for a session has been uploaded to a server, further session connections to that server can be made using public-key authentication.

# The Toolbar

The SecureCRT toolbar is user-manageable in that the user can toggle its display on or off. When enabled, the toolbar appears below the SecureCRT menu bar. Move the mouse cursor over a toolbar button to view its associated action. The toolbar is enabled by default. The toolbar can be <u>customized</u> to suit the preferences and needs of the user.

The table below depicts the toolbar buttons and their associated tasks.

Button	Description
¥67	Connect: Opens the connect dialog
Ġ.	Connect (Connect dialog button): Connects to selected session
<b>₩</b>	<b>Quick Connect</b> : Brings up the Quick Connect dialog
	Connect in Tab/Tile: Opens the Connect in Tab/Tile dialog
<u> </u>	New Session (Connect dialog button): Opens the Session Options dialog for a new session
<b>\$</b>	Reconnect: Reconnects last connection
*	Disconnect : Closes the current connection
	Connect Bar: Allows you to enter and open a session in a new tab or window.
	<b>Copy</b> : Copies the selected text to the Clipboard
	Paste: Pastes the contents of the Clipboard

	· · · · · · · · · · · · · · · · · · ·
<b>₽</b> A	<b>Find</b> : Searches the contents of the scrollback buffer and terminal screen for a given string
4	Print Screen: Prints the text currently displayed in the terminal window
	Print Selection: Prints the selected text
<b>5</b>	Auto Print: Toggles Auto Print on or off
<b>**</b>	Properties: Brings up the Session Options dialog
<b>%</b>	Keymap Editor: Invokes the Keymap Editor
9	Public-Key Assistant: Opens the Public-Key Assistant dialog
0	Help: Invokes the online help manual
æ	SecureFX: Starts VanDyke Software's SecureFX® (if the program is installed)

# Session Tabs

SecureCRT allows you to display multiple sessions either in separate application windows or in one window using session tabs. You turn on tabs by selecting the **Session Tabs** option on the <u>view</u> menu or by enabling the <u>Show session tabs</u> option on the <u>Global Options/Terminal/Appearance</u> dialog.

When session tabs are enabled, a separate tab will be displayed for each open session. To display a new session tab, click on the **Connect in Tab/Tile** toolbar button or open the <u>File</u> menu and select **Connect in Tab/Tile...**. This will open the **Connect in Tab/Tile** dialog allowing you to select from a list of your existing sessions.

**Note:** You can still open sessions in separate windows by using the **Connect** button to start other sessions.

Tabs can be moved using drag-and-drop operations and can be closed by pressing CTRL+F4.

Tabs can be renamed by right-clicking on the name on the session tab and selecting **Rename**. The name change remains in effect until the tab is reused or until SecureCRT exits.

You can also "clone" a tabbed session by right-clicking on the session and selecting the **Clone Session** option or by opening the **File** menu and selecting the **Clone Session** option. This will create a session that is identical to the current session. If the original session is using the SSH2 protocol, the transport is shared and re-authentication is not needed. For other protocols, authentication is required.

When working with tabs, the first session always dictates geometry and font choices of the other sessions regardless of the actual settings for size and font in the other sessions. Tabs are automatically sized if they do not fit in the SecureCRT window.

### **SFTP sessions**

If you have an SSH2 session open, you can also open an SSH file transfer protocol (SFTP) session using the same transport as the current SSH2 session. To do this, open the SecureCRT <u>File</u> menu and select the **Connect SFTP Session** option (you can also access this option from the

right-click context menu). This will open a new tab or tiled session with a SFTP command line.

Drag-and-drop transfers are supported in the SFTP tab. Single or multiple files can be dragged from Microsoft Explorer (or other OLE-enabled applications) into the SFTP tab, which cause a put command to be issued for the file(s).

SecureCRT also supports tab completion of command arguments on the SFTP command line. Begin typing a command and argument and then press the TAB key. SecureCRT will attempt to complete the argument(s) based on the available choices. If there are multiple choices, SecureCRT will display a list of possible options. You can choose from this list or continue typing until SecureCRT can identify the unique argument.

See **SFTP** Command-Line Options for a complete listing of available options.

# The Connect Bar

The SecureCRT connect bar provides a command-based method that allows you to enter and connect to a session or sessions.

The connect bar is accessed via the toolbar or by typing the ALT+R key combination. The connect bar can be displayed or hidden by selecting or deselecting the **Connect Bar** option on the <u>view</u> menu.

Pressing CTRL+ENTER in the connect bar will open the specified session in a new window.

# **Syntax Examples**

The following is an example of how to open a new connection. The host can be an IP address or a named host. If only the host is specified, the default protocol and port will be used.

```
[<protocol>://][<user>[:<password>]@]<host>[:<port>]
```

The following is an example of how to open a specific session.

```
/S <session name>
```

The following is an example of how to open connections to the specified sessions in two tabs (in the current window or a new window).

```
/S <session name> /S <session name>
```

The following is an example of how to open a connection to <host> using <firewall>

```
/FIREWALL=<firewall> ssh2://[<user>]@<host>
```

Using SecureCRT's dependent session option on the command line or in the connect bar (as in the following example), it's possible to link a session to an SSH2 session that it depends on, which allows connection to a jump host before connecting to other sessions.

Both the command line and the connect bar use the normal /Firewall specifier and the firewall will be Session: <session name> with the following syntax.

```
/FIREWALL=Session:<Session Name>
ssh2://[<user>]@<host>
```

Session: is case insensitive.

# The Button Bar

The SecureCRT button bar is a user-configurable row of buttons that you can use to quickly access frequently needed actions and commands. You can toggle the button bar display on or off by selecting or unselecting the Button Bar option on the view menu, or by selecting or unselecting the Show button bar option on the Terminal/Appearance/Window Category of the Session Options dialog or the Terminal/Appearance Category of the Global Options dialog.

**Note:** To perform an action, which has been mapped to a button, on all connected sessions, use the SHIFT+<click> combination. If the mapped action is Launch Application, only one instance of the application will be launched.

The button bar assigned to the default session is used for all new and Quick connect sessions. The default button bar is normally assigned to the default session, but this can be changed by editing the Default Session category in the **Global Options** dialog.

Note: The default button bar can be edited, but cannot be deleted.

To create a new button bar, right-click on any part of the button bar to open the context menu and then select the **New Button Bar...** option. This opens the **New Button Bar** dialog which allows you to add a named button bar to the global list of button bars available to SecureCRT. A button bar can be assigned to a session in the Terminal/Appearance/Window category of the Session Options dialog, or by selecting a different button bar in the button bar drop-down list while the session is active.

To add a button to the button bar, right-click anywhere in the button bar and select the **New Button...** option; this will open the Map Button dialog. To change an existing button, right-click on the button you want to change and choose the **Edit Button...** option. Each button on a bar can be assigned one action. The list of actions available is the same as the list of actions available for keymap key assignments and includes Send String, Run Script, Menu, Scroll Bar, Telnet, SSH, and VT function genres. You can rearrange the buttons on the button bar by right-clicking on a button and selecting the Move Button Left or Move Button Right option.

**Note:** In order to display an ampersand character (&) in a button bar, it must be preceded by a second ampersand. For example, to label a

button "A & B", the text entered in the **Label** field of the **Map Button Bar** dialog should be "A && B".

You can also create a button that launches local commands. By assigning a button to the function Launch Application and entering the appropriate executable file, SecureCRT can open whatever application you need.

**Note:** Some actions might fail or be ignored if the active session is not connected or is in a state where the action cannot be performed.

To remove a button from a button bar, right-click the button that you want to delete and select **Delete Button** from the context menu.

To delete a button bar, select the button bar you want to delete in the button bar drop-down list (making it the active button bar for the session), right-click anywhere in the button bar, and choose the **Delete Button Bar** option.

Assignments of keys in keymaps and of buttons in button bars are independent of each other; i.e., assigning a key to an action does not affect any assignments or labels in button bars, but you can edit your keymap and button bars separately to have corresponding settings.



The SecureCRT status bar (shown below) is made up of the following fields (as they appear from left to right):

Ready ssh2: AE5-128 1, 22 18 Rows, 69 Cols | Xterm | 02:17:05 | CAP | NUM | Status Bar

### Message

The messages include the following: the default "Ready"; short definitions of highlighted toolbar buttons and menu commands; and the coordinates when you resize the SecureCRT window or select an area within the window.

### **Protocol**

The protocol that is being used by the current  $\underline{session}$ . These protocols include  $\underline{ssh1}$ ,  $\underline{ssh2}$ ,  $\underline{Telnet}$ ,  $\underline{Telnet}$ ,  $\underline{RLogin}$ , and  $\underline{Serial}$ .

### **Cursor Location**

The row and column (row, column) in which the <u>terminal</u> cursor currently resides.

## **Rows Configured**

The number of rows to be displayed in the SecureCRT window (as configured in the <u>Session Options</u> dialog).

# **Columns Configured**

The number of columns displayed in the SecureCRT window (as configured in the <u>Session Options</u> dialog).

### **Emulation**

The type of terminal that SecureCRT is currently emulating. These emulations include ANSI, Linux, SCO ANSI, VT100, VT102, and VT220.

# **Elapsed time**

Indicates elapsed connection time for the session if the **Show elapsed time** option is enabled.

# Caps Lock

Indicates whether the keyboard Caps Lock control is on or off.

#### Num Lock

Indicates whether the keyboard Num Lock control is on or off.

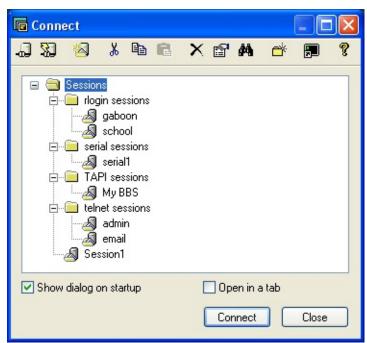


# **Arranging Entries in the Connect Dialog**

Sessions listed in the **Connect** dialog's session tree can be arranged alphabetically (i.e., A1, A10, A2, B), logically (i.e., A1, A2, A10, B), or manually.

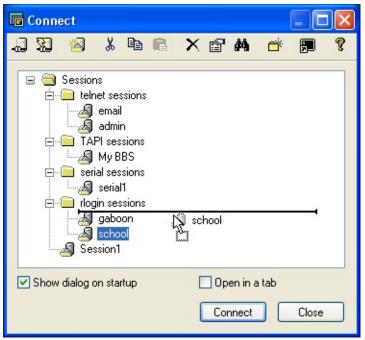
You can select the order by which you want to sort the session tree by using the <u>Connect dialog sort order</u> option in the <u>Global Options / General</u> dialog, or by right-clicking on any entry listed in the <u>Connect</u> dialog and choosing the **Arrange Tree** option from the resulting context menu. Changing the sort order with the **Arrange Tree** option will also change the **Connect dialog sort order** and vice versa.

By default, sessions are listed in alphabetical order as depicted in the sample sessions and folders listed in the **Connect** dialog shown in *Connect Dialog: Arranged In Alphabetical Order* below.



Connect Dialog: Arranged In Alphabetical Order

By choosing to manually arrange the entries in the session tree, you can easily move individual sessions or even entire folders using drag-and-drop operations. In *Connect Dialog: Manual Arrangement Using Drag-and-Drop Operations* below, the user is in the process of manually arranging the *school* session to be listed before or above the *gaboon* session in the *rlogin sessions* folder using drag-and-drop operations.



Connect Dialog: Manual Arrangement Using Drag-and-Drop Operations



# Copying and Pasting

SecureCRT provides you with a variety of ways to copy and paste text to and from the terminal screen. This section covers the basics on accomplishing the following tasks:

Selecting Text in the Terminal Window

**Customizing Word Delimiter Characters** 

Copying Text to the Clipboard

Pasting Text from the Clipboard to the Terminal Screen

Copy on Select

Middle/Right Button Paste

Copying and Pasting with CTRL+C and CTRL+V (CUA Copy and Paste)

# Selecting Text in the Terminal Window

There are a variety ways to select text in the SecureCRT terminal screen.

### Selecting Rows of Text

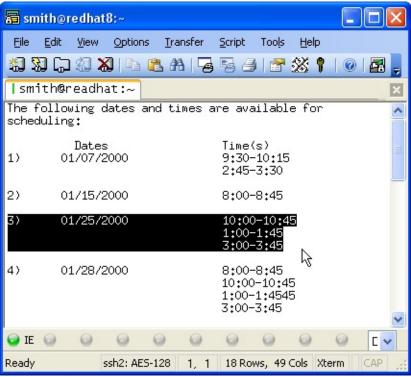
This method involves moving the mouse pointer across the words or rows of text you want selected while holding down the left mouse button. This is the default method for selecting text (see Selecting Rows of Text in the Terminal Screen below).

# **Selecting Columns (blocks) of Text**

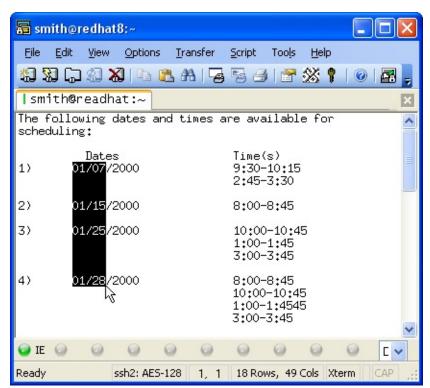
This method is useful for selecting columns of text rather than rows of text. To select a block of text made up of one or more columns of text on the terminal screen, hold down the ALT key while moving the mouse pointer down the column(s) you want selected while holding down the left mouse button (see Selecting Columns of Text in the Terminal Screen while Holding Down the ALT Key below).

### Selecting an Entire "Word" of Text

Another method of selecting text makes selecting an entire word easy. You can select an entire word just by positioning the mouse cursor over the word and then double-clicking on the word with the left mouse button. In addition to being able to select an entire word with a mere double-click, SecureCRT provides a way for you to customize the word delimiter characters on a per-session basis.



Selecting Rows of Text in the Terminal Screen



Selecting Columns of Text in the Terminal Screen while Holding Down the ALT Key

# **Modifying a Selection**

You can modify an existing selection in the terminal screen. To modify the current selection, hold down the SHIFT key, move the mouse pointer to the new desired start or end point, and press the left mouse button.

# **Customizing Word Delimiter Characters**

By default, SecureCRT uses all non-alphanumeric characters (except underscore, "\_") as word delimiters when selecting an entire "word" with a double-click action as described in the "Selecting Text in the Terminal Window" section of this help topic.

You can modify the word delimiter characters using the following steps:

- 1. In the <u>session Options</u> dialog, select the <u>Terminal</u> category.
- 2. In the **Terminal** panel, select the **Set word delimiter characters** option.
- 3. Enter your personal word delimiter characters. Do not separate the entries with spaces or any other character.

Note: A "space" character can be used as a delimiter character.

By default, SecureCRT uses all non-alphanumeric characters (except underscore, "\_") as delimiters when selecting a word. This includes punctuation such as ., !?()", but also  $\#/\$ . You might want some of the second character set ( $\#/\$ ) to be excluded from the word delimiter characters so that you can select currency amounts, path names, email addresses, and the like.

### **Copying Text to the Clipboard:**

To copy text from the terminal screen to the Clipboard:

- 1. Select the text you want to copy by dragging the mouse cursor over the desired area of text.
- 2. Select **Copy** from the **Edit** menu on the SecureCRT menu bar <u>or</u> click on the **Copy** button located on the SecureCRT toolbar.

The selected text is now copied to the Clipboard and ready for pasting.

The format of the data saved to the Clipboard cannot be changed in the SecureCRT user interface, but can be changed manually by editing the Global.ini file in the Config folder. In the Global.ini file, the **Clipboard** 

**Data Format** option can be set to CF\_TEXT, CF\_0EMTEXT, CF\_UNICODETEXT, DEFAULTFORMAT, or VDS\_TEXT.

**Note**: When copying to the Clipboard, SecureCRT always translates DEC line-drawing characters to the four characters: "+", "-", "\_", and "|". If the **Copy translates ANSI line drawing characters** option is enabled on the **Session Options/Terminal/Emulation/Advanced** dialog, SecureCRT also translates ANSI line-drawing characters to the three characters: "+", "-", and "|".

# **Pasting Text from the Clipboard to the Terminal Screen:**

To paste text from the Clipboard to the terminal screen:

Select **Paste** from the **Edit** menu on the SecureCRT menu bar <u>or</u> click on the **Paste** button **SecureCRT** located on the SecureCRT toolbar.

The contents of the Clipboard will be inserted at the point of the text cursor in the terminal screen.

SecureCRT also provides a feature that allows you to copy and paste from the terminal screen back to the terminal screen with just one step. Select the text you would like to paste, and then select **Copy & Paste** from the <u>Edit</u> menu on the SecureCRT menu bar to paste the selected text at the cursor or insertion point.

# **Copy on Select**

The **Copy on select** feature allows you to speed up the process of copying text from the terminal screen to the Clipboard. If this feature is enabled, text is copied to the Clipboard as soon as it is selected.

To enable/disable the **Copy on select** feature:

- 1. Select the <u>Terminal</u> category in the <u>Global Options</u> dialog.
- 2. In the **Mouse** group, select or unselect the **Copy on select** option.

# Middle/Right Button Paste

The **Paste on middle/right button** feature allows you to speed up the process of pasting content from the Clipboard to the terminal screen. If this feature is enabled, text is pasted from the Clipboard to the terminal screen as soon as the middle or right mouse button (as specified in the <a href="Global Options/Terminal">Global Options/Terminal</a> dialog) is pressed. If your mouse does not have a

middle mouse button, a middle button click can be simulated by pressing both the left and right mouse buttons simultaneously.

To enable/disable the **Paste on middle/right button** feature:

- 1. Select the <u>Terminal</u> category of the <u>Global Options</u> dialog.
- 2. In the **Mouse** group, select or unselect the **Paste on middle/right button** option.

# Copying and Pasting with CTRL+C and CTRL+V (CUA Copy and Paste)

To enable the CTRL+C, CTRL+V, CTRL+A, and CTRL+F accelerator keys for copy, paste, select all, and find, respectively:

- 1. Select the Terminal/Emulation/Mapped Keys Category in the Session Options dialog.
- 2. Select the **Use Windows copy and paste hotkeys** option.
- 3. A message box will appear with the following warning:

**WARNING:** Turning on **Use Windows copy and paste hotkeys** will disable your ability to send CTRL+C, CTRL+V, CTRL+A, and CTRL+F to the remote host. Most UNIX users should leave this option off.

- 4. Click on the **OK** button on the **Warning** dialog.
- 5. Click on the **OK** button at the bottom of the <u>Session Options</u> dialog.



# Customizing the User Interface

SecureCRT allows you to customize some parts of its user interface in two different ways: either by using the customize dialog, or by editing the menu configuration file (described below). Using either of these methods, you can modify or remove existing menus, rearrange the main toolbar buttons, and much more, however, you must choose one or the other method, as they cannot be used together.

# **Customizing the Menu Configuration File**

**WARNING:** Customizing the SecureCRT interface by editing the menu configuration file can be a complex undertaking. Some programming knowledge will be helpful.

The customizable interface features are listed below:

- Menu titles on the main window
- · Menu options on the main window menus, the system menu, and right-click menus
- Accelerators
- **Toolbar buttons**
- **ToolTips**
- Status bar messages

To make changes to the items mentioned above, you will need to edit the SecureCRT menu configuration file found in the SecureCRT data folder. The default configuration file is Default.mnu, but other files can be used by changing the file specified in the Custom menu and toolbar file option in the **Terminal** category of the **Global Options** dialog.

**Tip:** To return SecureCRT to its default configuration, either delete any existing configuration files or clear the entry in the Custom menu and toolbar file entry box, and restart SecureCRT. This will force SecureCRT to return to default values.

# **Configuration File Format**

This section covers the format and syntax of the menu and toolbar configuration file. The precedence for accelerator keys is F-Keys, Keymap, Menultem, built in.

### File format rules

The fields in a definition will be separated by a comma. Command arguments will be separated by a vertical bar or pipe (|).

Accelerator definitions will start with a unique string identifier. They will not contain a field for the command ID string, as that already exists in the associated menu or toolbar item definition.

String table definitions will start with a unique string identifier. They will not contain a field for the command ID string as that already exists in the associated menu or toolbar item definition.

Accelerator and string table definitions must appear in the file before any menu or toolbar definitions.

### **Command strings**

The SecureCRT menu configuration file supports command strings for most of the application commands. See the Menu Configuration Command Strings topic for a comprehensive list.

In addition to standard menu and toolbar commands, three user-defined commands are also available:

- OPEN\_SESSION -- This command connects you to the specified session .
- OPEN\_SESSION\_TAB -- This command connects you to the specified session in a tab.
- RUN -- This command runs the specified script. This command does not support arguments.
- SEND -- This command sends the specified string.

The above command strings can be used as in the following examples extracted from the example file at the end of this topic:

```
TOOLBARITEM "server.bmp", "OPEN_SESSION", "server", "server", "server"

TOOLBARITEM "run.bmp", "RUN", "C:\vss\test.vbs", "script"

TOOLBARITEM "send.bmp", "SEND", "Hello World\n", "hello"
```

### **Keywords and syntax**

Items enclosed in angle brackets (<>) are required. Items enclosed in square brackets ([]) are optional.

```
END (ends a definition of a menu, toolbar, popup menu, accelerator
keys, or string table)
MENU (for main menu definitions)
  MENU <"label">
  <definition of menu items or submenus>
  END
TOOLBAR (for toolbar definitions)
  TOOLBAR <"label">
  <definition of buttons in the toolbar>
  END
POPUPMENU (for context and system menu as well as sub-menus)
  POPUPMENU <"label">
  <definition of menu items or submenus>
ACCELERATOR (for the definition of accelerator keys)
  ACCELERATOR
  <definition of accelerator keys>
  END
  An accelerator key definition is of the form:
     <"unique identifier">,<"key">,<"key modifiers separated by '|' ">
STRINGTABLE (for the definition of status bar and ToolTip strings)
  STRINGTABLE
  <definition of status bar and tooltip strings>
  END
A string table definition is of the form:
     <"unique identifier">,<"status bar message\ntooltip">
SEPARATOR (indicates that a separator should be placed in the
menu or toolbar )
MENUITEM (designates that the following text defines a menu item)
```

```
MENUITEM <"label">, <"command str">[," command
argument(s)"][,"string table identifier"]
[,"accelerator identifier"]
```

TOOLBARITEM (designates that the following text defines a toolbar button)

```
TOOLBARITEM <"icon">,<"command str">[,"command argument(s)"][,"string table identifier"]
[,"accelerator identifier"]
```

```
// (designates a comment)
```

```
// [comment string]
```

Any line that has // as the first two characters will be considered as a comment. Comments must be on a line by themselves.

For MENUITEM, the "label" and "command str" are required. The remaining arguments are optional. However, order is important. If an optional argument is not provided and it is followed by an argument that is provided, the argument separator (,) must be included. For example:

```
MENUITEM &MyItem, MY_MENU_COMMAND,, "This is my menu item"
```

The above arguments are: <"label">,<"command str">, , <"stringtable identifier">

For TOOLBARITEM, one of "icon" and "command str" is required. Again order is important and skipped arguments must have the argument separator included.

### **Errors and warnings**

When an error condition occurs, you will be informed about the nature of the error and that the default menu or toolbar will be loaded and displayed.

When a warning condition occurs, you will be informed about what caused the warning condition and that the particular menu or toolbar item, accelerator, or hint will not be used. However, those parts of the menu or toolbar that are correct will be loaded and displayed.

# **Example file**

The following example file is the default menu and toolbar configuration file Default.mnu.

```
//*** This file will be overwritten when SecureCRT is upgraded. ***
//*** If you wish to modify the file, please make your changes in a copy.
// accelerator table
ACCELERATOR
  "Connect", "C", "VIRTKEY | ALT"
  "Connect in Tab", "B", "VIRTKEY | ALT"
"New SFTP Tab", "P", "VIRTKEY | ALT"
"Quick", "Q", "VIRTKEY | ALT"
  "Go to Chat Window", "G", "VIRTKEY | ALT"
END
// stringtable for status bar and tooltips
STRINGTABLE
  "Connect", "Connect to a remote host in a new
  window\nConnect"
  "Connect in Tab", "Connect to a remote host in a
  new tab\nNew Tab"
  "Clone Session", "Clone current session\nClone
  Session"
  "New SFTP Tab", "Connect new SFTP tab using
  current session\nSFTP"
  "Lock Session", "Lock/Unlock current
  session\nLock/Unlock Session"
  "Quick", "Quick connect\nQuick Connect"
  "Go to Chat Window", "Go to the chat window or
  session window\nGo to Chat Window"
END
// main menu
MENU "[main]"
  POPUPMENU "&File"
    MENUITEM "&Connect...", "MENU CONNECT", ,"Connect",
    "Connect"
```

```
MENUITEM "&Quick Connect...", "MENU QUICK CONNECT",
   ,"Quick", "Quick"
  MENUITEM "Connect in Ta&b...", "MENU TAB NEW",
   ,"Connect in Tab", "Connect in Tab"
  SEPARATOR
  MENUITEM "&Reconnect", "MENU RECONNECT"
  MENUITEM "Reconnect &All", "MENU RECONNECT ALL"
  MENUITEM "&Disconnect", "MENU DISCONNECT"
  MENUITEM "Disc&onnect All", "MENU DISCONNECT ALL"
  SEPARATOR
  MENUITEM "Clo&ne Session", "MENU TAB CLONE", ,"Clone
  Session",
  MENUITEM "Connect &SFTP Tab", "MENU TAB NEW SFTP",
   ,"New SFTP Tab", "New SFTP Tab"
  SEPARATOR
  MENUITEM "Loc&k Session...", "MENU TAB LOCK", ,"Lock
  Session",
  SEPARATOR
  POPUPMENU "&Print"
    MENUITEM "&Auto Print", "MENU_PRINT_AUTO"
    MENUITEM "&Screen", "MENU_PRINT_SCREEN"
    MENUITEM "Se&lection", "MENU_PRINT_SELECTION"
    MENUITEM "&Eject page",
    "MENU_PRINT_EJECT PAGE"
    MENUITEM "&Cancel", "MENU_PRINT_CANCEL"
  END
  MENUITEM "Print Set&up...", "MENU PRINT SETUP"
  SEPARATOR
  MENUITEM "&Log Session", "MENU LOG SESSION"
  MENUITEM "Ra&w Log Session", "MENU DEBUG LOG"
  MENUITEM "&Trace Options", "MENU TRACE OPTIONS"
  SEPARATOR
  MENUITEM "Recent Session", "MENU MRU FILE1"
  SEPARATOR
  MENUITEM "E&xit", "MENU EXIT"
END
POPUPMENU "&Edit"
  MENUITEM "&Copy", "MENU COPY"
```

```
MENUITEM "&Paste", "MENU PASTE"
  MENUITEM "C&opy && Paste", "MENU COPY PASTE"
  MENUITEM "Paste as &Quotation",
  "MENU EDIT PASTE AS QUOTATION"
  MENUITEM "Select &All", "MENU SELECT ALL"
  MENUITEM "&Find...", "MENU FIND"
  SEPARATOR
  MENUITEM "Print Se&lection", "MENU PRINT SELECTION"
  SEPARATOR
  MENUITEM "&Go to Chat Window",
  "MENU GOTO CHAT SESSION WINDOW"
  SEPARATOR
  MENUITEM "Clear Scroll&back",
  "MENU CLEAR SCROLLBACK"
  MENUITEM "Clear &Screen", "MENU CLEAR SCREEN"
  MENUITEM "Clear Screen && Scrollbac&k",
  "MENU CLEAR SCREEN AND SCROLLBACK"
  SEPARATOR
  MENUITEM "&Reset", "MENU RESET"
END
POPUPMENU "&View"
  MENUITEM "&Menu Bar", "MENU TOGGLE MENU BAR"
  MENUITEM "&Toolbar", "MENU TOGGLE TOOLBAR"
  MENUITEM "Session Ta&bs",
  "MENU TOGGLE SESSION TABS"
  MENUITEM "&Chat Window",
  "MENU TOGGLE CHAT WINDOW"
  MENUITEM "Connect Ba&r",
  "MENU TOGGLE CONNECT BAR"
  MENUITEM "&Status Bar", "MENU TOGGLE STATUS BAR"
  MENUITEM "&Button Bar", "MENU TOGGLE BUTTON BAR"
  POPUPMENU "Th&emes"
    MENUITEM "&Shaded", "MENU_THEMES_SHADED"
    MENUITEM "Office 200&0",
    "MENU THEMES OFFICE2000"
    MENUITEM "Office 200&3",
    "MENU_THEMES_OFFICE2003"
    MENUITEM "&Blue", "MENU_THEMES_BLUE"
```

```
MENUITEM "Blac&k", "MENU_THEMES_BLACK"
    MENUITEM "&Aqua", "MENU_THEMES_AQUA"
    MENUITEM "S&ilver", "MENU_THEMES_SILVER"
  END
  SEPARATOR
  MENUITEM "&Window Transparency",
  "MENU TOGGLE ALPHA TRANSPARENCY"
  SEPARATOR
  MENUITEM "&Horizontal Scroll Bar",
  "MENU TOGGLE HORZ SCROLLBAR"
  MENUITEM "&Vertical Scroll Bar",
  "MENU TOGGLE VERT SCROLLBAR"
  SEPARATOR
  MENUITEM "&Always on Top", "MENU ALWAYS ON TOP"
  MENUITEM "&Full Screen", "MENU TOGGLE FULL SCREEN"
END
POPUPMENU "&Options"
  MENUITEM "&Session Options...",
  "MENU SESSION OPTIONS"
  MENUITEM "&Global Options...", "MENU GLOBAL OPTIONS"
  SEPARATOR
  MENUITEM "&Auto Save Options", "MENU AUTO SAVE"
  MENUITEM "Save Settings & Now", "MENU SAVE NOW"
END
POPUPMENU "&Transfer"
  MENUITEM "&Send ASCII...", "MENU SEND ASCII"
  MENUITEM "&Receive ASCII...", "MENU RECEIVE ASCII"
  SEPARATOR
  MENUITEM "Send &Binary...", "MENU SEND BINARY"
  SEPARATOR
  MENUITEM "Se&nd Xmodem...", "MENU SEND XMODEM"
  MENUITEM "Re&ceive Xmodem...",
  "MENU RECEIVE XMODEM"
  SEPARATOR
  MENUITEM "Sen&d Ymodem...", "MENU SEND YMODEM"
  MENUITEM "Recei&ve Ymodem...",
  "MENU RECEIVE YMODEM"
```

```
SEPARATOR
  MENUITEM "&Zmodem Upload List...",
  "MENU ZMODEM ADD FILE"
  MENUITEM "Start Zmodem & Upload",
  "MENU SEND ZMODEM"
END
POPUPMENU "&Script"
  MENUITEM "&Run...", "MENU_SCRIPT_RUN"
  MENUITEM "&Cancel", "MENU SCRIPT CANCEL"
  SEPARATOR
  MENUITEM "&Start Recording Script",
  "MENU_SCRIPT_START_RECORDING"
  MENUITEM "S&top Recording Script..."
  "MENU SCRIPT STOP RECORDING"
  MENUITEM "C&ancel Recording Script",
  "MENU SCRIPT CANCEL RECORDING"
  SEPARATOR
  MENUITEM "Recent Script", "MENU SCRIPT MRU FILE1"
END
POPUPMENU "Too&ls"
  MENUITEM "&Keymap Editor...",
  "TOOLS MENU KEYMAP EDITOR"
  SEPARATOR
  MENUITEM "&Create Public Key...",
  "TOOLS CREATE PUBLIC KEY"
  MENUITEM "Convert Private Key to & OpenSSH Format...",
  "TOOLS CONVERT PRIVATE KEY"
  MENUITEM "&Export Public Key from Certificate...",
  "TOOLS EXPORT PUBLICKEY FROM CERTIFICATE"
  MENUITEM "&Public-Key Assistant...".
  "TOOLS PUBLIC KEY ASSISTANT"
  SEPARATOR
  MENUITEM "Ma&nage Agent Keys...",
  "TOOLS MANAGE AGENT KEYS"
END
POPUPMENU "&Help"
```

```
MENUITEM "&Help Topics", "MENU HELP FINDER"
    SEPARATOR
    MENUITEM "SecureCRT &Web Page...",
    "MENU HELP WEBPAGE"
    MENUITEM "&Order SecureCRT...", "MENU HELP ORDER"
    MENUITEM "Check for &Updates...", "MENU_HELP_UPDATE"
    MENUITEM "VanDyke Software & Forum...",
    "MENU FORUMS WEBPAGE"
    SEPARATOR
    MENUITEM "&Enter License Data...",
    "MENU ENTER LICENSE DATA"
    SEPARATOR
    MENUITEM "&About SecureCRT...", "MENU APP ABOUT"
  END
END
POPUPMENU "[context]"
  MENUITEM "&Copy", "MENU_COPY"
MENUITEM "&Paste", "MENU_PASTE"
  MENUITEM "Copy && Paste", "MENU_COPY_PASTE"
  MENUITEM "Paste as &Quotation",
   "MENU EDIT PASTE AS QUOTATION"
  SEPARATOR
  MENUITEM "&Open URL", "MENU_OPEN_URL"
  SEPARATOR
  MENUITEM "&Find...", "MENU_FIND"
  MENUITEM "Select &All", "MENU_SELECT_ALL"
  MENUITEM "Print Se&lection",
  "MENU_PRINT_SELECTION"
  SEPARATOR
  MENUITEM "Clear Scroll&back",
  "MENU CLEAR SCROLLBACK"
  MENUITEM "Clear &Screen", "MENU_CLEAR_SCREEN"
  MENUITEM "Clear Screen && Scrollbac&k",
  "MENU CLEAR SCREEN AND SCROLLBACK"
END
POPUPMENU "[system]"
  MENUITEM "&Restore", "SYS_MENU_RESTORE"
```

```
MENUITEM "&Move", "SYS_MENU_MOVE"
  MENUITEM "&Size", "SYS_MENU_SIZE"
  MENUITEM "Mi&nimize", "SYS_MENU_MINIMIZE"
  MENUITEM "Ma&ximize", "SYS_MENU_MAXIMIZE"
  SEPARATOR .
  MENUITEM "&Always on Top", "MENU_ALWAYS_ON_TOP"
  MENUITEM "Save Settings No&w", "MENU_SAVE_NOW"
  MENUITEM "&Toggle Menu Bar",
  "MENU TOGGLE MENU BAR"
  SEPARATOR
  MENUITEM "&Close", "SYS_MENU_CLOSE"
END
// the toolbar
TOOLBAR "[toolbar]"
  TOOLBARITEM "MENU_CONNECT", "MENU_CONNECT", ,
  "Connect", "Connect"
  TOOLBARITEM "MENU_QUICK_CONNECT",
  "MENU_QUICK_CONNECT", , "Quick", "Quick"
  TOOLBARITEM "MENU_TAB_NEW", "MENU_TAB_NEW", ,
  "Connect in Tab", "Connect in Tab"
  TOOLBARITEM "MENU_RECONNECT", "MENU_RECONNECT"
  TOOLBARITEM "MENU_DISCONNECT", "MENU_DISCONNECT"
  TOOLBARITEM "CONNECT_BAR", "CONNECT_BAR"
  SEPARATOR
  TOOLBARITEM "MENU_COPY", "MENU_COPY"
  TOOLBARITEM "MENU_PASTE", "MENU_PASTE"
TOOLBARITEM "MENU_FIND", "MENU_FIND"
  SEPARATOR 
  TOOLBARITEM "MENU_PRINT_SCREEN",
  "MENU PRINT SCREEN"
  TOOLBARITEM "MENU PRINT SELECTION",
  "MENU PRINT SELECTION"
  TOOLBARITEM "MENU_PRINT_AUTO", "MENU_PRINT_AUTO"
  SEPARATOR
  TOOLBARITEM "MENU_SESSION_OPTIONS",
  "MENU SESSION OPTIONS"
  TOOLBARITEM "TOOLS_MENU_KEYMAP_EDITOR",
  "TOOLS MENU KEYMAP EDITOR"
```

```
TOOLBARITEM "TOOLS_PUBLIC_KEY_ASSISTANT",

"TOOLS_PUBLIC_KEY_ASSISTANT"

SEPARATOR

TOOLBARITEM "MENU_HELP_FINDER", "MENU_HELP_FINDER"

SEPARATOR

TOOLBARITEM "TOOLBAR_LAUNCH_SECUREFX",

"TOOLBAR_LAUNCH_SECUREFX"

END
```



# **Keyboard Accelerators**

This page lists keyboard accelerators (shortcut keys) that are supported in SecureCRT. The accelerators are as follows:

- CTRL+TAB takes you through multiple SecureCRT session tabs.
- CTRL+SHIFT+TAB cycles backwards through the sessions.
- CTRL+F4 closes the active tab.
- ALT+1 (2, 3, 4, 5, 6, 7, 8, 9, & 0) allows you to jump directly to one
  of the first ten tabs.
- ALT+B opens the Connect in Tab/Tile dialog.
- ALT+C opens the Connect dialog.
- ALT+G switches the focus between the chat window and the active session.
- ALT+P opens an SFTP tab using the active tab's session.
- ALT+Q opens the Quick Connect dialog.

Within the chat window, CTRL+SHIFT+TAB switches the focus to the active session.

For the advanced user, the <u>Table of Keymap Functions</u> lists keyboard menu functions that you can link to keystrokes using the <u>Keymap Editor</u>.



# **Switching Between Multiple Sessions**

There are several ways to switch between multiple SecureCRT sessions

.

- If you enable the <u>Show session tabs</u> option on the <u>Global</u>
   <u>Options/Terminal/Appearance</u> dialog, you can switch between sessions by clicking on the tabs in the SecureCRT window.
- If you are using the Minimize to Activator in system tray option, right-click on the Activator icon in the system tray and select the SecureCRT session window that you want to open from the pop-up menu.
- You can also use the CTRL+TAB keyboard shortcut to take you through multiple SecureCRT session windows, while CTRL+SHIFT+TAB will take you backwards through the sessions.
- Using ALT+1 through ALT+0 will allow you to directly navigate to any of the first ten tabs.

# Using the Activator

The Activator allows you to minimize specific sessions or all sessions to the system tray instead of the taskbar. This can be useful if you have a number of sessions running or if you only use SecureCRT to port forward vour connections to another machine.

The Activator also supports SSH agent functionality, so that if SecureCRT is closed, authentication will happen automatically as long as the Activator is still running. If you would like to use the Activator's SSH agent functionality but do not want to minimize to the Activator. the Activator can be added to the Windows **Start** menu or can be started manually.

## **Enabling the Activator**

The Activator can be enabled as a global option (affecting all sessions). To enable the Activator option for all sessions, complete the following steps:

- 1. Select **Global Options...** from the options menu.
- 2. Click on the **General** category.
- 3. Check the Minimize to Activator in the system tray check box to turn on the Activator. This option is off by default.
- 4. Click the **OK** button to save your changes.

### **Restoring Sessions**

To restore sessions that have been minimized to the system tray, rightclick on the Activator icon in the tray and choose the session that you want to restore from the resulting menu.

### **Managing Agent Keys**

In the Activator menu, selecting the **Manage Agent Keys...** command will open the **Manage Agent Keys** dialog which allows you to add keys to and remove keys from the list of accepted agent keys.



# Overview of Command-Line Options

### **Command-Line Options**

There are two categories of SecureCRT command-line options: standard options, and protocol-specific options.

Standard options may be used in any combination with other standard options.

Protocol-specific options are mutually exclusive; a protocol-specific option cannot be used in combination with any other protocol-specific option.

#### Syntax:

SecureCRT.exe [standard options] [protocol-specific | You may find it helpful to see the examples of using command-line options.



# Table of Standard Command-Line Options

Standard Option	Arguments	Description
/ARG	argument	Passes option(s) to SecureCRT scripts by way of Arguments scripting objects. More than one /ARG option can be used in a command.
/COLOR_SCHEME	scheme	Specifies an existing color scheme to be used for the session being started. When this switch is used, any changes to the session color scheme will be ignored. If a nonexistent scheme is specified, the session configuration color scheme will be used.
/F	folder pathname	Points to an alternate configuration folder location. If there is no configuration file in the alternate folder, SecureCRT will create one. Without this option, SecureCRT will use the configuration information in the default folder location.
/FIREWALL	firewall name	Connects session using specified firewall settings. Replaces /PROXY and /SOCKS options. The format of this command is as follows:     /FIREWALL= <name> If the firewall name contains spaces, it should be enclosed by quotation marks.</name>

If no firewall argument is specified, SecureCRT will use the firewall specified by the session being connected. If the session firewall is set to None, SecureCRT will use the firewall specified in the current global firewall settings. If a firewall is named in the command argument, SecureCRT will temporarily override the session's firewall settings and use the firewall specified for the current session.

**Note:** When specifying a firewall on the same command line used to open multiple sessions, SecureCRT will apply the firewall to each of the sessions.

Using SecureCRT's dependent session option on the command line or in the connect bar (as in the following example), it's possible to link a session to an SSH2 session that it depends on, which allows connection to a jump host before connecting to other sessions.

Both the command line and the connect bar use the normal /Firewall specifier and the firewall will be Session: <session name> with the following syntax.

/FIREWALL=Session:
<Session Name>
ssh2://[<user>]@<host>

		Session: is case insensitive.
/LOG	filename	Specifies a log file to be used for the session being started.
/MAX_COLS	n	Specifies the maximum number of columns. The maximum value for n is 1024 and the minimum value is 132.
/N	tabname	Specifies the name to display on the tab.
/NOMENU		Hides the menu bar.
/NOTOOLBAR		Hides the toolbar.
/NOSAVE		No changes to settings will be saved.
/P0S	x y	Specifies the initial position of the SecureCRT window, where <i>x</i> and <i>y</i> specify the upper left corner of the SecureCRT window in pixel coordinates.
/SCRIPT	filename	Runs script contained in filename.
/SESSION_FOLDER	folder	Opens the sessions in the specified folder that would be opened if the folder was designated as an auto session. To open all the sessions, use the following command:  /SESSION_FOLDER \ The /SESSION_FOLDER option can be used with the /S option.
/TITLEBAR	"Title	Sets the text in the title bar to the

t d.	o isplay" s	string given as argument.
------	----------------	---------------------------

# **Examples of Command Line Usage:**

# Example:

To launch a new Telnet session, displaying the name "Houston, TX" on the tab, use the following:

/T /N "Houston, TX" /TELNET 192.168.0.6

# Table of Protocol-Specific Command-Line **Options**

Any of the standard options may be used in combination with a protocolspecific option.

Protocol Option And Mandatory Arguments	Related Options and Arguments	Description
/S session_name		Starts SecureCRT and opens a connection with session_name. If session_name includes any spaces, it must be enclosed with quotation marks, (e.g., "My Session"). Multiple sessions can be opened by specifying multiple "/S session_name" argument pairs on the command line.  Note: The /S option is considered a protocol-specific option because the protocol to be used is defined by the session parameters.
/T	/S	When used in conjunction with the /S options, the /T option opens the specified session or sessions in a separate tab or tabs. For example, the command:  /T /S alpha opens session alpha in a new tab. The command:

		/T /S alpha /S beta opens both sessions alpha and beta in their own tabs.  Note: /T must precede /S in the command string.  The /T option can also be used with "ad hoc" sessions such as:  /T /SSH2 myserver	
/SSH1 [ssh_options] hostname	none	Starts SecureCRT and opens the default session with the SSH1 protocol and begins connecting to hostname. SecureCRT also accepts username@hostname syntax.  Note: /SSH1 and its related options are only available if SSH1 support was selected for installation during the installation process.	
	[/ACCEPTHOSTKEYS]	Instructs SecureCRT to automatically accept host keys.	
	[/ C cipher]	Specifies a cipher for encrypting the session. Valid values for SSH1 cipher are NONE, DES, 3DES, RC4 and BLOWFISH. The default SSH1 cipher is 3DES.	
	[/ENCRYPTEDPASSWORD password]	Specifies a password in encrypted form. This	

	password must have been encrypted by SecureCRT. To obtain an encrypted password, specify a password in the <b>Session Options</b> dialog, then open that session's .ini file and copy the encrypted string from the <b>Password</b> field. To use passwords in scripts, refer to the Session Object connect method.
[ / I identityfile ]	Specifies the location of the user's identity file. The identity file contains the private key needed to connect to the server using RSA authentication. The absence of this option causes password authentication to be used.
[/L username]	Specifies the username when connecting to the SSH1 server .
[/P port]	Specifies the SSH server port. The default value is 22.
[ /PASSPHRASE passphrase ]	Logs on to the SSH1 server using <i>passphrase</i> as the passphrase for the identity file given with the /L option.
[/PASSWORD password]	Logs on to the SSH1 server using <i>password</i> as

			the passwousername option.	ord for the given with the /L
		[/ Z compressionlevel]	level from compression	on = fastest) to compression = Setting this turns off
/SSH2 [ssh_options] hostname		one		Starts SecureCR default session vand begins conn SecureCRT also username@hostNote: /SSH2 and are only available was selected for the installation p
	[ ,	/ACCEPTHOSTKEYS]		Instructs Secure accept host keys
	[,	/AUTH method ]		Specifies the aut method(s) to be connecting to the multiple methods must be comma- spaces (i.e., /AU password, pub- interactive,
		/C cipher]		Specifies a ciphe session. Valid va are NONE, 3DES TWOFISH. The o is 3DES.

[/COMPRESSIONS type]	Specifies the cor the session. Valid compression typ and ZLIB@OPEI default type is No
[/ENCRYPTEDPASSWORD password]	Specifies a pass form. This passwer encrypted by Section of the specify a passwer options dialog, session's ini file encrypted string field.  To use password the Session Objection of the session of the
[/I identityfile]	Specifies the loc identity file. The the private key near the server using authentication. Toption causes parauthentication to
[/L username]	Specifies the use connecting to the
[ /L0CAL [localip:]localport:hostname:hostport]	Allows you to spool IP address when (not required), and connection requer (on localhost) to hostname.
[/M <u>mac</u> ]	Specifies the Me Code ( <i>MAC</i> ) to ι
[/P port]	Specifies the SS

			default value is 2
	[/PASSPHRASE passphras	/PASSPHRASE passphrase]	
	[/PASSWORD password]		Logs on to the S password as the username given
	[ /REMOTE [listenip:]listenpo host:destination-port]	REMOTE [listenip:]listenport:destination- pst:destination-port]	
	[/Z compressionlevel]		Specifies the cor 1 (lowest compre- (highest compre- Setting this optio compression.
/TELNET hostname [por	none	with the Telnet	efault session protocol and ecting to [port]
/RLOGIN hostname	none	with the RLog begins conne	fault session gin protocol and ecting to ecureCRT also
	[/L username]	Specifies the	username to

		use with the RLogin protocol.
/TAPI phone number	none	Starts SecureCRT and opens the default session with the TAPI protocol and begins connecting to the specified <i>phone number</i> .  Note: This command must be formatted as in the following example:  /TAPI "+1 (555) 555-5555"
	[/REDIAL attempts]	Specifies the number of redial attempts that SecureCRT will make before giving up on the connection.
/SERIAL port	none	Starts SecureCRT and opens the default session with <i>port</i> . If no <i>port</i> value is entered, SecureCRT will use the value of the <b>Port</b> option in the <b>Global Options</b> / <b>General / Default Session</b> Category. If no setting is entered for the default session, SecureCRT will use COM2 as the <i>port</i> entry. <b>Note:</b> /SERIAL and its related options are only available if Serial capability is selected during installation.
	[/BAUD baudrate]	Set the baud rate. Valid values for <i>baudrate</i> are 110, 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 38400,

	57600, 115200, 230400, 380400, 460800, and 921600. The default value is 38400.
[/CTS   /NOCTS]	Enable or Disable RTS/CTS (request-to-send/clear-to-send) hardware flow control. By default, RTS/CTS is enabled.
[/DATA databits]	Set the data bits. Valid values for n are 5, 6, 7 or 8. The default value is 8.
[/DSR   /NODSR]	Enable or Disable DTR/DSR (data-terminal- ready/data-set-ready). By default, DTR/DSR are not enabled.
[/PARITY parity]	Set the parity. Valid values for <i>parity</i> are NONE, ODD, EVEN, MARK, SPACE. The default is NONE.
[/ST0P stopbits]	Sets the stop bits. Valid values for <i>stopbits</i> are 0, 1 or 2 ( 0 indicates 1 stop bit, 1 indicates 1.5 stop bits, 2 indicates 2 stop bits). The default value is 0 (1 stop bit).
[/XON   /NOXON]	Enable or Disable XON/XOFF software flow control. By default, software flow control is disabled.

# **Examples of Command Line Usage:**

#### Example 1:

The following example opens a connection to the "My Company" session in the <a href="mailto:specified position">specified position</a>:

SecureCRT.exe /POS 100 50 /S "My Company"

#### Example 2:

The following example creates an SSH2 connection to the remote machine named mySSH2server.myISP.com. It uses the username myusername to begin the logon process. It also specifies the cipher to use as Twofish, and the MAC to use as MD5. In addition, port forwarding is set up with the remote machine imap.myserver.com, instructing the local machine to listen for IMAP requests on port 8143 (local machine):

SecureCRT.exe /SSH2 /L myusername /C twofish /M MD5 /LOCAL & Example 3:

The following example connects to mycompany.com using the SSH1 protocol specifying the username "bob", the location of an identity containing a private RSA key and the Blowfish cipher:

SecureCRT.exe /SSH1 /L bob /I "C:\ SecureCRT\RSAId Example 4:

The following example connects to mycompany.com using the SSH2 protocol, username "bob", level 4 compression, and sets up port forwarding of connections from port 555 on the localhost (the PC running SecureCRT) to port 6666 on mycompany.com:

SecureCRT /SSH2 /L bob /Z 4 /LOCAL 555:mycompany.com

# SFTP Tab Command Options

SFTP Tab command options are described below. Commands that specify a path can contain the wildcard characters \* and ?. SFTP will expand these wildcard characters. Multiple wildcard characters are permitted in a path (e.g., "\*.\*" or "\*.?"). The SFTP Tab also supports file "globbing". The sequence [...] will match any one of the characters enclosed. Within [...], a pair of characters separated by a quotation mark (") matches any character lexically between the two. The backslash can be used to turn off wildcard characters (e.g., \\* or \[).

SecureCRT also supports tab completion of command arguments on the SFTP Tab. Begin typing a command and argument and then press the TAB key. SecureCRT will attempt to complete the argument(s) based on the available choices. If there are multiple choices, SecureCRT will display a list of possible options. You can choose from this list or continue typing until SecureCRT can identify the unique argument.

To open an SFTP tab session, right-click on an existing, connected SSH2 tab and choose **Connect SFTP Session** from the displayed menu.

Option	<u>Arguments</u>	<u>Description</u>
ascii		Sets the file transfer mode to ASCII.
binary		Sets the file transfer mode to binary.
cd	[path]	Changes the remote directory to that specified byS the path. If a path is not specified, the user is placed in their "home" directory.
chgrp	group path	Changes group of file "path" to "group".
chmod	mode path	Changes permissions of file "path" to "mode".
chown	owner path	Changes owner of file "path" to

		"owner".		
detail	remotepath	Displays system information about the specified remote file of folder.		
exit		Exits the SFTP application.		
get	[-r] [-a -b] remotepath	Retrieves the remote path and stores it in the current local directory. The -r argument specifies a recursive get. The -a argument specifies that the files are transferred as ASCII; -b specifies binary. If neither -a nor -b is specified, the current transfer mode will be used (see type command).		
help		Displays usage.		
include	filename or < filename	Includes the commands in the specified file as though they had been typed in.		
lcd	[path]	Changes the local directory to that specified by the path. If a path is not specified, the local working directory is set to the local user's "home" directory. Adding the -a argument will show all files and folders (including those hidden), and the -d argument will show only directories.		
ldetail	localpath	Displays system information about the specified local file or folder.		
ln	existingpath	Creates symbolic link on remote		

	linkpath [- s]	file.	
lls	[-1] [-a -d] [path]	Displays the local directory listing of the path specified or of the current local working directory if the path is not specified. Adding the -a argument will show all files and folders (including those hidden), and the -d argument will show only directories.	
lmkdir	path	Creates local directory specified by the path.	
lmv	oldpath newpath	Moves/renames local file. Identical to the local Rename (Irename) command.	
lpwd		Prints local working directory.	
lrename	oldname newname	Renames the specified file. Identical to the local Move (lmv) command.	
lrm	path	Deletes local file.	
lrmdir	path	Removes local directory.	
ls		Displays the remote directory listing of either the path or of the current directory if the path is not specified. If the -I flag is set, SFTP displays permission and ownership information. Adding the -a argument will show all files and folders (including those hidden), and the -d argument will show only directories.  Note: This command, if used with a / (i.e., 1s /), lists the	

		roots when connecting to a server that allows multiple roots, such as VanDyke Software's VShell®.	
mkdir	path	Creates remote directory specified by the path.	
mv	oldpath newpath	Moves/renames remote file. Identical to the Rename command.	
pwd		Prints the remote working directory.	
put	[-r] [-a -b] localpath	Uploads the local path to the currently open directory on the remote machine. The -a argument specifies that the files are transferred as ASCII; -b specifies binary. If neither -a nor -b is specified, the current transfer mode will be used (see type comment).	
rename	oldname newname	Renames the specified file. Identical to the Move (mv) command.	
rm	[-r] path	Deletes the remote file specified by the path. Adding the -r argument will initiate a recursive remove operation.	
rmdir	path	Deletes the remote directory specified by the path.	
su	username	Substitutes the current user with the one specified. <b>Note:</b> This command can only be called if the remote machine	

		is running VShell for Windows 3.5 or later.
type	[transfer mode]	Displays or sets the file transfer mode. Issuing this command with no parameter will display the current mode; issuing it with a mode parameter (either ascii or binary) will set the mode to that parameter.
view	filename	Displays or, if applicable, runs the specified file or files. Displayed files can be modified but changes made will not be uploaded to the original file.



# Overview of Xmodem

Xmodem is a simple file transfer protocol that supports basic error detection methods to ensure that information is not lost or corrupted during a transfer. Generally, Xmodem uses a checksum to provide error detection, however, the SecureCRT implementation of Xmodem also supports the CRC option for error detection.

The SecureCRT Xmodem implementation supports both 128-byte and Xmodem-1K (1024-byte) packet sizes.

Xmodem does not support multiple-file ("batch") transfers. However, SecureCRT does support zmodem which is capable of multiple-file transfers.

#### Topics for this section include:

Changing the Default File Transfer Directories Downloading a File with Xmodem Uploading a File with Xmodem

# Changing the Default File Transfer Directories

Files downloaded from a remote host will be saved to the default download directory unless otherwise specified. When selecting files to be uploaded to a remote machine, SecureCRT initially displays the contents of the default upload directory.

To change the default download and upload directories:

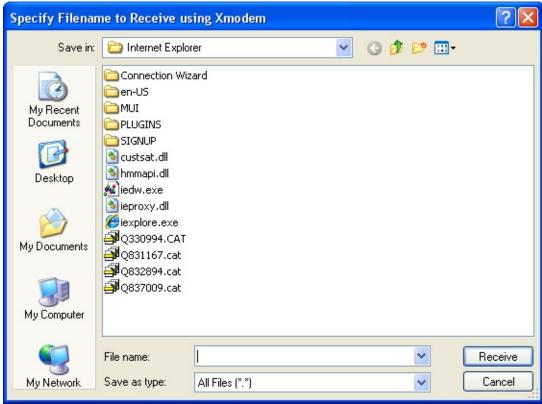
- 1. In the Session Options dialog, select the Terminal/ X/Y/Zmodem category.
- 2. In the **Directories** group, enter the full path of the new default upload or download directory <u>or</u> use the **Browse** button to open the **Browse for Folder** dialog and select the new directory.



# **Downloading a File with Xmodem**

To download a file from a remote machine, complete the following steps:

- On the remote machine, issue the appropriate Xmodem command to instruct the Xmodem software to send the file that you want to download. The remote machine software should then prompt you to initiate the transfer.
- 2. To initiate the Xmodem download, open the SecureCRT <u>Transfer</u> menu and select **Receive Xmodem**.
- 3. In the <u>specify Filename to Receive using Xmodem</u> dialog, provide a filename for the incoming file. The file will be saved to the <u>default download directory</u> unless otherwise specified.
- Click the Receive button to start the download. The Xmodem
   Transfer dialog will appear, displaying the status of the download.



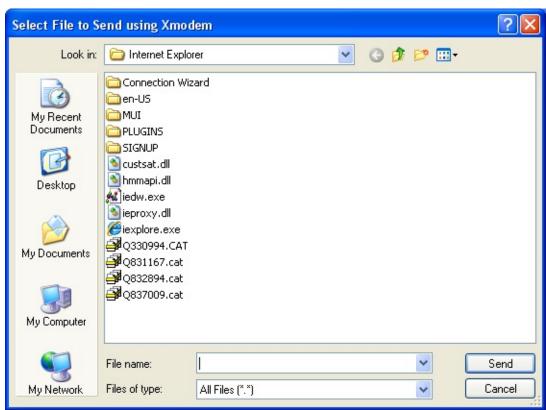
Specify Filename to Receive using Xmodem Dialog



# **Uploading a File with Xmodem**

To upload a file to a remote machine, complete the following steps:

- On the remote machine, issue the appropriate Xmodem command to instruct the Xmodem software to receive the file that you want to upload. The remote machine software should then prompt you to initiate the transfer.
- 2. To initiate the Xmodem upload, open the SecureCRT <u>Transfer</u> menu and select **Send Xmodem**.
- 3. In the **Select File to Send using Xmodem** dialog, select the file that you want to upload to the remote host. By default, the dialog opens to the <u>default upload folder</u>.
- 4. Click the **Send** button to start the upload. The **Xmodem Transfer** dialog will appear, displaying the status of the upload.



Select File to Send using Xmodem Dialog



# **Overview of Zmodem**

zmodem is a full-duplex file transfer protocol that supports fast data transfer rates and effective error detection. Zmodem is very user friendly, allowing either the sending or receiving party to initiate a file transfer. Zmodem supports multiple file ("batch") transfers, and allows the use of wildcards when specifying filenames. Zmodem also supports resuming most prior Zmodem file transfer attempts.

The <u>Zmodem protocol features can be disabled</u> to prevent undesired automatic uploads to the local machine from an unknown source.

#### Topics for this section include:

Changing the Default File Transfer Directories

Downloading a File with Zmodem

Uploading a File with Zmodem

**Disabling Zmodem** 

In its default configuration, SecureCRT initiates Zmodem file transfers by sending an "rz" command to the remote server. If you are trying to connect to a server that expects a different command (e.g., "lrz" in FreeBSD), you can change this command by editing the session's .ini file. In the .ini file, locate the following entry:

S:"Zmodem Receive Command"=rz\r

You can set the receive command to whatever the remote system expects and add command-line arguments if you want.



# Downloading a File with Zmodem

To download a file from a remote machine, complete the following steps:

- 1. On the remote machine, issue the appropriate **Zmodem** command to instruct the Zmodem software to send the file that you want to download.
- 2. A **Zmodem Transfer** dialog will appear, displaying the status of the download. The file will be saved to the default download directory.



# **Uploading a File with Zmodem**

To upload a file to a remote machine, complete the following steps:

 On the remote machine, issue the appropriate Zmodem command to instruct the <u>zmodem</u> software to receive the file that you want to upload. This will cause SecureCRT to open the **Select Files to Send using Zmodem** dialog (shown below). Note that Zmodem must be installed on the remote machine to transfer files with Zmodem.

**Note:** Another way to open this dialog is to open the <u>Transfer</u> menu and select **Zmodem Upload List**.

- In the Select Files to Send using Zmodem dialog, select the file(s) that you want to upload to the remote machine and click on the Add button to add the file(s) to the Files to send list. You can also choose to send your file(s) in ASCII instead of binary by checking the Upload files as ASCII check box.
- 3. When you have finished adding all the files that you want to upload, click on the **OK** button.



Select Files to Send using Zmodem Dialog



# Disabling Zmodem

To disable the **Zmodem** protocol features in SecureCRT:

- 1. In the <u>Session Options</u> dialog, open the <u>File Transfer/X/Y/Zmodem</u> category.
- 2. Select the **Disable Zmodem** option.

When the Zmodem protocol is disabled, SecureCRT will ignore the character sequence that starts a Zmodem transfer.



## Administrative Template

VanDyke Software has created an administrative template file that administrators can use to configure administrative options (e.g., FIPS Mode) either on a per-machine basis or a per-user basis. To use this file, you must request it from VanDyke Software and then configure your system to recognize it.

#### **Requesting an Administrator Template File**

To request a SecureCRT administrator template file, open your browser to the appropriate website shown below and follow the instructions there.

For Windows Vista/7 and newer:

https://secure.vandvke.com/cgi-bin/adm\_request.php?pid=SecureCRT&format=admx

For Windows XP/2003:

https://secure.vandyke.com/cgi-bin/adm\_request.php?pid=SecureCRT&format=adm

#### Instructions for using ADM file (Windows XP/2003)

- 1. As an administrator, start MMC.
- 2. From the **File** menu, select **Add/Remove Snap-in**.
- 3. In the Add/Remove Snap-in dialog, click on the Add button.
- 4. Add the stand-alone snap-in **Group Policy**.
- 5. For the **Group Policy Object**, choose **Local Computer** (the default). This can also be done for domains. Click on Finish, and close all the dialogs by clicking either Close or OK.
- 6. From the Console Root, expand Local Computer Policy and then expand User Configuration.
- 7. Right-click on **Administrative Templates** and select **Add/Remove** Templates.
- 8. Click on the **Add** button.
- 9. In the **Policy Templates** dialog, browse to the location where you stored the file VanDykePolicy.adm and select that file. Close the Add/Remove Templates dialog.
- 10. This will have added a folder called "VanDyke Software Settings" to

the **Administrative Templates** folder under the **Local Computer Policy/User Configuration** snap-in. From here, you can configure all the VanDyke administrator options.

# Instructions for using ADMX and ADML files (Windows Vista/7 and newer)

1. As an administrator, save the ADMX file in the correct location. In most cases, this location is:

C:\Windows\PolicyDefinitions\

2. As an administrator, save the ADML file in the correct location. In most cases for those in the US, this location is:

C:\Windows\PolicyDefinitions\en-US

Additional information can be found at:

http://msdn.microsoft.com/enus/library/bb530196.aspx

3. As an administrator, run the following snap-in:

gpedit.msc

4. When the **Group Policy** window appears, a folder named "VanDyke Software Settings" can be found in the **Administrative Templates** folder under **User Configuration**. From here, you can configure all the VanDyke administrator options.

# FIPS Mode

Federal, state, and local government agencies, critical infrastructure providers like public utilities, and private sector organizations that do business over the public network with these agencies must use encryption that meets the FIPS 140-2 standard.

If you need to protect data in transit as outlined by FIPS 140-2 or NIST 800-53, SecureCRT has an administrator option to run in "FIPS Mode". When this option is set, SecureCRT will use a FIPS 140-2 validated cryptographic library and only allow FIPS-approved algorithms.

VanDyke Software has partnered with RSA Security, Inc. to use the BSAFE Crypto-C Micro Edition cryptography module which has been tested by Atlan Laboratories, an accredited testing laboratory for FIPS compliance. This module has met all Level 1 requirements for FIPS 140-2 compliance when operated in FIPS mode.

The following FIPS-approved Cryptographic algorithms are used: DSA (Cert. #143); Triple-DES (Cert. #378); AES (Cert, #303); RSA (Cert. #96); SHA-1; Diffie-Hellman (used for key exchange in SSH2 is allowed in FIPS mode but not approved).

The following algorithms are not available in FIPS mode: MD5; Twofish; Blowfish; RC4.

#### **Enabling FIPS Mode**

To run SecureCRT in FIPS mode, you must first configure the Microsoft Management Console (MMC) to recognize the VanDyke Software administrative template (.adm) file; once this is done, you can enable FIPS mode by completing the follow steps:

- 1. In MMC, navigate to **VanDyke Software Settings** and select the **Enable FIPS Mode** setting.
- 2. Open the **Properties** dialog by selecting **Properties** from the **Action** menu.
- 3. On the **Setting** tab, select **Enabled**. Press the **OK** button to save the change.

FIPS mode will be enabled the next time SecureCRT is started.

### Sessions Overview

SecureCRT supports connections with a variety of protocols: SSH1, SSH2, Telnet, Telnet/SSL, RLogin, and Serial. In addition, SecureCRT supports modem dialing with the TAPI connection protocol. This section of the SecureCRT help manual contains information on how to set up a session to connect with a remote machine using these protocols.

SecureCRT allows you to personalize your settings and options and automatically use them every time you establish a connection to a remote computer This is accomplished by creating a session that will store your personalized settings under a session name.

**Note:** A session is a set of options and customized settings that are assigned to a connection to a remote machine. These settings and options are saved under a session name and allow you to have different preferences for different hosts.

SecureCRT provides two different methods for creating a session:

- The **Quick Connect** dialog
- The **connect** dialog

You may find it useful to read an Overview of Connection Protocols before creating or editing a session. If you have already created a session, you may want to learn about Changing Session Options.

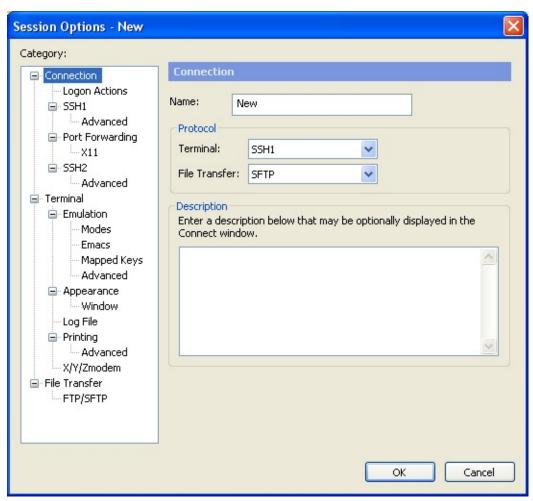


## 🗺 Creating a New Session with Connect

In contrast to the Quick Connect dialog (see Creating a New Session with Quick **Connect** ), the **New Session** button **Session** on the toolbar of the **Connect** dialog allows you to name the session something other than the hostname or IP address of the computer to which you are connecting.

Note: "Default" is not a valid name for a session that is at the root level in the **Connect** dialog.

When you use the **New Session** button, the session options dialog appears (shown below), allowing you to customize all of the settings associated with the new session. Enter the **Name** and **Protocol** of your new session and the **Connection** subcategories will change based on your chosen protocol. To complete your setup, click on the subcategory for the **Protocol** you chose and enter the required information. Your **Protocol** setting options are SSH1, SSH2, Telnet, Telnet/SSL, RLogin, Serial, and TAPI.



Session Options: Creating a New Session

When you have finished setting up the new session, you will see the new session listed in the **Connect** dialog.

Once you have created a session, you may want to customize its behavior. For more information, see the topics in the "Customizing Session Behavior" section.



### Overview of Connection Protocols

SecureCRT supports the following protocols: <u>ssh1</u>, <u>ssh2</u>, <u>Telnet/ssl</u>, <u>Telnet</u>, RLogin, Serial (COM), TAPI, and Raw.

If you are using SecureCRT to connect to a BBS or any other type of terminal server by directly dialing the remote machine using your computer's modem, you will need to use either the Serial or TAPI protocol.

If you are using SecureCRT to connect to a remote machine via the Internet or your LAN, you should use the SSH1, SSH2, Telnet, Telnet/SSL, or RLogin protocol depending on the services provided by the remote machine to which you are connecting.

If you are using SecureCRT to establish a secure connection using encryption, you will need to use either the SSH1 or SSH2 protocol.

When the Raw protocol is used, little or no interpretation of the remote data takes place, which can be helpful for troubleshooting.

This following topics provide an explanation of the settings related to each of the protocols supported in SecureCRT:

**Telnet Connection Settings** 

**RLogin Connection Settings** 

**Serial Connection Settings** 

**TAPI Dialer Settings** 

**SSH1 Connection Settings** 

SSH2 Connection Settings

Telnet/SSL Settings

**FTP Connection Settings** 

**Raw Connection Settings** 

Note: Only the protocols selected during the installation process will be available.



## **Changing Session Options**

Changing <u>session</u> options is done with the <u>session Options</u> dialog. The **Session Options** dialog is accessed in a variety of ways.

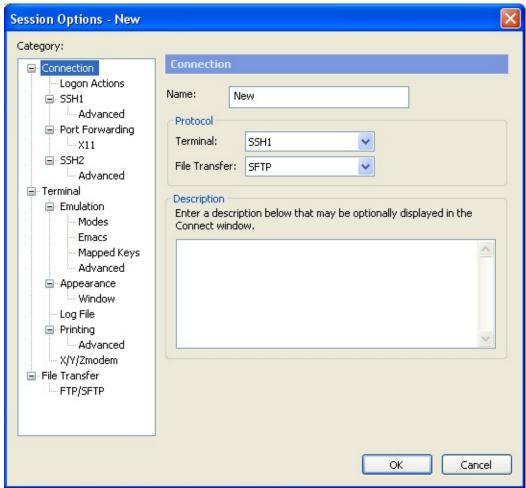
# If you are not currently connected with a session to a remote machine:

- 1. Open the **Connect** dialog by clicking on the **Connect** button ., or by opening the **File** menu and selecting the **Connect** menu item.
- 2. Select the session listed in the **Connect** dialog that you would like to edit.
- 3. Open the **Session Options** dialog by either clicking on the **Properties** button , or by right-clicking on the target session and selecting **Properties** from the pop-up menu. You can also open the **Session Options** dialog by selecting the target session and using the ALT+ENTER hot-key sequence.

# If you are connected with a session to a remote machine and would like to edit the current open session:

Open the <u>Session Options</u> dialog by either clicking on the **Properties** button , or by opening the <u>Options</u> menu and selecting the <u>Session</u> **Options...** menu item.

The **Session Options** dialog (*Session Options Dialog* below) is divided into two major sections: an options category tree view, and an options category panel. The options category tree view allows you to select the category of options you wish to change. The options category panel displays all the options associated with the category selected in the options category tree view. For example, in the figure below, the <u>connection</u> category is selected, and all of the options associated with the **Connection** category are displayed in the options category panel.



Session Options Dialog



## Security Considerations

Session security depends on several factors, including whether the connection you are using to the host is a trusted connection. If it is not, consider whether private or confidential information will be sent and received. A Telnet session will transmit user ID, password, and other sensitive or private information in an easily readable format.

For maximum security, DO NOT put passwords in script files or run them as mapped keys or as button functions. This information is stored in the SecureCRT configuration file and may be accessible to other users that have access to the computer on which SecureCRT is installed.

Maximum security and privacy on the Internet and local networks requires the use of the Secure Shell Protocols (SSH1 or SSH2) or the Secure Sockets Layer (SSL) supported in SecureCRT.

Note that although SecureCRT does support the Telnet protocol, SecureCRT Telnet sessions that are not Telnet/SSL are not encrypted. Encrypted connections are achieved through the SSH1, SSH2, and Telnet/SSL protocols.

## Port Forwarding with SSH

#### **Overview of Port Forwarding**

Port forwarding is a powerful tool that allows you to secure TCP/IP traffic using SecureCRT's <u>SSH1</u> and <u>SSH2</u> protocol support. This means that you can encrypt application data using protocols such as IMAP, POP3, and SMTP. For example, if you receive your email from an Internet Service Provider (ISP), you could encrypt the communication between your workstation running the email client and the ISP's SSH server. SecureCRT also supports X11 forwarding, which allows X Windows traffic between the X server and X client to be encrypted.

In general, with any port forwarded by SecureCRT for an application, the application needs to be reconfigured to use the localhost or loopback address 127.0.0.1 as its application server address.

#### **Setting up Port Forwarding**

To set up port forwarding, follow these steps:

- 1. Click on File / Connect and select the SSH session for which you would like to use forwarded ports.
- 2. Click on the **Properties** button or right-click on the session and select **Properties** from the pop-up menu to bring up the session Options dialog.
- 3. Under the Connection category, click on the Port Forwarding subcategory.
- 4. To add a new locally forwarded port, click on the **Add** button and fill in the local port, remote hostname and remote port. Click on the OK button to save your settings.
- 5. To add a new remotely forwarded port for an SSH2 connection, click on the Remote/X11 subcategory and then click on the Add button. Fill in the local port, remote hostname and remote port, then click **OK**.

**Note:** As you enter the local port, the remote port is automatically filled in from the existing session information. Ports may be defined either by their port number or by their service name.

6. Connect with this session to start port forwarding, then run the client

application.

Port forwarding works by forwarding data from a local port to the remote host/port. For example, to secure POP3 traffic through your mail client, set up port forwarding with the following settings:

- · local port=110
- · remote hostname: set to the mail server's hostname
- · remote port=110.

Configure your mail client to use 127.0.0.1 (otherwise known as "localhost") as the POP3 server's IP address. Hostname and port configuration needs to be done in both SecureCRT and the client application (e.g., email). After connecting with this session, POP3 traffic is encrypted to the SSH server as long as SecureCRT is running. If the connection to the SSH server is broken or closed, the forwarded ports will no longer be forwarded, and the client applications may receive an error when they try to connect to the local port.

#### **Forwarding X11 Packets**

X11 Forwarding is configured as part of the port forwarding setup. To enable X11 packet forwarding, follow the steps outlined in the "Setting up Port Forwarding" section (above) with the added step of selecting the **Forward X11 packets** option on the **Remote/X11** category.

**Note:** SecureCRT is <u>not</u> an X Server. The **Forward X11 packets** option allows SecureCRT to accept X11 data from the remote machine and forwards it to the X server running on the local machine. The local X Server must be running before any X11 sessions can be displayed. If you are using Xhost authority access on the local X11 server, you will need to add the localhost or loopback address 127.0.0.1 to your server's Xhost list.

#### **Security Considerations with Port Forwarding**

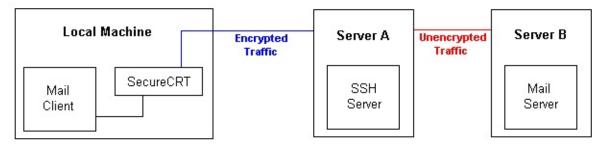
It is important to understand that the client data is only encrypted between the machine that SecureCRT is running on and the SSH server that SecureCRT is connected to. Any data moving from the SSH server across the network to another server is not encrypted.

Two configurations are presented below to illustrate different machine/network configurations and their affect on security. Your

evaluation of the connection between Servers A and B is the critical factor in deciding whether the aggregate security meets your needs.

#### **Configuration 1 (Less secure)**

SecureCRT forwards POP3 mail to a remote mail server that is a different machine than the SSH server.



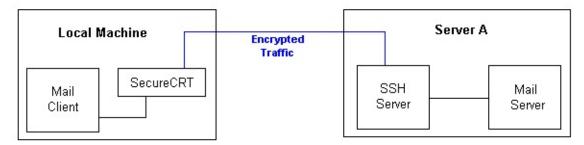
- Between the Local Machine and Server A the data is encrypted.
- Between Server A and Server B the data is not encrypted.
- Since the SSH server and mail server are on different machines your data can be viewed on this connection.

In Configuration 1, the connection between Servers A and B could be one of the following:

- · On the Internet an unsecure network.
- On an internal LAN a network that may or may not deliver a satisfactory level of security.

#### **Configuration 2 (More secure)**

SecureCRT forwards POP3 mail to a remote mail server that is running on the same machine as the SSH server.



- Between the Local Machine and Server A the data is encrypted.
- Since there is no network traffic between the SSH server and Mail server, security is increased over Configuration 1.

## Configuring Port-Forwarding Filters

SecureCRT port-forwarding filters let you allow or deny connections from specific IP address or range of subnets. The filter configuration is stored in the session initialization file (<session name>.ini) in the SecureCRT Config/Sessions folder.

Port-forwarding filters are defined in a list and applied from the first to last. When an application makes a port-forward connection via SecureCRT, the IP address of the application's machine is checked against the filter list. SecureCRT compares the address with the first filter entry and then goes through the list to the last filter entry. When an address meets the criteria of a filter entry, that filter is applied and the rest of the list is ignored. An empty list will deny all port-forwarding connections.

The filter syntax is as follows:

```
"Port Forward Filter"=[access,host] [access,host]
[...]
```

In the above syntax, the argument values are as follows:

```
access - allow or deny
```

host - IP address or a range of subnets

The default settings for the port-forward filter is as follows:

```
S: "Port Forward Filter"=allow, 127.0.0.1,0
deny, 0.0.0.0/0.0.0.0,0
```

This setting allows connections from localhost, but denies connections from any other machine.

#### **Reverse Port Forwarding**

SecureCRT also supports reverse port forwarding.

The filter syntax is as follows:

```
"Reverse Forward Filter"=[access, host] [access, host]
[...]
```

In the above syntax, the argument values are as follows:

```
access - allow or deny
```

host - IP address or a range of subnets

The default settings for the port-forward filter is as follows:

```
S:"Reverse Forward Filter"=allow, 127.0.0.1, 0 deny, 0.0.0.0/0.0.0.0, 0
```

This setting allows connections from localhost, but denies connections from any other machine.

#### **Example Filters**

The following is an example of a port-forward filter list that allows connections from the specified machines (Alice's cable modem at 129.24.32.16 and Bob's DSL subnet at 198.129.241.85/255.255.255.128,0) but denies access from other machines:

```
S:"Port Forward Filter"=allow, 129.24.32.16, 0 allow, 198.129.241.85/255.255.255.128, 0 deny, 0.0.0.0/0.0.0.0, 0
```

This example list will allow all connections:

```
S:"Port Forward Filter"=allow, 0.0.0.0/0.0.0.0, 0 Or, for IPv6:
```

```
S:"Port Forward Filter"=allow,::/0,0
```

And this example list will allow everything in a local reserved class C network and any localhost loopback adapter (e.g., 127.0.0.2, 127.0.0.3, etc.), and deny everything else:

```
S:"Port Forward
Filter"=allow, 192.168.0.0/255.255.255.0,0
allow, 127.0.0.0/255.255.255.0,0
deny, 0.0.0.0/0.0.0,0
```

An example of a more common filter would be the following, which allows local host on 127.0.0.1,0, allows any host on local 192.168.0.0 network, and denies everything else:

```
S:"Port Forward Filter"=allow, 127.0.0.1, 0 allow, 192.168.0.0/255.255.255.0, 0 deny, 0.0.0.0/0.0.0.0, 0
```

The following is an example of a port-forward filter list that allows specific

#### IPv6 connections:

```
S:"Port Forward
Filter"=allow,::ffff:198.129.241.85/128,0
```

For reverse port forwarding, the syntax is similar. For example, this example list will allow everything in a local reserved class C network and any localhost loopback adapter (e.g., 127.0.0.2, 127.0.0.3, etc.), and deny everything else:

```
S:"Reverse Forward
Filter"=allow, 192.168.0.0/255.255.255.0,0
allow, 127.0.0.0/255.255.255.0,0
deny, 0.0.0.0/0.0.0,0
```



## Public-Key Authentication for SSH1

#### Overview of RSA Authentication

RSA authentication uses a public-private key pair to authenticate and log onto an SSH1 Server. It offers a higher level of authentication security than password authentication by requiring both the private key and the passphrase that protects the private key in order to complete authentication.

Setting up RSA public-key authentication for a SecureCRT SSH1 Session is a multi-step process. Identity files are first created with the SecureCRT Key Generation wizard. The identity file is defined for global or session-specific use in the SSH1 category of the Global Options dialog or in the Connection/SSH1 category of the Session Options dialog. Then the public key is added to the authorized keys file, which is located on the SSH1 server.

#### **Creating Identity Files**

To create an RSA identity file:

- 1. In the Connect dialog, select the SSH1 session with which you would like to use the identity files.
- 2. Open the <u>session Options</u> dialog and click on the <u>connection/SSH1</u> category.
- 3. In the **Authentication** field, choose RSA as the authentication type and click on the **Properties...** button.
- 3. In the Public Key Properties dialog, select whether you want the new identity file to be global or session specific and click on the **Create Identity File** button.
- 4. Follow the instructions in the Key Generation wizard to create your identity files. Once your public-private key pair has been generated by the Key Generation wizard, you will be prompted for the path and filename in which your private key will be stored. Be sure to specify a secure location for this file such that you are the only individual with access to it. The public key will be placed in a file with the same base name as the private key file, but with an extension of .pub.

#### **Using Your Identity Files**

Once you have created your identity files, there are several steps that will

need to be completed so that you can make use of them with SecureCRT. The necessary steps are 1) Transfer the public-key file (e.g., Identity.pub) to the remote machine; 2) Put the contents of the public-key file in your authorized\_keys file on the remote machine; and 3) Configure SecureCRT to use the identity file with RSA authentication.

#### 1. Transfer the identity file to the SSH server

In order to use your public key you must transfer the public-key file (with the extension .pub) created by the Key Generation wizard to the ~/.ssh/ directory on the SSH server. One of the easiest ways to transfer this file to the SSH server is to use the zmodem file transfer feature provided by SecureCRT. If the remote machine does not have zmodem installed, you will need to use an FTP client to transfer the identity file to the remote machine. Another way to transfer the identity file to the remote machine is to copy and paste the contents of the identity file. To accomplish this, use Notepad.exe to open the identity file and copy the contents of the identity file to the clipboard. On the remote machine, create a file using your favorite editor and paste the contents of the clipboard using the paste button.

# 2. Update the authorized\_keys file on the remote machine

Once you have transferred the Identity.pub file to the remote machine, you must update the authorized\_keys file on the remote machine. The following steps outline this update process.

- Connect to the remote SSH server using SSH1 and password authentication.
- 2. Append the contents of the public-key file created with the Key Generation wizard to the file ~/.ssh/authorized\_keys on the remote host. The default name of the public-key file is Identity.pub. Create the ~/.ssh/authorized\_keys file if it does not already exist. If you want multiple authorized keys, append the contents of the public-key file to the authorized\_keys file. The simplest way to do this is typically:

% cat identity.pub >> ~/.ssh/authorized\_keys (note that the name of the identity file can be different than the example)

**Note on placing public keys:** The format of the authorized\_keys file requires that each entry consist of a single long line. If you use Copy and Paste to add a public key to the ~/.ssh/authorized\_keys file, make sure that the entry contains no additional new-line characters.

#### 3. Configure SecureCRT to use your identity files

The identity file created by the Key Generation wizard contains both your new public key and your new private key. To configure SecureCRT to use the identity file:

- 1. In the **Connect** dialog, select the SSH session with which you would like to use the identity file.
- 2. Open the <u>session Options</u> dialog and in the **Connection/SSH1** category, change the **Authentication** setting from Password to RSA.
- 3. If you have more than one identity file, you may need to click on the **Properties...** button and verify that the session is using the session-specific key you have created.
- 4. Click on the **OK** button to save the changes. If you supplied a passphrase when you created your key, you will be prompted to enter it during the connection process.



## Public-Key Authentication for SSH2

#### **Overview of Public-Key Authentication**

Public-key authentication USes a public-private key pair to log onto an SSH2 server . Setting up public-key authentication for an SSH2 SecureCRT session is a multi-step process. Identity files must be created using the Key Generation wizard. One of the identity files created by the Key Generation wizard will contain a private key that will be assigned either on a global level for all SSH2 sessions or on a session-specific level. The global or session-specific characteristic of the private key is specified in the SSH2 category. The other identity file created by the Key Generation wizard will contain the corresponding public key and will need to be transferred to the proper location on the SSH2 server.

**Note:** Public keys generated using VanDyke Software products comply with the established IETF draft specification defining the format of Secure Shell public key files. This does not guarantee that SecureCRT will work with public key files generated using other Secure Shell software implementations which may or may not comply with this specification.

Since there is no IETF specification defining the format of Secure Shell private key files, SecureCRT may not be able to use private key files generated with other implementations. It should also be noted that, since the private key generated by SecureCRT uses a different format from OpenSSH's private key, OpenSSH cannot use a VanDyke Software generated private key.

SecureCRT supports SSH2 public-private key files generated with VanDyke Software products and the public-private key files generated with the OpenSSH ssh-keygen utility.

#### **Creating Global Identity Files**

- 1. Open the Global Options dialog and click on the SSH2 category
- 2. Click on the **Create Identity File** button.
- 3. Follow the instructions in the Key Generation wizard to create your identity files. The Key Generation wizard will ask for a passphrase, but the passphrase is not required. If the public key is going to be used as

part of an automated process, you may not want to use a passphrase.

4. Once your public-private key pair has been generated by the Key Generation wizard, you will be prompted for the path and filename in which your identity files will be stored. Be sure to specify a secure location for these files such that you are the only individual with access to them. The public key will be placed in a file with the same name as the private key file, but with an extension of .pub.

Note: SecureCRT supports both DSA and RSA key types.

#### **Creating Session-Specific Identity Files**

- 1. In the **Connect** dialog, select the SSH2 session with which you would like to use the identity files.
- 2. Open the session Options dialog
- 3. In the **Authentication** group, set one of your authentication methods to be PublicKey and click on the associated **Properties** button.
- 4. In the Public Key Properties dialog, click on the Create Identity File button.
- 5. Follow the instructions in the Key Generation wizard to create your identity files. The Key Generation wizard will ask for a passphrase, but the passphrase is not required. If the public key is going to be used as part of an automated process, you may not want to use a passphrase.
- 6. Once your public-private key pair has been generated by the Key Generation wizard, you will be prompted for the path and filename in which your identity files will be stored. Be sure to specify a secure location for these files such that you are the only individual with access to them. The public key will be placed in a file with the same name as the private key file, but with an extension of .pub.

#### **Using Your Identity Files**

Once you have created your identity files, there are several steps that will need to be completed so that you can make use of them with SecureCRT. The necessary steps are:

- 1. Configure the SSH2 server to recognize your public-key file (e.g., Identity.pub). Instructions are provided for configuring <u>VanDyke Software's VShell® server</u>, <u>OpenSSH</u>, <u>SSH Communications</u>, and <u>Data Fellows</u> SerVers.
- 2. Configure SecureCRT to use the identity file with public-key

authentication on the local machine. The identity file created by the Key Generation wizard contains both your new public key and your new private key. To configure SecureCRT to use the identity file complete the following instructions:

- a. In the **Connect** dialog, select the SSH2 session with which you would like to use the identity file.
- b. Open the <u>session Options</u> dialog and in the <u>connection/ssh2</u> category, change the **Authentication** setting from Password to PublicKey.
- c. If you have more than one identity file, you may need to click on the **Properties** button and verify that the session is using the session-specific key you have created.
- d. Click on the **OK** button to save the changes. If you supplied a passphrase when you created your key, you will be prompted to enter it during the connection process.

# Configure OpenSSH to Recognize Your Public-Key File

In order to use your public key you must transfer the public-key file (typically Identity.pub) created by the Key Generation wizard to the ~/.ssh directory on the <code>ssh2 server</code>. It is recommended that you follow the procedure below for using <code>copy and paste</code> to create a copy of the Identity.pub file in the ~/.ssh directory on the remote machine. If you decide instead to transfer the Identity.pub file to the ~/.ssh directory using an FTP <code>client</code>, be sure to transfer the file in ASCII mode. Also be sure to complete step 4d below.

To use copy and paste to configure the SSH2 server to recognize your public-key file:

- 1. Log on to the remote SSH2 server using SSH2 and password authentication .
- 2. On the local machine, use Notepad.exe to open the Identity.pub file that was created with the Key Generation wizard.
- 3. With the Identity.pub file opened in the Notepad application, open the **Edit** menu and choose **Select All**. Once everything is selected, open the **Edit** menu again and select **Copy**.
- 4. On the remote machine, complete the following steps:
  - a. % cat >  $\sim$ /.ssh/identity.pub
  - b. Click on the SecureCRT paste button to paste the contents of the Clipboard (which should now contain the contents of your Identity.pub file).
  - c. Issue a CTRL+D to close the Identity.pub file.
  - d. Convert the key to one that OpenSSH will recognize using the following command:

% ssh-keygen -i -f ~/.ssh/identity.pub >> ~/.ssh/authorized\_keys If you are connecting to a version of OpenSSH that is older than 2.9, use the following command instead:

```
% ssh-keygen -X -f ~/.ssh/identity.pub >> ~/.ssh/authorized_keys2
```

**Note:** If the key is an OpenSSH-format key, such as a converted key or a key exported from a certificate, the key does not need to be converted as shown above. Instead, use the following command:

```
% cat ~/.ssh/identity.pub >>
~/.ssh/authorized_keys
```

If you are connecting to a version of OpenSSH that is older than 2.9, use the following command:

```
% cat ~/.ssh/identity.pub >>
~/.ssh/authorized_keys2
```

e. Ensure that you are the only user with permissions to write to your "authorized\_keys" file. This can be done using the following command:

% chmod 600 ~/.ssh/authorized keys

The method described above uses only a single public key in the authorization file. It is possible to have more than one public key in the "authorized\_keys" file. To add additional public keys, repeat steps 2 through 4.

Return to Public-Key Authentication for SSH2

# Configure an SSH Communications Server to Recognize Your Public-Key File

In order to use your public key, you must transfer the public <code>identity file</code> (generally Identity.pub) created by the Key Generation wizard to the  $\sim$ /.ssh2 folder on the <code>ssh2 server</code>. It is recommended that you follow the procedure below for using <code>copy and paste</code> to create a copy of the Identity.pub file in the  $\sim$ /.ssh2 folder on the remote machine. If you decide instead to transfer the Identity.pub file to the  $\sim$ /.ssh2 folder using an FTP <code>client</code>, be sure to transfer the file in ASCII mode. Also be sure to complete step 4-d below.

To use copy and paste to configure the SSH2 server to recognize your Identity.pub file:

- 1. Log on to the remote SSH2 server using SSH2 and password authentication .
- 2. On the local machine, use Notepad.exe to open the Identity.pub file that was created with the Key Generation wizard.
- 3. With the Identity.pub file opened in the Notepad application, open the **Edit** menu and choose **Select All**. Once everything is selected, open the **Edit** menu again and select **Copy**.
- 4. On the remote machine, complete the following steps:
  - a. % cat >  $\sim$ /.ssh2/identity.pub
  - b. Click on the SecureCRT paste button to paste the contents of the Clipboard (which should now contain the contents of your Identity.pub file).
  - c. Issue a CTRL+D to close the Identity.pub file. Once you have typed CTRL+D, the Identity.pub file will have been created in the ~l.ssh2 folder on the remote machine.
  - d. % echo "Key identity.pub" >> ~/.ssh2/authorization

Return to Public-Key Authentication for SSH2

# Configuring a VanDyke Software VShell® Server to Recognize Your Public-Key File

# Using SecureCRT-Generated Public Keys with VShell for Windows

In order to use your public key, you must transfer the Identity.pub (or whatever name you chose) file created by the Key Generation wizard to the individual user's folder under the Publickey folder on the VShell  $_{\underline{\text{server}}}$ . For example:

C:\Program
files\VShell\Publickey\%User%\Identity.pub

If you are using VShell version 2.1 or newer, you can use the SecureCRT <u>public-key assistant</u> to upload your public key for you without knowing the exact location of the server's public-key folder.

If you are not using VShell version 2.1 or newer, it is recommended that you follow the procedure below for using copy-and-paste operations to create a copy of the Identity.pub file in the Publickey folder on the remote machine. If you decide instead to transfer the Identity.pub using an FTP client, be sure to transfer the file in ASCII mode.

To use copy-and-paste operations to configure the VShell server to recognize your Identity.pub file:

- 1. Log on to the remote VShell server using <u>SSH2</u> and password <u>authentication</u>.
- 2. On the local machine, use Notepad.exe to open the Identity.pub file that was created with the SecureCRT Key Generation wizard.
- 3. With the Identity.pub file open in Notepad, open the **Edit** menu and choose **Select All**. Once everything is selected, open the **Edit** menu again and select **Copy**.
- 4. On the remote machine, complete the following steps:
  - a. Change to the individual user's folder under the Publickey folder. For example:

\Program files\VShell\Publickey\%User%

b. Type copy con identity.pub on the command line and press the

ENTER key.

- c. Click on the SecureCRT **Paste** button to paste the contents of the Clipboard (which should now contain the contents of your Identity.pub file).
- d. Press CTRL+Z ENTER to close the Identity.pub file. Once you have typed CTRL+Z ENTER, the Identity.pub file will have been created in the folder on the remote VShell server.

# Using SecureCRT-Generated Public Keys with VShell for UNIX

In order to use your public key, you must transfer the Identity.pub (or whatever name you chose) file created by the Key Generation wizard to the individual user's .vshell/publickey directory. For example:

/home/someuser/.vshell/publickey/Identity.pub
or

~someuser/.vshell/publickey/Identity.pub

You can use the SecureCRT public-key assistant to upload your public key for you without having to copy the identity file by-hand. If you are not able to use the public-key assistant, it is recommended that you follow the procedure below for using copy-and-paste operations to create a copy of the Identity.pub file in the Publickey folder on the remote machine. If you decide instead to transfer the Identity.pub using an FTP or SFTP client, be sure to transfer the file in ASCII mode if applicable.

To use copy-and-paste operations to configure the VShell server to recognize your Identity.pub file:

- 1. Log on to the remote VShell server using SSH2 and password authentication.
- 2. On the local machine, use Notepad.exe to open the Identity.pub file that was created with the SecureCRT Key Generation wizard.
- 3. With the Identity.pub file open in Notepad, open the **Edit** menu and choose **Select All**. Once everything is selected, open the **Edit** menu again and select **Copy**.
- 4. On the remote machine, complete the following steps:

a. Issue a cd command to change the current working directory to the individual user's .vshell/publickey folder. For example:

cd ~someuser/.vshell/publickey

- b. Type cat >
  - ~someuser/.vshell/publickey/identity.pub on the command line and press the ENTER key.
- c. Click on the SecureCRT **Paste** button to paste the contents of the Clipboard (which should now contain the contents of your Identity.pub file).
- d. Press CTRL+D to close the Identity.pub file. Once you have typed CTRL+D, the Identity.pub file will have been created in the user's .vshell/publickey directory on the remote VShell server.

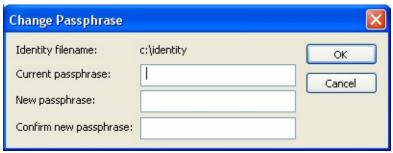
Return to Public-Key Authentication for SSH2



### Changing the Passphrase for a Private Key

If you would like to change the passphrase for a private key, follow the steps outlined below.

- 1. In the **Connect** dialog, select the <u>SSH</u> <u>session</u> that uses the private key for which you want to change the passphrase.
- 2. Open the <u>session Options</u> dialog by selecting a session and either clicking on the **Properties** button , or by right-clicking on the selected session and selecting the **Properties** menu item from the pop-up menu.
- 3. For SSH1 sessions: click on the <u>connections/SSH1</u> category, select RSA as the <u>Authentication</u> method and then click on the associated **Properties** button.
- 4. For SSH2 sessions: in the <u>connections/SSH2</u> category, select PublicKey in the **Authentication** window and then click on the associated **Properties** button.
- 5. Click on the **Change Passphrase** button in the <u>Public Key Properties</u> dialog.
- 6. In the **Change Passphrase** dialog (shown below), enter your current passphrase and the new passphrase (twice), and click on the **OK** button.



Change Passphrase Dialog

# Key Terminology

Most encryption technologies are based around a system of "keys" that allow information to be encoded and then decoded. The terms "private key", "public key", and "host key" are used extensively in <u>SSH</u> communication and can be confusing to those who are new to the terminology. This topic hopes to define and clarify these terms and their use.

#### **Private Key**

A private key is one of two keys used in public-key encryption (the other being a public key). The user keeps the private key secret and uses it to encrypt outgoing messages and decrypt incoming messages.

The permissions for the private key should be set so that only the owner has read/write access. This is especially important if the key does not have a passphrase.

#### **Public Key**

A public key is one of two keys used in public-key encryption (the other being a private key). The user releases a copy of this key to the public to allow anyone to use it for encrypting messages to be sent to the user and for decrypting messages received from the user.

#### **Public-Key Encryption**

A scheme using an asymmetric algorithm to create a pair of keys for encryption: a public key encrypts data, and a corresponding private key decrypts it. In some situations, such as digital signatures, the process is reversed: the sender uses a private key to create a unique number that can be read by anyone who has the corresponding public key. The agreement of the public and private keys verifies that the message is truly from the sender.

#### **Public-Key Authentication**

In public-key  $\underline{\text{authentication}}$ , public-private key pairs are used to identify a user to an  $\underline{\text{SSH2 server}}$ . A user creates both a public and private key, and then transfers a copy of the public key to an SSH2 server to which the user wants secure access. The private key is kept on the user's local

machine and is used to verify the identity of the user when the user attempts to connect to the SSH2 server. The public and private keys must be correct for the server to allow the connection .

#### **Host Key**

A host key is the public key in a public-private key pair that is used to identify a server host to a client in SSH2 connections. When a client connects to a server host, the server sends a host key to the client (the server keeps the private key secret). The first time the client connects to a server, the client's user is asked if they want to save the host key. If the user chooses to save the host key, the client adds the key to its host key database. Each time the client connects to that server, the client expects to receive the same key. If the server sends a different host key, the client is alerted to the fact that there may be a problem, which could be anything from a corrupt key file to a fraudulent server. The client then takes whatever action that is required to accept or reject the connection. Also see "Accepting a Host Key" and Global Options/SSH2 Host Keys.



## **Using X.509 Certificates**

#### **Overview**

X.509 is a proposed standard used for generating digitally-signed public-key <u>authentication</u> certificates. These certificates can be used for authentication in supporting <u>SSH</u> systems. Not all SSH servers support X.509 certificates.

Before you can use X.509 certificates, you must configure your system to support these certificates. In the example system shown below, a  $\frac{1}{2}$  client machine is connected to an SSH  $\frac{1}{2}$  which in turn is able to query a Certification Authority (CA) server.



X.509 System Configuration

#### **Obtaining an X.509 Certificate**

To obtain an X.509 certificate, begin by generating a public/private key pair on your client machine (refer to your system documentation for information on this process). Keep your private key on the local machine and forward the public key with any other required information to the CA in the form of a request for certification. If the request is approved, the CA digitally signs the certificate and returns it to you in the form of a smart card, token, or other medium.

SecureCRT supports X.509 systems that work with Windows compatible smart card readers.

#### **Configuring SecureCRT to Use X.509 Certificates**

Once you have your X.509 certificate, you need to configure SecureCRT to use it.

- 1. On the client machine, start SecureCRT.
- 2. Select the <u>session</u> that you want to use with the X.509 certificate and open the <u>session Options</u> dialog.

- 3. In the <a href="connection/ssh2">connection/ssh2</a> category, select Public Key as one of your <a href="Authentication">Authentication</a> types and click on the associated **Properties** button.
- 4. In the <u>Public Key Properties</u> dialog, clear the **Use global public key setting** option and check the **Use certificate** option.
- 5. Click on the **OK** button to save your settings.

**Note:** SecureCRT also supports the use of X.509 certificates for public keys on a global basis. This option is available in the <u>ssh2</u> category of the <u>Global Options</u> dialog.

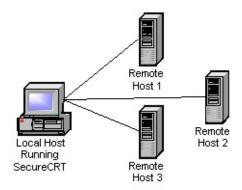


## **Using the Agent**

#### **Overview**

Agents are programs that work in the background gathering information or performing small processing tasks. In SecureCRT, the implemented agent temporarily holds private keys for use with <u>public-key authentication</u> to multiple remote hosts.

Public-key <u>authentication</u> uses a *public-private key pair* to log on to a Secure Shell <u>server</u>. After you <u>create your public-private key pair</u>, you store your private key on your local machine and transfer your public key to the remote host to which you want to connect. If you want to connect to more than one remote machine, you must transfer your public key to each machine. Many users choose to encrypt their private key with a <u>passphrase</u>. To log on to all the machines in the example below, you would have to enter your passphrase three times; once each time you made a <u>connection</u>, even if the machines use the same <u>public-private key pair</u>.

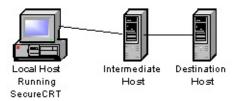


Connecting to Multiple Hosts

If you use the SecureCRT agent, however, you only have to enter your passphrase when making the first connection. The agent holds your decrypted private key and authenticates any further connections to machines using the same public-private key pair.

#### **Agent Forwarding**

Agent forwarding is using the agent to connect to a remote machine through another remote machine (see the figure below).



#### Connecting Through a Remote Host

To connect to the destination machine without using the agent, you would have to transfer your public key to both the intermediate and destination hosts <u>and</u> you would have to store your private key on the intermediate machine as well as on your local machine.

With the agent enabled, it acts as your proxy in authenticating to the destination host and allows you to keep your private key on just the local machine.

**Note:** Agent forwarding will only work if all intermediate machines are OpenSSH agent protocol servers running <u>SSH2</u>. Destination servers must be running SSH2 but do not have to be OpenSSH agent protocol servers.

#### **Enabling the Agent**

To enable the agent options follow these steps:

- 1. On the local machine, start SecureCRT.
- 2. Open the Global Options dialog, and select the SSH2 category.
- 3. To enable the SSH2 Agent, check the **Add keys to agent** check box in the **Advanced** group.
- 4. To enable agent forwarding on a global basis, check the **Enable OpenSSH agent forwarding** check box in the **Advanced** group and click on the **OK** button to save your settings.
- 5. To enable agent forwarding on a per-session basis, open the <u>session</u> <u>options</u> dialog and select the <u>ssh2/Advanced</u> category. Check the **Enable OpenSSH agent forwarding** check box in the **Options** group and click on the **OK** button to save your settings. This is a tri-state option; If this option is set to the tri-state value (square), the setting from the <u>Global Options/SSH2</u> Category will be used.

#### **Flushing the Agent Cache**

To delete the public keys from the agent's cache you will need to assign

the SSH\_FLUSH\_AGENT command to a specified keyboard key. You can do this by following these steps:

- 1. From the Tools menu, select **Keymap Editor...**
- 2. In the Keymap Editor, select the key that you want to associate with the SSH\_FLUSH\_AGENT command. This should be a key that is not already assigned or frequently used such as F2 or F11 for example.
- 3. Press the **Map Selected Key...** button to open the Map Selected Key dialog.
- 4. From the **Function** list, select SSH Function.
- 5. From the **SSH Function** list, select SSH\_FLUSH\_AGENT and click the **OK** button.
- 6. Save the keymap that you created and close the **Keymap Editor**.
- 7. Restart SecureCRT using the new keymap.
- 8. Press the key you chose to flush the agent cache.

**Note:** There will be no feedback letting you know that the agent cache has been flushed except that the previously cached public keys will not longer work.

## **Xterm Escape Sequences**

SecureCRT supports many standard escape sequences. The following is a list of the xterm sequences supported by SecureCRT.

**Note:** The mouse tracking information is provided for users developing applications for remote systems in order to support mouse tracking. Typically this implementation will be transparent to the user. SecureCRT supports X10 mouse tracking.

General Sequences	Action
ESC ] 2 ; title BEL	Set text parameters; change window title to title
Mouse Tracking Sequences	Action
ESC [ ? 9 h	Send mouse X and Y on button press
ESC [ ? 1000 h	Send mouse X and Y on button press and release

## **Example Script: Changing the Window Title**

SecureCRT supports the xterm escape sequence for changing the session window title bar on the fly. The Perl script below shows how this can be done:

```
#! /usr/local/bin/perl
# title.pl - sends xterm escape sequence to
# change window title to @ARGV

$esc = "\x1b";
$bel = "\x7";

$txt = join(" ", @ARGV);
```

```
print $esc, "]2;", $txt, $bel;
```

This script can be placed in a shell alias to display the current working folder in the title bar of the session window:

```
alias cd 'cd \!\!*; prompt; ~/bin/title.pl $host\: `pwd` '
```

## **256-Color Extension Support**

SecureCRT supports the 256-color extensions introduced for xterm versions that need to support more than 16 colors. For 256-color xterm escape sequences to work, you must have the **ANSI Color** option enabled in the **Terminal/Emulation** category of the **Session Options** dialog.

The general form of the 256-color escape sequences is as follows.

To set foreground color:

```
ESC [ 38 ; 5 ; <color> m CSI 38 ; 5 ; <color> m To set background color:

ESC [ 48 ; 5 ; <color> m CSI 48 ; 5 ; <color> m
```

The following Perl script demonstrates using these sequences to lay out a "color cube" with a colored "X" in each color cell:

```
#!/usr/bin/perl
$ESC="\x1b";
$CSI="${ESC}\[";
for ($green=0;$green<6;$green++)
{
    for ($red=0;$red<6;$red++)
    {
        for ($blue=0;$blue<6;$blue++)
        {
            $bgcolor=16+($red*36)+($green*6)+$blue;
            $fgcolor=16-$red-$green-$blue;
            print "${CSI}48;5;${bgcolor}m";
            print "${CSI}38;5;${fgcolor}mX";
        }
```

```
print "${CSI}0m";
}
print "\n";
}
```



## 藡 Creating a Personal Font Map

SecureCRT provides a method for you to create a personal font map, in case the font maps available in the Character Encoding Option are not sufficient for your needs. This method consists of two tasks: creating a font map file and adding that file to a SecureCRT session.

#### **Creating a Font Map**

A font map is a text file of up to 256 lines (0 through 255). Each line in this file corresponds (by its position in the file) to a character cell in the selected codepage. The hexadecimal number on each line tells SecureCRT which Unicode character to display for that cell. For example, if the value 00df appeared on line 65 of the file, any "A" characters received from the remote machine would be displayed as " \( \mathbb{G}'' \) characters.

The line entry format is as follows:

```
<hexadecimal number><comments>
```

#### Example:

```
0.000
0001
00df "Sharp S"
0.0ff
```

Anything on the line after the hex number is ignored by SecureCRT, making the space useful for comments. Blank lines in the file will be ignored.

If you only want to translate a few characters, you can choose not to include all 256 lines in your map file, however, the entries must be sequential and all entries must be included through the last assignment you want to make. For example, if you wanted to change the characters displayed for lines 2 and 10, you still must include lines 0 through 10 in your file.

## Adding a Font Map to a Session

Font map files are session specific. To assign a font map to a session,

add a line like the following to your session's .ini file:

S: "User Font Map"=<path>

In this command, the "path" variable is the full path of your font map.



## Overview of Color Schemes

SecureCRT allows you to customize the foreground and background colors of both normal and bold text attributes. The foreground and background colors are defined in what is called a "color scheme" which can then be customized and assigned to a session . SecureCRT comes with a number of predefined color schemes. SecureCRT also allows you to create any number of personalized color schemes.

Note: If ANSI color is enabled for a session, the color scheme specified by the session will not be used. For information on how to enable, disable, or customize ANSI color settings, see the ANSI color settings in the **Global Options** and **Session Options** dialogs.



## 🗺 Creating or Modifying a Color Scheme

You can modify a color scheme from either the Global Options Or Session Options dialogs in SecureCRT.

From the <u>Session Options</u> dialog:

- Select the **Terminal/Appearance** Category.
- To create a color scheme, in the **Current color scheme** group, click on the New... button.
- To modify a color scheme, in the **Current color scheme** group, click on the Edit... button.

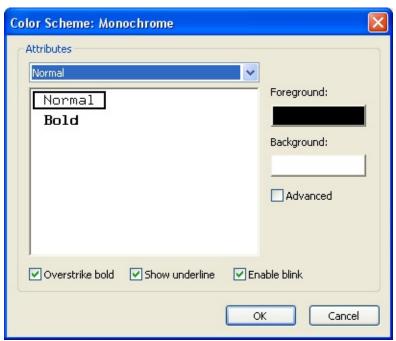
From the Global Options dialog:

- In the Global Options dialog, select the Terminal/Appearance Category and click on the **Color Schemes** subcategory.
- To create a color scheme, in the **Color scheme** panel, click on the **New...** button.
- To modify a color scheme, select the color scheme that you want to modify from the **Schemes** list and click on the **Edit...** button.

In either case, to create or modify a color scheme:

- In the Color Scheme dialog (shown below), select the text attribute (Normal, Bold, etc...) you want from among the list of attributes by using the drop-down box or by clicking on the attribute name as it is displayed in the attribute preview area. When an attribute is selected, it will be indicated with a box drawn around it (in the figure below, the Normal attribute is selected) in the attribute preview area.
  - To change the foreground color of the selected text attribute, click on the **Foreground** button (see below) and select the new foreground color from the **Color** dialog.
  - To change the background color of the selected text attribute, click on the **Background** button and select the new background color from the Color dialog.

Note: To customize the foreground and background colors of additional text attributes such as Underline, Bold Underline, Blink, Blink Bold, Blink Underline, and Blink Bold Underline, select the Advanced option in the Color Scheme dialog. The Color Scheme dialog also allows you to enable or disable the display of **Overstrike bold**, **Show underline** and **Enable blink** text attributes within the terminal screen (see below).



Color Scheme Dialog

## Overview of Keymaps

Terminal emulators, such as SecureCRT, use a keyboard emulation or "keymap" to map keystrokes from your local keyboard to the keystrokes that a remote host would expect. For example, assume you are a Windows NT user running a SecureCRT session that is configured with the VT220 terminal emulation and associated keymap. When you press the F1 key (which in NT would open the Help system), SecureCRT sends a Hold Screen command to the remote host as if you were pressing the F1 key on a VT220 terminal.

In SecureCRT, every session uses a keymap. Keymaps can also be used for more than one session. SecureCRT provides built-in keymaps for ANSI, Linux, SCO ANSI, VShell, VT100, VT102, VT220, WYSE50, and WYSE60 terminal emulation. You can also create custom keymaps using the Keymap Editor.

## Selecting a Keymap

SecureCRT automatically chooses a built-in keymap to use based on the terminal emulation selected in the Terminal/Emulation category of the Session options dialog. To select a different keymap for a session, check the **Select** an alternate keyboard emulation option in the Terminal/Emulation Category and select the built-in or custom keymap that you want to use.



## 藡 Customizing a Keymap

SecureCRT provides two mechanisms for customize the functions defined by a keymap: the Mapped Keys Category; and the Keymap Editor.

## Mapped Keys

In the Terminal/Emulation/Mapped Keys Category of the Session Options dialog, you can associate functions and arguments with specified keys or key combinations. These associations only apply to the session in use and will not carry over to other sessions. The associations made only supersede the current keymap and do not change the keymap itself.

You can also return a keymap back to its original configuration by clicking on the default button in the Map Selected Key dialog.

## **Keymap Editor**

The Keymap Editor allows you to modify an existing keymap or create a new one. In this way, you can make keymap customizations that can be used with multiple sessions.



## **Table of Keymap Functions**

Key combinations can be mapped to any one of the Send String, Run Script, Menu, Scroll Bar, Telnet, SSH, and VT function genres, or can be disabled completely.

You can also map a key that launches local commands. By assigning a key to the function Launch Application and entering the appropriate executable file, SecureCRT can open whatever application you need.

Function Genre	Function Name and Parameter	Action
Send String	N/A string	Sends "string".
Run Script	N/A scriptfile	Runs scriptfile.
Menu Function	MENU_CLEAR_SCREEN	Clears the terminal screen.
	MENU_CLEAR_SCROLLBACK	Clears the scrollback buffer.
	MENU_CLEAR_SCREEN_AND_SCROLLBACK	Clears both the terminal screen and the scrollback buffer.
	MENU_CONNECT	Opens a ne SecureCRT window and brings up th SecureCRT

	Connect dialo
MENU_COPY	Copies the text selecter in the SecureCRT terminal screen.
MENU_COPY_PASTE	Copies the text selecter in the SecureCRT terminal screen and pastes it back to the terminal screen.
MENU_DISCONNECT	Closes the current
MENU_EXIT	Exits SecureCRT
MENU_FIND	Opens the <b>Find</b> dialog for text searching.
MENU_GOTO_CHAT_SESSION_WINDOW	Switches focus between the chat and session windows.
	Toggles

MENU_LOG_SESSION	session
MENU_NEW_WINDOW	Opens a ne SecureCRT window.
MENU_PASTE	Pastes the contents of the Clipboal to the SecureCRT terminal screen.
MENU_PRINT_AUTO	Toggles Auto
MENU_PRINT_CANCEL	Turns Auto Pri off.
MENU_PRINT_EJECT_PAGE	Sends the contents of the pass-through printing buffer to the printer.
MENU_PRINT_SCREEN	Prints the contents of the SecureCRT
MENU_PRINT_SELECTION	Prints the current selection
	Opens a ne

MENU_QUICK_CONNECT	SecureCRT window and brings up th SecureCRT Quick Connect dialog.
MENU_RECENT_n	Opens the session indicated by the <i>n</i> <sup>th</sup> item on the Recently Used Sessions Li group of the <b>File</b> menu. Valid values for <i>n</i> are the integers 1-4
MENU_RECONNECT	Reconnects to the machine whose connection was most recently closed.
MENU_RESET	Resets the SecureCRT terminal emulator.
MENU_SAVE_SETTINGS_NOW	Saves session settings.
	Selects the

MENU_SELECT_ALL	contents of the scrollback buffer.
MENU_TAB_1	Sets focus t Tab 1.
MENU_TAB_2	Sets focus t Tab 2.
MENU_TAB_3	Sets focus t Tab 3.
MENU_TAB_4	Sets focus t Tab 4.
MENU_TAB_5	Sets focus t Tab 5.
MENU_TAB_6	Sets focus t Tab 6.
MENU_TAB_7	Sets focus t Tab 7.
MENU_TAB_8	Sets focus t Tab 8.
MENU_TAB_9	Sets focus t Tab 9.
MENU_TAB_10	Sets focus t Tab 10.
MENU_TAB_CLONE	Creates a new session that is identical to the current

	session. Fo more information, see the <u>Sessi</u> <u>Tabs</u> topic.
MENU_TAB_CLOSE	Closes the active tab.
MENU_TAB_NEW	Creates a new tab.
MENU_TAB_NEW_SFTP	Opens a ne tab containing a SFTP command line. (available ir SecureCRT only)
MENU_TAB_NEXT	Changes th focus to the next tab.
MENU_TAB_PREV	Changes th focus to the previous tak
MENU_TAB_RENAME	Allows you rename the active tab.
	Starts VanDyke Software's SecureFX® with the same

MENU_TAB_SECUREFX	session configuratio if the program is installed (available ir SecureCRT only)
MENU_TOGGLE_ALPHA_TRANSPARENCY	Toggles the active window transparence between ful opaque and the transparence value set in the Terminal / Appearance / Advanced category of the Global Options dialog
MENU_TOGGLE_CHAT_WINDOW	Toggles display of th SecureCRT chat windov
MENU_TOGGLE_FULL_SCREEN	Toggles full screen mode.
MENU_TOGGLE_MENU_BAR	Toggles display of th SecureCRT menu bar.
	Toggles

MENU_TOGGLE_SESSION_TABS	display of tabs.
MENU_TOGGLE_STATUS_BAR	Toggles display of the. SecureCRT status bar.
MENU_TOGGLE_TOOLBAR	Toggles display of th SecureCRT toolbar.
MENU_WINDOW_n	Cycles focu to the <i>n</i> <sup>th</sup> SecureCRT window as designated the <b>Windov</b> menu. Valid values for <i>n</i> are integers 1-0, where the value 0 indicates the 10 <sup>th</sup> window
MENU_WINDOW_NEXT	Cycles focu to the next SecureCRT window
MENU_WINDOW_PREV	Cycles focu to the previous SecureCRT window.

Scroll Bar Function	SB_BEGIN	Scrolls to th beginning o the scrollback buffer.
	SB_COLUMN_LEFT	Moves the contents of the terminal screen right by one column. Same as clicking on the horizont scroll bar learrow.
	SB_COLUMN_RIGHT	Moves the contents of the terminal screen left k one column Same as clicking on the horizont scroll bar right arrow.
	SB_END	Scrolls to th end of the scrollback buffer.
		Moves the contents of the terminal

SB_HALF_PAGE_DOWN	screen up b one-half of a page.
SB_HALF_PAGE_UP	Moves the contents of the terminal screen dow by one-half of a page.
SB_LINE_DOWN	Moves the contents of the terminal screen up b one line. Same as clicking on the vertical scroll bar down arrow
SB_LINE_UP	Moves the contents of the terminal screen dow by one line. Same as clicking on the vertical scroll bar up arrow.
	Moves the contents of the terminal screen up b one page. Same as

SB_PAGE_DOWN	clicking between the down arrow and the slid in the vertic scroll bar "trough".
SB_PAGE_LEFT	Moves the contents of the terminal screen right by one page (number of logical columns). Same as clicking between the left arrow and the slid on the horizontal scroll bar.
SB_PAGE_RIGHT	Moves the contents of the terminal screen left to one page (number of logical columns). Same as clicking between the right arrow and the slid on the

		horizontal scroll bar.
	SB_PAGE_UP	Moves the contents of the terminal screen dow by one page Same as clicking between the up arrow an the slider in the vertical scroll bar "trough".
Telnet Function	TN_ABORT_OUTPUT	Sends the Abort Outpu (AO) controfunction to the remote host.
	TN_ARE_YOU_THERE	Sends the Are You There ( AYT control function to the remote host.
	TN_BREAK	Sends the BREAK control function to the remote host.

	TN_INTERRUPT_PROCESS	Sends the Interrupt Process (IP control function to the remote host.
	TN_SYNCH	Sends the SYNCH control function to the remote host. Used combination with the IP control function to ensure that the host and client are in synch with each other after an IP control function is called.
VT Function		Each VT function sends the key codes associated with the name of the VT function.
	VT_AUTO_PRINT	Toggles Auto Print.

VT_BREAK	See Telnet function TN_BREAK
VT_CURSOR_DOWN	Sends ESC [ B
VT_CURSOR_LEFT	Sends ESC [ D
VT_CURSOR_RIGHT	Sends ESC [ C
VT_CURSOR_UP	Sends ESC [ A
VT_F6	Sends ESC [ 1 7 ~
VT_F7	Sends ESC [ 1 8 ~
VT_F8	Sends ESC [ 1 9 ~
VT_F9	Sends ESC [ 2 0 ~
VT_F10	Sends ESC [ 2 1 ~
VT_F11	Sends ESC [ 2 3 ~
VT_F12	Sends ESC [ 2 4 ~
VT_F13	Sends ESC [ 2 5 ~

VT_F14	Sends ESC [ 2 6 ~
VT_F15	Sends ESC [ 2 8 ~
VT_F16	Sends ESC [ 2 9 ~
VT_F17	Sends ESC [ 3 1 ~
VT_F18	Sends ESC [ 3 2 ~
VT_F19	Sends ESC [ 3 3 ~
VT_F20	Sends ESC [ 3 4 ~
VT_FIND	Sends ESC [ 1 ~
VT_HOLD_SCREEN	Performs a Hold Screer
VT_INSERT_HERE	Sends ESC [ 2 ~
VT_KEYPAD_0	Sends 0
VT_KEYPAD_1	Sends 1
VT_KEYPAD_2	Sends 2
VT_KEYPAD_3	Sends 3
VT_KEYPAD_4	Sends 4

VT_KEYPAD_5		Sends 5
VT_KEYPAD_6		Sends 6
VT_KEYPAD_7		Sends 7
VT_KEYPAD_8		Sends 8
VT_KEYPAD_9		Sends 9
VT_KEYPAD_COMI	MA	Sends a ","
VT_KEYPAD_ENTE	ER .	Sends a CR
VT_KEYPAD_MINU	S	Sends a "-"
VT_KEYPAD_PERIO	OD	Sends a "."
VT_NEXT_SCREEN	J	Sends ESC [ 6 ~
VT_PF1		Sends ESC 0 P (VT100 PF1 key).
VT_PF2		Sends ESC 0 Q (VT100 PF2 key).
VT_PF3		Sends ESC 0 R (VT100 PF3 key).
VT_PF4		Sends ESC 0 S (VT100 PF4 key).
VT_PREV_SCREEN	N	Sends ESC [ 5 ~

	VT_PRINT_SCREEN		Performs Print Scre	
	VT_REMOVE		Sends ES	SC
	VT_SELECT		Sends ES	SC
Disable	N/A		Disables selected l or key combinat	ke <u>'</u>
SSH Function		SSH functions will only work with SSH connections.		
	SSH_BREAK	Sends the BREAK control function to the remote host.		
	SSH_CLOSE_SESSION	Closes the current terminal session.		
	SSH_FLUSH_AGENT	Flushes the SSH agent cache.		
	SSH_NEW_SESSION	Opens a new terminal session.		
	SSH_NEXT_SESSION	Switches to the next terminal session.		
	SSH_PREV_SESSION	Switches to the previous terminal session.		



# ASCII Control Codes

This table lists the values for the ASCII control Characters (ASCII decimal values 0 - 31). Decimal and Hex values are provided to aid in conversion.

Note: SecureCRT requires octal values.

CTRL	Decimal	Hex	Octal
^@	0	0x00	\000
^A	1	0x01	\001
^B	2	0x02	\002
^C	3	0x03	/003
^D	4	0x04	\004
^E	5	0x05	\005
^F	6	0x06	\006
^G	7	0x07	\007
^H	8	0x08	\010
^	9	0x09	\011
^J	10	0x0A	\012
^K	11	0x0B	\013
^L	12	0x0C	\014
^M	13	0x0D	\015
^N	14	0x0E	\016
^O	15	0x0F	\017
^P	16	0x10	\020
^Q	17	0x11	\021
^R	18	0x12	\022
^S	19	0x13	\023
^T	20	0x14	\024
^U	21	0x15	\025

^V	22	0x16	\026
^W	23	0x17	\027
^X	24	0x18	/030
^Y	25	0x19	\031
^Z	26	0x1A	\032
^[	27	0x1B	/033
^/	28	0x1C	\034
^]	29	0x1D	\035
^^	30	0x1E	\036
<	31	0x1F	\037



# Send String and Function Key Commands

The following commands can be included as part of a keymap, button, or antiidle string.

Command	Action
\r	Sends a carriage return (CR)
\n	Sends a linefeed (LF)
\p	Sends a backspace
le	Sends an escape
\t	Sends a tab
\v	Pastes the contents of the Clipboard to the active session window
\p	Pauses for one second

Note: When mapping a key, you can send arbitrary octal codes using Send String commands.

# Emulation-Specific Keymappings

The SecureCRT built-in keymaps for Linux, VT220, and VT100 have several key assignments that do not match the traditional Windows key assignments. The following tables list those emulation-specific keys.

Linux				
Key	Literal Value	Octal Value	Hexidecimal Value	
F1	ESC[[A	\033\133\133\101	0x1B 0x5B 0x5B 0x41	
F2	ESC[[B	\033\133\133\102	0x1B 0x5B 0x5B 0x42	
F3	ESC[[C	\033\133\133\103	0x1B 0x5B 0x5B 0x43	
F4	ESC[[D	\033\133\133\104	0x1B 0x5B 0x5B 0x44	
F5	ESC[[E	\033\133\133\105	0x1B 0x5B 0x5B 0x45	
VT220				
Key	Literal Value	Octal Value	Hexidecimal Value	
F1	VT_HOLD_SCREEN	J II	local function - nothing sent to remote	
F2	VT_PRINT_SCREE	local function - nothing sent to remote		
F3	VT_PF3 \033\117\122		0x1B 0x4F 0x52	
F4	VT_PF4	\033\117\123	0x1B 0x4F 0x53	
		protocol func	tion - no escape	

F5	TN_BREAK	sequence	
INS	VT_FIND	\033\133\061\176	0x1B 0x5B 0x31 0x7F
НОМЕ	VT_INSERT_HERE	\033\133\062\176	0x1B 0x5B 0x32 0x7F
PG_UP	VT_REMOVE	\033\133\063\176	0x1B 0x5B 0x33 0x7F
DEL	VT_SELECT	\033\133\064\176	0x1B 0x5B 0x34 0x7F
END	VT_PREV_SCREEN	\033\133\065\176	0x1B 0x5B 0x35 0x7F
PG_DN	VT_NEXT_SCREEN	\033\133\066\176	0x1B 0x5B 0x36 0x7F
		1	1

VT100			
Key	Literal Value	Octal Value	Hexidecimal Value
F1	VT_PF1	\033\117\120	0x1B 0x4F 0x50
F2	VT_PF2	\033\117\121	0x1B 0x4F 0x51
F3	VT_PF3	\033\117\122	0x1B 0x4F 0x52
F4	VT_PF4	\033\117\123	0x1B 0x4F 0x53
NM_LK	VT_PF1	\033\117\120	0x1B 0x4F 0x50
/	VT_PF2	\033\117\121	0x1B 0x4F 0x51
*	VT_PF3	\033\117\122	0x1B 0x4F 0x52
_	VT_PF4	\033\117\123	0x1B 0x4F 0x53

# Determine Escape Sequence Key Codes

If you need to determine the escape sequence for a particular key that a terminal emulator sends to a remote system (e.g., for use in scripting or custom key mappings), there are a couple of ways to do this.

Using any terminal emulator, connect to a remote system running UNIX. You should be able to use the following command from your shell to determine what sequence of characters the remote system receives when a particular key is pressed:

For example, if you press the PF1 key from a VT100 terminal (or an emulator which maps the PF1 key), you should see:

(This is typically read: ESC O P.)

In SecureCRT, using the VT100 keyboard emulation, this can be sent by pressing the NUM LOCK key.

Using SecureCRT's ActiveX scripting support with VBScript, you could send this sequence using the following command string:

```
crt.Screen.Send Chr(27) & "OP"
```

Using SecureCRT's Python scripting support, you could send this sequence using the following command string:

```
crt.Screen.Send(chr(27) + "OP")
```

In the examples above, "27" is the ASCII decimal value for the ESC character.

Similarly, in SecureCRT's Session Options/Terminal/Emulation/Mapped Keys dialog or Keymap Editor, you can use "\e" to represent the ESC character, or use the octal ASCII value "\033". More information about this can be found in the Table of Keymap Functions and Send String and Function Key Commands Sections.

Using the Keymap Editor, you can also find out the default mappings for the keys you press. For example, with the VT100 keymap loaded, clicking on the F1 button in the Keymap Editor will show you "VT PF1", which is an internal alias for the sequence "ESC O P" (or "^[OP", "\033OP", etc.).

It is important to understand that there are no absolute values that any particular key sends, so what one remote system accepts for PF1 may not be consistent on a different system. Moreover, there are many ways of representing the same escape sequence, including decimal (e.g., "Chr(27)" in VBScript), backslash escape characters (e.g., "\e" for ESC in SecureCRT), and octal (e.g., "\033OP" or "\033\117\120").



# Creating ActiveX Scripts

SecureCRT is capable of hosting "ActiveX Script" engines. The most common ActiveX script engines are VBScript and JScript (Microsoft's version of JavaScript), both of which are freely available from Microsoft. Chances are you already have them installed if you've installed Internet Explorer 4.0.

ActiveX script engines communicate with SecureCRT via standard interfaces. Therefore, SecureCRT can host any compliant script engine to run your scripts. The advantage of this approach is that you can script SecureCRT using the language of your choice. If an ActiveX script engine is available for your preferred scripting language, you can write scripts that will work with SecureCRT.

**Note:** To run a logon script in SecureCRT, you must first turn off the Automate logon option in the connection/Logon Actions Category of the Session Options dialog and then, on the same dialog, check the **Logon script** check box and choose the script that you want to run.

**Note2:** A script cannot be run from another script. This includes scripts started manually and logon scripts.

Scripts for SecureCRT are text files that you create with your text editor.

A script header is not required as long as the file extension is registered in the registry with a script engine (as common file extensions are). If you want to use a header (for example, if you are using a file without an extension or with an extension that is not common), the header must begin on the first line of the script.

Script headers will be used by SecureCRT to identify which script language the script is written in and the version of SecureCRT scripting interface. Each line of the script header must begin with a (#) character. A SecureCRT script header includes a **\$language** line that identifies the script engine and an \$interface line to identify SecureCRT's interface version.

The syntax of the script header is always the same regardless of the script language you are using.

A simple but complete SecureCRT script with a header that identifies it as VBScript is shown below:

```
# $language = "VBScript" # $interface = "1.0"

Sub Main
   'Display SecureCRT's version
   MsgBox "SecureCRT version is: " & crt.Version
End Sub
```

**Note:** A SecureCRT script header may also contain blank lines that begin with (#).

The quoted string following **\$language** identifies the script engine. If you are writing scripts that use Microsoft's JScript language, the appropriate identifier is JScript. If you are using another script engine you'll need to consult the documentation for the identifier for that language. Currently the script header should specify version 1.0 for **\$interface**. Future versions of SecureCRT may support other versions. The example script above has a subroutine named main where all of the script's code is located. When SecureCRT executes scripts it always attempts to run a main routine if you have defined one.

It is not a requirement that you place your code within a main however there may be reasons why you would want to do this. The VBScript and JScript engines will parse and execute global script code (script code you have defined outside of any subroutine) before your main is executed. If you have "initialization" code that you want to ensure has been completely executed before your actual script code begins, it may be useful to place your initialization code at the global level. This will ensure that your initialization code will all execute before your main code runs.

Another reason you may want a main routine is to allow your scripts a way of aborting themselves in case of problems. In VBScript there is no built-in way of exiting a script at the global level. However, if you want to exit a subroutine it is possible to use the Exit Sub syntax to do so. For example, in VBScript:

```
condition = DoSomething()
If condition = 0 Then
' Error, bailout
Exit Sub
End If
```

Sub Main

#### End Sub

When the main routine ends the script has finished running. By placing your code within a main you have the option of invoking Exit Sub whenever it might be necessary.

The previous script samples are written in VBScript. The remainder of code samples in this document are all written in VBScript unless it is stated otherwise. The properties and methods of SecureCRT's interface can be used as documented by any compatible scripting language.

For files stored on NTFS file systems, the script header can optionally be stored in an NTFS alternate data stream called header.txt. One way to do this is to choose **Run** from the Windows Start menu and enter the following:

notepad myscript.txt:header.txt

Script files can be stored in UTF-8 or Unicode format. Storing a script file in UTF-8 or Unicode format enables characters from languages such as Chinese or Japanese to be included in the script.

For examples of SecureCRT scripts, see the sample scripts in the SecureCRT Installation\Scripts folder and on the VanDyke Software website in the Scripting FAO Section.

# Creating Python Scripts

SecureCRT contains an embedded Python interpreter, which means that Python scripts can be run from SecureCRT without having to install additional software.

**Note:** To run a logon script in SecureCRT, you must first turn off the Automate logon option in the Connection/Logon Actions category of the Session Options dialog and then, on the same dialog, check the **Logon script** check box and choose the script that you want to run.

**Note2:** A script cannot be run from another script. This includes scripts started manually and logon scripts.

Scripts for SecureCRT are text files that you create with your text editor.

A script header is not required for Python scripts as long as the file extension ".py" is used. If you want to use a header (for example, if you are using a file without an extension or with an extension that is not common), then the header must begin on the first line of the script.

Script headers will be used by SecureCRT to identify which script language the script is written in and the version of SecureCRT scripting interface. Each line of the script header must begin with a pound/hash (#) character. A SecureCRT script header includes a \$language line that identifies the script engine and an \$interface line to identify SecureCRT's interface version.

The syntax of the script header is always the same regardless of the script language you are using.

A simple but complete SecureCRT script with a header that identifies it as Python is shown below:

```
# $language = "Python"
# $interface = "1.0"
def main():
 # Display SecureCRT's version
 crt.Dialog.MessageBox("SecureCRT version is: " + crt.Version)
main()
```

Note: A SecureCRT script header may also contain blank lines that begin with #.

The quoted string following **\$language** identifies the script engine. For Python scripts, the appropriate identifier is Python. Currently the script header should specify version 1.0 for **\$interface**. Future versions of SecureCRT may support other versions

It is not a requirement that you place your code within a main routine however there may be reasons why you would want to do this. The Python interpreter will parse and execute global script code (script code you have defined outside of any subroutine) before your main is executed. If you have "initialization" code that you want to ensure has been completely executed before your actual script code begins, it may be useful to place your initialization code at the global level. This will ensure that your initialization code will all execute before your main code runs.

Another reason you may want a main routine is to allow your scripts a way of aborting themselves in case of problems. In Python for SecureCRT, there is no built-in way of exiting a script at the global level. However, if you want to exit a subroutine it is possible to use the return syntax to do so. For example:

```
def main():
  condition = DoSomething()
  if (condition == 0):
    # Error, bailout
  return
main()
```

For examples of SecureCRT scripts, see the sample scripts on the VanDyke Software website in the <u>Scripting FAQ</u> section.

Many excellent Python resources can be found on the Internet. Here are a few you may find useful:

http://docs.python.org/reference/index.html http://diveintopython.org http://wiki.python.org/main/BeginnersGuide



# **Handling Script Errors**

When scripting, SecureCRT provides several Application script methods to aid you in managing errors. The first is GetLastError. This method returns the error code of the script exception that most recently happened. The next method is GetLastErrorMessage, which returns the error text of the script exception that most recently happened. And, to help you determine where an error came from, SecureCRT supports the ClearLastError, method which resets the response from GetLastError and GetLastErrorMessage to ERROR\_SUCCESS and "The operation completed successfully."



# Overview of SecureCRT Script Objects

Scripts interact with SecureCRT by invoking properties and methods on SecureCRT's "top-level" or Application object or by invoking the properties and methods on "sub-objects" available through SecureCRT's application object. SecureCRT's application object is accessed in scripts with the name 'crt'. Properties and methods on SecureCRT's sub-objects may be accessed by creating a reference to a sub-object, or through the use of VBScript's multiple dot syntax. For example:

```
Dim dlg
  Set dlg = crt.Dialog
  dlg.Prompt("Login:")
In Python, the lines would look like:
```

dlg = crt.Dialog dlg.Prompt("Login:")

Or, in VBScript and Python, without creating the reference:

crt.Dialog.Prompt("Login:")

For a complete reference to all SecureCRT script objects, see the "Script Objects Reference" section in this book.



# SendKeys Keystroke Table

Some keystrokes do not generate characters (such as ENTER and TAB). Some keystrokes represent actions (such as BACKSPACE and BREAK). To send these kinds of keystrokes, send the arguments shown in the following table:

Key	Argument	
BACKSPACE	{BACKSPACE}, {BS}, or {BKSP}	
BREAK	{BREAK}	
CAPS LOCK	{CAPSLOCK}	
DEL or DELETE	{DELETE} or {DEL}	
DOWN ARROW	{DOWN}	
END	{END}	
ENTER	{ENTER} or ~	
ESC	{ESC}	
HELP	{HELP}	
HOME	{HOME}	
INS or INSERT	{INSERT} or {INS}	
LEFT ARROW	{LEFT}	
NUM LOCK	{NUMLOCK}	
PAGE DOWN	{PGDN}	
PAGE UP	{PGUP}	
PRINT SCREEN	{PRTSC}	
RIGHT ARROW	{RIGHT}	

SCROLL LOCK	{SCROLLLOCK}	
TAB	{TAB}	
UP ARROW	{UP}	
F1, F2, F16	{F1}, {F2}, {F16}	
0, 1, 9 on number pad	{NUM_0}, {NUM_1}, {NUM_9}	
. on number pad	{NUM}	
/ on number pad	{NUM_/}	
* on number pad	{NUM_*}	
- on number pad	{NUM}	
+ on number pad	{NUM_+}	
ENTER on number pad	{NUM_ENTER}	
HOME on number pad	{NUM_HOME}	
PAGE UP on number pad	{NUM_PGUP}	
END on number pad	{NUM_END}	
PAGE DOWN on number pad	{NUM_PGDN}	
UP ARROW on number pad	{NUM_UP}	
DOWN ARROW on number pad	{NUM_DOWN}	
LEFT ARROW on number pad	{NUM_LEFT}	
RIGHT ARROW on number pad	{NUM_RIGHT}	



# **Application Object**

# **Description**

Top-level object. Provides access to all of SecureCRT's properties and methods.

# **Syntax**

crt.Property [ = expression ]
crt.Method([arglist])

#### **Remarks**

The name crt is used to access all of SecureCRT's top-level properties and methods. Some of these top-level properties return references to other objects. When a script is run by SecureCRT it is not necessary to create the crt object. It is automatically part of the script namespace.

# **Application Object Properties and Methods**

<u>Properties</u>	Methods
<u>ActivePrinter</u>	ClearLastError
<u>Arguments</u>	<u>GetActiveTab</u>
Clipboard	GetLastError
Dialog	GetLastErrorMessage
<u>FileTransfer</u>	<u>GetScriptTab</u>
Screen	<u>GetTabCount</u>
<u>ScriptFullName</u>	<u>GetTab</u>
Session	<u>OpenSessionConfiguration</u>
Version	Quit
Window	Sleep

# **Properties**

#### **ActivePrinter**

# **Description** Returns or sets the name of the active printer. Remarks Read/Write string property. **VBScript Syntax** crt.ActivePrinter [ = printerName ] **Example** If crt.ActivePrinter <> "\\SERVER\\InkJet2" Then crt.ActivePrinter = "\\SERVER\InkJet2" MsgBox "Updated Printer" End If **Python Syntax** crt.ActivePrinter [ = printerName ] **Example** if crt.ActivePrinter != "\\SERVER\InkJet2": crt.ActivePrinter = "HP LaserJet 8000 Series PCL 5"

#### **Arguments**

#### **Description**

Returns a reference to SecureCRT's Arguments object.

crt.Dialog.MessageBox("Updated Printer")

#### Remarks

In VBScript, object variables assigned from the Arguments property require the "Set" syntax. See the Arguments object documentation for a description of its properties.

#### **VBScript**

#### **Syntax**

Set *object* = crt.Arguments

# **Python**

#### **Syntax**

*object* = crt.Arguments

# **Clipboard**

# Description

Returns a reference to SecureCRT's Clipboard object.

#### Remarks

In VBScript, object variables assigned from the Clipboard property require the "Set" syntax. See the Clipboard object documentation for a description of its properties.

#### **VBScript**

#### **Syntax**

Set object = crt.Clipboard

#### **Python**

#### **Syntax**

object = crt.Clipboard

#### **Dialog**

#### **Description**

Returns a reference to SecureCRT's Dialog object.

#### Remarks

In VBScript, object variables assigned from the Dialog property require the "Set" syntax. See the Dialog object documentation for a description of its properties and methods.

#### **VBScript**

#### **Syntax**

Set *object* = crt.Dialog

# Python

# **Syntax**

object = crt.Dialog

#### **FileTransfer**

#### **Description**

Returns a reference to SecureCRT's FileTransfer object.

#### Remarks

In VBScript, object variables assigned from the FileTransfer property require the "Set" syntax. See the FileTransfer object documentation for a description of its properties and methods.

#### **VBScript**

# **Syntax**

Set *object* = crt.FileTransfer

# **Python**

## **Syntax**

object = crt.FileTransfer

#### Screen

#### **Description**

Returns a reference to SecureCRT's Screen object.

#### Remarks

In VBScript, object variables assigned from the Screen property require the "Set" syntax. See the Screen object documentation for a description of its properties and methods.

#### **VBScript**

**Syntax** 

Set *object* = crt.Screen

#### Python

**Syntax** 

*object* = crt.Screen

#### **ScriptFullName**

#### Description

Returns the full path and filename of the currently running script.

#### Remarks

None.

## **VBScript**

**Syntax** 

Set object = crt.ScriptFullName

#### **Python**

**Syntax** 

object = crt.ScriptFullName

#### **Session**

#### Description

Returns a reference to SecureCRT's Session object.

#### Remarks

In VBScript, object variables assigned from the Session property require the "Set" syntax. See the Session object documentation for a description of its properties and methods.

#### **VBScript**

**Syntax** 

Set *object* = crt.Session

# Python

**Syntax** 

```
object = crt.Session
```

#### Version

#### Description

Returns a string containing SecureCRT's version.

#### Remarks

Read-only string property

#### **VBScript**

**Syntax** 

crt.Version

#### **Example**

MsgBox "The version of SecureCRT is: " & crt. Version

#### **Python**

#### **Syntax**

crt.Version

#### **Example**

crt.Dialog.MessageBox("The version of SecureCRT is: " +
crt.Version)

#### Window

#### **Description**

Returns a reference to SecureCRT's Window object.

#### Remarks

In VBScript, object variables assigned from the Window property require the "Set" syntax. See the Window object documentation for a description of its properties and methods.

# **VBScript**

# **Syntax**

Set *object* = crt.Window

# Python

## **Syntax**

object = crt.Window

# **Methods**

#### **ClearLastError**

## Description

Resets the response from GetLastError and

GetLastErrorMessage to ERROR\_SUCCESS and "The operation completed successfully."

#### Remarks

This is intended to be used in conjunction with GetLastErrorMessage so that the user can determine where the error came from.

#### **VBScript**

**Syntax** 

crt.ClearLastError

#### Python

**Syntax** 

crt.ClearLastError()

#### **GetActiveTab**

#### Description

Returns the Tab object associated with the tab or tiled session window that is currently selected in the GUI.

#### Remarks

None.

# **VBScript**

**Syntax** 

Set object = crt.GetActiveTab

# Python

**Syntax** 

object = crt.GetActiveTab()

#### **GetLastError**

#### Description

Returns the error code of the script exception that most recently happened.

#### Remarks

This is most useful when the script is in the "resume next" error mode, meaning the script keeps executing even if errors happen.

#### **VBScript**

#### **Syntax**

varname = crt.GetLastError

# Example

```
Sub Main()
         on error resume next
         CRT.Session.ConnectInTab("nowhere")
         errcode = crt.GetLastError
         crt.Dialog.MessageBox "Error Code: " &
    errcode
    End Sub
Python
  Syntax
    varname = crt.GetLastError()
  Example
    def main():
       try:
         crt.Session.ConnectInTab("nowhere")
       except ScriptError:
         errcode = crt.GetLastError()
      crt.Dialog.MessageBox("Error Code: " + str(errcode))
    main()
```

#### <u>GetLastErrorMessage</u>

#### **Description**

Returns the error text of the script exception that most recently happened.

#### Remarks

This is most useful when the script is in the "resume next" error mode, meaning the script keeps executing even if errors happen.

#### **VBScript**

# **Syntax**

```
varname = crt.GetLastErrorMessge
Example
```

```
Sub Main()
    on error resume next
    CRT.Session.ConnectInTab("nowhere")
    errmsg = crt.GetLastErrorMessage
    crt.Dialog.MessageBox "Error Message: " &
errmsg
End Sub
```

```
This example would display:
         Error Message: Connection failed
  Python
    Syntax
       varname = crt.GetLastErrorMessage()
    Example
       def main():
           try:
                crt.Session.ConnectInTab("nowhere")
           except ScriptError:
                errmsg = crt.GetLastErrorMessage()
           crt.Dialog.MessageBox("Error Message: " +
       errmsg)
       main()
       This example would display:
         Error Message: Connection failed
GetScriptTab
  Description
    Returns the tab or tiled session window from which the script was
    started.
  Remarks
    None.
  VBScript
    Syntax
       Set object = crt.GetScriptTab
    Example
       Set ScriptTab = crt.GetScriptTab
       MsgBox "Tab which started this script is tab number: " &
       ScriptTab.index
  Python
    Syntax
      object = crt.GetScriptTab()
    Example
       ScriptTab = crt.GetScriptTab()
       crt.Dialog.MessageBox("Tab which started this
```

#### **GetTabCount**

#### **Description**

Returns the number of tabs or tiled session windows (connected or not) that exist in the current SecureCRT window.

#### Remarks

Return value will always be greater than 0 (zero).

#### **VBScript**

**Syntax** 

varname = crt.GetTabCount

#### **Python**

**Syntax** 

varname = crt.GetTabCount()

#### **GetTab**

#### **Description**

Returns the tab object of the specified index.

#### Remarks

This does not bring the tab or tiled session window to the foreground. When sessions are tabbed, the index for each tab object matches its position in the tab bar. When sessions are tiled, the indexes of the tab objects may not match the indexes when tabbed, but will remain consistent while the sessions are tiled.

#### **VBScript**

Syntax

Set *object* = crt.GetTab(*arg*)

# **Python**

**Syntax** 

varname = crt.GetTab(arg)

# **OpenSessionConfiguration**

## Description

Loads the configuration for the specified session.

#### Remarks

SessionPath is a string parameter that is the relative path of

```
the session. Returns a Config object. If SessionPath is not
    specified, the Default session's configuration object is returned.
    To access the session configuration associated with an active
    connection, use crt.Session.Config or
    objTab.Session.Config.
  VBScript
    Syntax
       Set object.OpenSessionConfiguration [SessionPath]
    Example
       Set objConfig1 =
       crt.OpenSessionConfiguration("Routers\Telnet\S
       Set objDConfig =
       crt.OpenSessionConfiguration("Default")
  Python
     Syntax
       object.OpenSessionConfiguration([SessionPath])
     Example
       objConfig1 =
       crt.OpenSessionConfiguration("Routers\Telnet\S
       obiDConfig =
       crt.OpenSessionConfiguration("Default")
Quit
  Description
    Causes SecureCRT to exit.
  Remarks
    A script that invokes Quit will be terminated immediately and
    cause SecureCRT to exit.
  VBScript
     Syntax
       crt.Quit
  Python
     Syntax
       crt.Quit()
```

# <u>Sleep</u>

# Description

Specifies the time (in milliseconds) to pause the script's

execution.

#### Remarks

The amount of time that it takes to execute the Sleep() method itself is a factor in how long the script pauses. Therefore, the accuracy of invoking Sleep with small values of 1 or 10 milliseconds will be affected by this overhead.

## **VBScript**

#### **Syntax**

crt.Sleep millisec

### **Example**

' Send a CR and pause for one second crt.Screen.Send vbCr crt.Sleep 1000

#### **Python**

#### **Syntax**

crt.Sleep(millisec)

## **Example**

# Send a newline and pause for one second
crt.Screen.Send("\n")
crt.Sleep(1000)



# **Description**

The Arguments object allows scripts to access arguments that are passed to the script by one or more SecureCRT /ARG command-line options.

# **Syntax**

crt.Arguments.*Property* crt.Arguments.*Method*(arg)

#### Remarks

The SecureCRT /ARG command-line option allows you to write common scripts where specific values used by the scripts (such as a hostname or port) are passed to the script through the SecureCRT command line. Script arguments are passed as an argument to each /ARG <argument> option.

# **Arguments Object Properties and Methods**

<u>Properties</u> <u>Methods</u> <u>Count</u> <u>GetArg</u>

# **Properties**

#### Count

# Description

Returns the number of arguments passed to SecureCRT using SecureCRT's /ARG command-line option.

#### Remarks

Read-only numeric property.

If SecureCRT is started with no /ARG command-line options, the crt.Arguments.Count property is set to 0.

# **VBScript**

# **Syntax**

crt.Arguments.Count

# **Python Syntax**crt.Arguments.Count

#### **Methods**

#### **GetArg**

#### **Description**

Returns the argument data associated with each /ARG command-line option passed to SecureCRT.

#### Remarks

The GetArg method returns the data passed to SecureCRT. The first argument passed to SecureCRT is retrieved by calling GetArg(0). The last argument is retrieved by passing the count returned by crt.Arguments.Count minus 1.

**Note:** The GetArg method is the default method on the Arguments object therefore it will be invoked implicitly if it is not named. This means the following two statements are equivalent:

MsgBox crt.Arguments.GetArg(0) MsgBox crt.Arguments(0)

#### **VBScript**

## **Syntax**

crt.Arguments.GetArg(number)

# **Example**

#\$language = "VBScript" #\$interface = "1.0"

Sub Main()

- ' This script should be run by SecureCRT that has two
- '/ARG parameters for a hostname and port, for example:

<sup>&#</sup>x27;SecureCRT.exe /ARG myhostname /ARG 5555

```
' Detect lack of arguments.
       If crt.Arguments.Count <> 2 Then
          MsgBox "This script requires hostname and port
          arguments"
          Exit Sub
       End If
       MsgBox "Connecting to hostname: " & crt.Arguments(0)
       MsgBox "Connecting to port: " & crt.Arguments(1)
       ...
     End Sub
Python
  Syntax
     crt.Arguments.GetArg(number)
  Example
     #$language = "Python"
     #$interface = "1.0"
     def main():
       # This script assumes that SecureCRT was run with two
       # /ARG parameters for a hostname and port, such as:
       #
       # SecureCRT.exe /ARG myhostname /ARG 5555
       # Detect lack of arguments.
       if crt.Arguments.Count != 2:
          crt.Dialog.MessageBox(
           "This script requires hostname and port " +
           "arguments")
          return
       crt.Dialog.MessageBox(
           "Connecting to hostname: " +
           crt.Arguments[0])
       crt.Dialog.MessageBox(
```

```
"Connecting to port: " + crt.Arguments[1])
...
main()
```

# **Description**

The Clipboard object provides access to the application's clipboard.

# **Syntax**

```
crt.Clipboard.Property [ = expression ]
```

# **Clipboard Object Properties**

```
Properties

Format

CF_OEMTEXT

CF_TEXT

CF_UNICODETEXT

DEFAULTFORMAT

VDS_TEXT

Text
```

# **Properties**

#### **Format**

#### Description

Returns or sets the clipboard format.

#### Remarks

```
Possible formats on Windows are: CF_TEXT, CF_OEMTEXT, CF_UNICODETEXT, and VDS_TEXT.
```

Possible formats on Mac are: CF\_UNICODETEXT and VDS TEXT.

The following command restores the value to the global format for the clipboard:

```
crt.Clipboard.Format =
crt.Clipboard.DEFAULTFORMAT
```

# **VBScript**

```
Example
    #$language = "VBScript"
    #$interface = "1.0"
    crt.Screen.Synchronous = True
    Sub Main
      ' Get the current clipboard format
      format = crt.Clipboard.Format
        Set the clipboard format to VDS_TEXT:
      crt.Clipboard.Format =
      crt.Clipboard.VDS TEXT
       Restore the original clipboard format
      crt.Clipboard.Format = format
    End Sub
Python
  Example
    #$language = "Python"
    #$interface = "1.0"
    crt.Screen.Synchronous = True
    def main():
      # Get the current clipboard format
      format = crt.Clipboard.Format
      # Set the clipboard format to VDS_TEXT:
      crt.Clipboard.Format =
      crt.Clipboard.VDS TEXT
```

#### **CF OEMTEXT**

#### Description

main()

Read-only property that returns the format string for CF\_0EMTEXT clipboard format. This text format contains characters in the OEM character set. Each line ends with a carriage return/linefeed (CR-LF) combination. A null character signals the end of the data.

# Restore the original clipboard format

crt.Clipboard.Format = format

#### Note

This format is only supported on Windows.

# **VBScript**

```
Syntax
  var = crt.clipboard.CF_OEMTEXT
Python
Syntax
  var = crt.clipboard.CF_OEMTEXT
```

#### **CF TEXT**

#### Description

Read-only property that returns the format string for CF\_TEXT clipboard format. In this text format, each line ends with a carriage return/linefeed (CR-LF) combination. A null character signals the end of the data. Use this format for ANSI text.

#### Note

This format is only supported on Windows.

# **VBScript**

**Syntax** 

var = crt.clipboard.CF\_TEXT

**Python** 

**Syntax** 

var = crt.clipboard.CF\_TEXT

#### **CF UNICODETEXT**

# Description

Read-only property that returns the format string for CF\_UNICODETEXT clipboard format. For Windows: In Unicode text format, each line ends with a carriage return/linefeed (CR-LF) combination. A null character signals the end of the data.

# **VBScript**

**Syntax** 

var = crt.clipboard.CF\_UNICODETEXT

**Python** 

**Syntax** 

var = crt.clipboard.CF\_UNICODETEXT

#### **DEFAULTFORMAT**

#### Description

Read-only Property that returns the format string for

DEFAULTFORMAT clipboard format. Setting the format to this property restores the value to the global default format for the clipboard.

```
VBScript
   Syntax
   var = crt.clipboard.DEFAULTFORMAT
Python
   Syntax
   var = crt.clipboard.DEFAULTFORMAT
```

#### **VDS TEXT**

#### Description

Read-only property that returns the format string for VDS\_TEXT clipboard format. This is a private clipboard format that translates certain special characters to ASCII when copying data from the screen to the clipboard. The characters that are translated are as follows:

- directional quotation marks (also known as "smart quotes") are changed to ASCII quotation marks
- · "em" dashes are changed to hyphens
- · "en" dashes are changed to hyphens

```
VBScript
   Syntax
   var = crt.clipboard.VDS_TEXT
Python
   Syntax
   var = crt.clipboard.VDS_TEXT
```

#### **Text**

#### **Description**

Returns or sets the contents of the clipboard.

```
VBScript
   Syntax
    crt.clipboard = varname
    [ varname = ] crt.clipboard
   Example
    # $language = "VBScript"
```

```
# $interface = "1.0"
    ' Put the selected text into the clipboard:
    crt.Clipboard.Text = crt.Screen.Selection
    ' Transfer text from the clipboard into a
    variable for
      use within the script:
    MyStr = crt.Clipboard.Text
    ' Send the contents of the clipboard to the
    remote
    ' machine:
    crt.Screen.Send crt.Clipboard.Text
Python
  Syntax
    crt.clipboard = varname
    [ varname = ] crt.clipboard
  Example
    #$language = "Python"
    \#$interface = "1.0"
    # Put the selected text into the clipboard:
    crt.Clipboard.Text = crt.Screen.Selection
    # Transfer text from the clipboard into a
    variable for
    # use within the script:
    MyStr = crt.Clipboard.Text
    # Send the contents of the clipboard to the
    remote
    # machine:
    crt.Screen.Send(crt.Clipboard.Text)
```

# **Description**

The Dialog object provides access to simple user-interface features provided by SecureCRT.

# **Syntax**

crt.Dialog.Method([arglist])

#### Remarks

SecureCRT's Dialog object is accessed through the top-level object's Dialog property.

# **Dialog Object Methods**

**Methods** 

**FileOpenDialog** 

MessageBox

**Prompt** 

## **Methods**

# **FileOpenDialog**

# Description

Display a file browse dialog from which the user can select a single file.

#### Remarks

If the defaultFilename parameter is simply a filename (no path provided), the file dialog browser will open in current working directory. If the defaultFilename parameter specifies an absolute path to a file, the file dialog browser will open in the parent directory of the file. The filename filter is in the following format:

```
<Name of Filter> (*.<extension>)|*.
<extension>||
```

```
For example:
```

```
Text Files (*.txt)|*.txt||
or
  Text Files (*.txt)|*.txt|Log File
  (*.log)|*.log||
```

The FileOpenDialog method returns the path to the selected file.

#### **VBScript**

#### **Syntax**

crt.Dialog.FileOpenDialog([title, [buttonLabel, [defaultFilename, [filter]]]])

## **Examples**

'Browse for a file of any type filePath = crt.Dialog.FileOpenDialog("Please select a file")

'Browse for only .txt files filePath = crt.Dialog.FileOpenDialog("Please select a text file", "Open", "", "Text Files (\*.txt)|\*.txt||")

#### **Python**

#### **Syntax**

crt.Dialog.FileOpenDialog([title, [buttonLabel, [defaultFilename, [filter]]]])

## **Examples**

# Browse for a file of any type

filePath = crt.Dialog.FileOpenDialog(title="Please select a file")

# Browse for only .txt files

filePath = crt.Dialog.FileOpenDialog(title="Please select a text file", filter="Text Files (\*.txt)|\*.txt||")

# **MessageBox**

# Description

Display a message

#### Remarks

The MessageBox function displays a message string to the user. The optional title string sets the title or caption of the

MessageBox. The buttons that appear on the MessageBox can be configured by passing a combination of numeric values in the optional 'buttons' parameter. By default MessageBox will display the message string with an **OK** button. However, many possibilities exist for displaying message boxes with different icons, and buttons.

The MessageBox function returns a numeric value that can be used to identify which button was clicked.

#### **VBScript**

#### **Syntax**

crt.Dialog.MessageBox(message [, title [, buttons]])

#### **Example**

The following code sample defines the constants that can be combined to form the "button" parameter as well as the possible numeric return values:

' button parameter options

Const ICON\_STOP = 16 ' display the ERROR/ST(
Const ICON\_QUESTION = 32 ' display the '?' icon
Const ICON\_WARN = 48 ' display a '!' icon.
Const ICON INFO= 64 ' displays "info" icon.

Const BUTTON\_OK = 0 'OK button only
Const BUTTON\_CANCEL = 1 'OK and Cancel butto
Const BUTTON\_ABORTRETRYIGNORE = 2 'Abort, Retry
Const BUTTON\_YESNOCANCEL = 3 'Yes, No, and Cancel buttons
Const BUTTON\_YESNO = 4 'Yes and No buttons
Const BUTTON\_RETRYCANCEL = 5 'Retry and Cancel

'OK button only
'OK and Cancel button
'OK and Cancel button
'Yes, No, and Cancel button

Const DEFBUTTON1 = 0 'First button is default
Const DEFBUTTON2 = 256 'Second button is default
Const DEFBUTTON3 = 512 'Third button is default

'Possible MessageBox() return values

Const IDOK = 1 'OK button clicked

Const IDCANCEL = 2 'Cancel button clicked

Const IDABORT = 3 'Abort button clicked

'Retry button clicked

'Ignore button clicked

Const IDYES = 6 'Yes button clicked

```
Const IDNO = 7 'No button clicked
```

```
' Display a messagebox with Yes/No buttons.
```

' Make the 'No' button the default.

result = crt.Dialog.MessageBox("Login Failed, Retry?", "Erro If result = IDNO Then

Exit Sub

End If

## **Python**

#### **Syntax**

crt.Dialog.MessageBox(message [, title [, buttons]])

### **Example**

The following code sample defines the constants that can be combined to form the "button" parameter as well as the possible numeric return values:

```
# button parameter options
                                # display the
ICON STOP = 16
ERROR/STOP icon.
ICON_QUESTION = 32
                                # display the
'?' icon
ICON WARN = 48
                                # display a
'!' icon.
                                # displays
ICON INFO= 64
"info" icon.
BUTTON OK = 0
                                # OK button
only
BUTTON CANCEL = 1
                                # OK and
Cancel buttons
```

BUTTON\_ABORTRETRYIGNORE = 2 # Abort,

Retry, and Ignore buttons

BUTTON\_YESNOCANCEL = 3 # Yes, No,

and Cancel buttons

 $BUTTON\_YESNO = 4$  # Yes and No

buttons

BUTTON\_RETRYCANCEL = 5 # Retry and

Cancel buttons

```
DEFBUTTON1 = 0
                      # First button is
default
                      # Second button is
DEFBUTTON2 = 256
default
DEFBUTTON3 = 512
                      # Third button is
default
# Possible MessageBox() return values
                      # OK button clicked
IDOK = 1
IDCANCEL = 2
                      # Cancel button
clicked
IDABORT = 3
                      # Abort button clicked
                      # Retry button clicked
IDRETRY = 4
IDIGNORE = 5
                      # Ignore button
clicked
IDYES = 6
                      # Yes button clicked
IDNO = 7
                      # No button clicked
# Display a messagebox with Yes/No buttons.
# Make the 'No' button the default.
result = crt.Dialog.MessageBox("Login
Failed, Retry?", "Error", ICON_QUESTION |
BUTTON YESNO | DEFBUTTON2 )
if result == IDNO:
  return
```

## **Prompt**

### Description

Prompt a user to enter a string.

#### Remarks

The Prompt function displays a simple dialog that has message and an edit field for the user to enter a string. The message parameter is an informational string displayed in the prompt dialog. Optionally the title of the prompt dialog may be set by passing a title string. By default the edit field is empty, but the initial contents of the edit field may be set with the optional default string. Finally, if the text entered in the edit field is to be obscured as it is entered (such as when entering a password)

```
then the Boolean isPassword field should be set to True.
  If the user clicks OK, Prompt returns the entered string; whereas,
  if the user clicks Cancel, Prompt returns an empty string.
VBScript
  Syntax
     crt.Dialog.Prompt(message [, title [, default [, isPassword ]]])
  Example
     Dim pass
       pass = crt.Dialog.Prompt("Enter your password:", "Logon
     Script", "", True)
       If pass = "" Then
         'User clicked Cancel button
        Else
         'User added data
        End If
Python
  Syntax
     crt.Dialog.Prompt(message [, title [, default [, isPassword ]]])
  Example
     password = crt.Dialog.Prompt("Enter your password:", "Logon
     Script", "", True)
     if password == "":
      # User clicked Cancel button
     else:
      # User added data
```



# FileTransfer Object

## **Description**

The FileTransfer object provides methods for performing file transfers initiated by scripts.

## **Syntax**

crt.FileTransfer.Method([arglist])
crt.FileTransfer.Property [ = varname ]

#### Remarks

SecureCRT's FileTransfer object is accessed through the top-level object's FileTransfer property.

## **FileTransfer Object Properties and Methods**

<u>Properties</u> <u>Methods</u>

<u>DownloadFolder</u> <u>AddToZmodemUploadList</u>

ZmodemUploadAscii SendXmodem

ReceiveXmodem

## **Properties**

#### **DownloadFolder**

### Description

Returns path for session download folder.

#### Remarks

DownloadFolder is a read-only property that returns the path for the current session's download folder. Note, files downloaded with either Xmodem or Zmodem are always placed in the session's download folder. Scripts can use this property to determine the location of the downloaded files.

## **VBScript**

### **Syntax**

crt.FileTransfer.DownloadFolder

#### Example

MsgBox "Download completed to: " & vblf & crt.FileTransfer.DownloadFolder

### **Python**

### **Syntax**

crt.FileTransfer.DownloadFolder

### **Example**

crt.Dialog.MessageBox("Download completed to: \n" +
crt.FileTransfer.DownloadFolder)

### **ZmodemUploadAscii**

### **Description**

Specifies ASCII or binary mode for **Zmodem** uploads.

#### Remarks

ZmodemUploadAscii specifies whether files uploaded using Zmodem will be in ASCII or binary format. When ZmodemUploadAscii is set to "true", uploads will be made in ASCII. After you have run a script that sets the ZmodemUploadAscii option to true, that setting will be saved as a session option until it is changed back either by running another script that resets it to false or by changing the option in the Select Files to Send using Zmodem dialog. If no argument is specified for this property, the current value is returned.

#### Note

For this option to work properly, the installation of Zmodem on the remote machine must be able to convert ASCII to use the local convention for line termination.

### **VBScript**

## **Syntax**

crt.FileTransfer.ZmodemUploadAscii = [True | False]

## **Example**

```
#$language = "VBScript"
#$interface = "1.0"
```

Option explicit

```
Dim szPrompt
szPrompt = "->"
```

crt.screen.synchronous = True

```
' Zmodem ASCII uploads
     crt.FileTransfer.AddToZmodemUploadList "c:\temp\File1.txt"
     crt.FileTransfer.AddToZmodemUploadList "c:\temp\File2.txt"
     crt.FileTransfer.AddToZmodemUploadList
     "c:\temp\anotherFile.txt"
     Let CRT know that we want these files to be transferred as
     ASCII
     crt.FileTransfer.ZmodemUploadAscii = True
     ' Start the Zmodem upload
     crt.Screen.Send "_rz " & vbCR
     ' Wait for transfer to complete
     crt.Screen.WaitForString szPrompt
     ' Reset ASCII flag
     crt.FileTransfer.ZmodemUploadAscii = False
Python
  Syntax
     crt.FileTransfer.ZmodemUploadAscii = [True | False]
  Example
     #$language = "Python"
     #$interface = "1.0"
     crt.Screen.Synchronous = True
     # Zmodem ASCII uploads
     crt.FileTransfer.AddToZmodemUploadList("c:\\temp\\File1.txt")
     crt.FileTransfer.AddToZmodemUploadList("c:\\temp\\File2.txt")
     crt.FileTransfer.AddToZmodemUploadList("c:\\temp\\anotherFile
     # Let CRT know that we want these files to be transferred as
     ASCII
     crt.FileTransfer.ZmodemUploadAscii = True
     # Start the Zmodem upload
```

```
crt.Screen.Send("rz\n")

# Wait for transfer to complete
crt.Screen.WaitForString("->")

# Reset ASCII flag
crt.FileTransfer.ZmodemUploadAscii = False
```

#### **Methods**

### **AddToZmodemUploadList**

#### **Description**

Places file on Zmodem upload list.

#### Remarks

AddToZmodemUploadList places the specified file on a list of files that will be uploaded during the next Zmodem upload. Once one or more files have been added to the upload list, a Zmodem upload can be initiated by the script sending the appropriate command to the remote system.

#### Errors:

If the path provided to AddToZmodemUploadList is not a valid file, a script error is generated and the following message is displayed:

"FileTransfer.AddToZmodemUploadList: <filepath> does not exist."

## **VBScript**

## **Syntax**

crt.FileTransfer.AddToZmodemUploadList filepath

## Example

```
crt.FileTransfer.AddToZmodemUploadList
"c:\temp\File1.txt"
crt.FileTransfer.AddToZmodemUploadList
"c:\temp\File2.txt"
```

```
' Start the upload of the two files... crt.Screen.Send "rz" & vbCR
```

### **Python**

### **Syntax**

crt.FileTransfer.AddToZmodemUploadList(filepath)

#### **Example**

```
crt.FileTransfer.AddToZmodemUploadList("c:\\tel
crt.FileTransfer.AddToZmodemUploadList("c:\\tel
# Start the upload of the two files...
crt.Screen.Send("rz\n")
```

#### SendXmodem

## Description

Sends specified file.

#### Remarks

SendXmodem sends the specified file using the Xmodem protocol. The full path to the file must be specified. Note, the appropriate Xmodem receive command should be sent or initiated on the remote system before executing SendXmodem for the transfer to begin properly.

#### Errors:

1. If the SendXmodem method is executed while not connected the following script error is generated:

"FileTransfer.SendXmodem: not connected"

2. If the path provided to SendXmodem is not a valid file then a script error is generated and the following message is displayed:

"FileTransfer.SendXmodem: <filepath> does not exist."

### **VBScript**

## **Syntax**

crt.FileTransfer.SendXmodem filepath

### **Example**

' upload C:\temp\myFile.txt using Xmodem crt.Screen.Send "rx myFile.txt" & vbCR crt.Screen.WaitForString "ready to receive" crt.FileTransfer.SendXmodem "C:\temp\myFile.txt"

#### **Python**

### **Syntax**

crt.FileTransfer.SendXmodem(filepath)

### **Example**

# upload C:\temp\myFile.txt using Xmodem crt.Screen.Send("rx myFile.txt\n") crt.Screen.WaitForString("ready to receive") crt.FileTransfer.SendXmodem("C:\\temp\\myFile.txt")

#### ReceiveXmodem

### Description

Initiates file download to download folder.

#### Remarks

ReceiveXmodem initiates an Xmodem download of a file and saves the file as the specified filename in the session's download folder. Downloaded files are always placed in the session's download folder. Note that the Xmodem send command should be sent or initiated on the remote system before executing ReceiveXmodem for the transfer to proceed properly.

#### Errors:

1. If the ReceiveXmodem method is executed while not connected, the following script error is generated:

"FileTransfer.ReceiveXmodem: not connected"

2. If a filename passed to ReceiveXmodem is not a simple filename (i.e., if it is a path or contains path separators), the following script error will be generated:

"FileTransfer.ReceiveXmodem: Invalid filename \"% s\". Argument should not include path information."

## **VBScript**

### **Syntax**

crt.FileTransfer.ReceiveXmodem filepath

### **Example**

crt.Screen.Send "sx -X myFile.txt" & vbCR

' Customize the wait string below to match output of remote Xmodem program.

crt.Screen.WaitForString "Give your local XMODEM receive command now." & vbCR

crt.FileTransfer.ReceiveXmodem "yourFile.txt"

**Note:** When using a WaitForString command as in the above example, the string being waiting for should be the exact Xmodem output, including carriage returns and line feeds.

### **Python**

### **Syntax**

crt.FileTransfer.ReceiveXmodem(filepath)

#### **Example**

crt.Screen.Send("sx -X myFile.txt\n")

# Customize the wait string below to match output of remote Xmodem program.

crt.Screen.WaitForString("Give your local XMODEM receive command now.")

crt.FileTransfer.ReceiveXmodem("yourFile.txt")

**Note:** When using a WaitForString command as in the above example, the string being waiting for should be the exact Xmodem output, including carriage returns and line feeds.

The Screen object provides access to SecureCRT's terminal screen.

# **Syntax**

Screen.Property [ = expression ] Screen.Method([arglist])

### **Remarks**

SecureCRT's Screen object is accessed through the top-level object's Screen property.

# **Screen Object Properties and Methods**

<u>Properties</u>	<u>Methods</u>
CurrentColumn	<u>Clear</u>
CurrentRow	<u>Get</u>
Columns	Get2
<u>IgnoreEscape</u>	<u>IgnoreCase</u>
MatchIndex	<u>Print</u>
Rows	ReadString
Selection	Send
Synchronous	<u>SendKeys</u>
	<u>SendSpecial</u>
	WaitForCursor
	<u>WaitForKey</u>
	WaitForString
	<u>WaitForStrings</u>

# **Properties**

### **CurrentColumn**

Returns the current column of the cursor.

#### Remarks

Read-only numeric property. The first column is 1. An error will be returned if there is no connection open.

### **VBScript**

### **Syntax**

[ varname = ] object.CurrentColumn

### **Python**

### **Syntax**

[ varname = ] object.CurrentColumn

#### **CurrentRow**

### **Description**

Returns the current row of the cursor.

#### Remarks

Read-only numeric property. The first row is 1. An error will be returned if there is no connection open.

## **VBScript**

## **Syntax**

[ varname = ] object.CurrentRow

## **Python**

## **Syntax**

[ varname = ] object.CurrentRow

#### **Columns**

### Description

Returns the current number of columns.

#### Remarks

Read-only numeric property.

## **VBScript**

## **Syntax**

[ varname = ] object.Columns

## Python

### **Syntax**

[ varname = ] object.Columns

## **IgnoreEscape**

Allows the script to control whether escape sequences are ignored or not for WaitForString(s) and ReadString methods.

#### Remarks

The IgnoreEscape property toggles whether ReadString() or WaitForString() recognizes escape sequences. By default, Readstring will capture and look for escape sequences.

#### **VBScript**

### **Syntax**

object.IgnoreEscape [ = True | False ]

### **Example**

crt.screen.IgnoreEscape = False

' The following call will return after we receive either an

' 'Escape' or 'BEL' character from the remote MsgBox crt.screen.Readstring(chr(27), chr(7)) crt.screen.IgnoreEscape = True

' Now that IgnoreEscape is set to true, the following call

' will always time out after 5 seconds because ReadString is

' ignoring non-printable characters
MsgBox crt.screen.Readstring(chr(27), chr(7),
5)

# Python

# **Syntax**

object.IgnoreEscape [ = True | False ]

### **Example**

crt.Screen.IgnoreEscape = False

# The following call will return after we receive either an

# 'Escape' or 'BEL' character from the remote
crt.Dialog.MessageBox(crt.Screen.ReadString([c|
chr(7)]))

crt.Screen.IgnoreEscape = True

# Now that IgnoreEscape is set to true, the following call

# will always time out after 5 seconds because
ReadString is

```
# ignoring non-printable characters
crt.Dialog.MessageBox(crt.Screen.ReadString([c|
chr(7)], 5))
```

#### **MatchIndex**

#### **Description**

Determines which index within your list of strings was found by the ReadString or WaitForStrings method.

#### Remarks

Is only useful in combination with either the ReadString or WaitForStrings method.

The crt.Screen.MatchIndex property is used in conjunction with the ReadString method when ReadString is passed multiple strings to wait for. When used, MatchIndex will indicate which string was found by ReadString (1=first\_string, 2=second\_string, etc.). A MatchIndex value of 0 indicates that a timeout occurred before a match was found.

## **VBScript**

```
Syntax
```

```
[ varname = ] object.MatchIndex
```

### **Example**

```
szOutput = crt.Screen.ReadString("error",
"warning", "#", 10)
Select Case crt.Screen.MatchIndex
Case 0
MsgBox "Timed out!"
Case 1
MsgBox "Found 'error'"
Case 2
MsgBox "Found 'warning'"
Case 3
MsgBox "Found '#'"
End Select
```

### Python

### **Syntax**

```
[ varname = ] object.MatchIndex
```

## **Example**

```
szOutput = crt.Screen.ReadString(["error",
       "warning", "#"], 10)
       index = crt.Screen.MatchIndex
       if (index == 0):
         crt.Dialog.MessageBox("Timed out!")
       elif (index == 1):
         crt.Dialog.MessageBox("Found 'error'")
       elif (index == 2):
         crt.Dialog.MessageBox("Found 'warning'")
       elif (index == 3):
         crt.Dialog.MessageBox("Found '#'")
Rows
  Description
    Returns the current number of rows.
  Remarks
     Read-only numeric property.
  VBScript
    Syntax
       [ varname = ] object.Rows
  Python
    Syntax
       [ varname = ] object.Rows
Selection
  Description
    Returns the current selection.
  VBScript
    Syntax
       [ varname = ] object. Selection
     Example
       # $language = "VBScript"
       # $interface = "1.0"
       ' GoogleSelectedText.vbs
       ' send the selected text to the clipboard
       crt.Clipboard.Text = crt.Screen.Selection
       'Extract the selected text from the clipboard into a variable as
       "Text"
```

```
szSelection = crt.Clipboard.Text
     ' Now search on Google for the information.
     g szSearchBase = "http://www.google.com/search?hl=en&g="
     Set g shell = CreateObject("WScript.Shell")
     g shell.Run "iexplore " & g szSearchBase & szSelection
Python
  Syntax
     [ varname = ] object. Selection
  Example
     \#$language = "Python"
     #$interface = "1.0"
     import SecureCRT
     import webbrowser
     # send the selected text to the clipboard
     crt.Clipboard.Text = crt.Screen.Selection
     # Extract the selected text from the clipboard into a variable as
     "Text"
     szSelection = crt.Clipboard.Text
     # Now search on Google for the information.
     g szURL = "http://www.google.com/search?q=" + szSelection
     webbrowser.open(g szURL)
```

### **Synchronous**

## Description

Returns or sets the Synchronous setting of the screen.

#### Remarks

If Synchronous is False then under certain circumstances a script can miss data sent by the  $\underline{\text{server}}$  that it is expecting to see.

Synchronous is set to False by default.

# **VBScript**

### **Syntax**

object.Synchronous [ = True | False ]

## **Example**

The following code that waits for two different strings, could potentially miss the second string while it is performing some operation after receiving the first string. In order to prevent this kind of condition, it temporarily sets Synchronous to True:

```
crt.screen.Synchronous = True
       crt.screen.Send("someCommand")
       crt.screen.WaitForString("thisString")
       ... do something else
       crt.screen.WaitForString("thatString")
Python
  Syntax
    object.Synchronous [ = True | False ]
  Example
    The following code that waits for two different strings, could
    potentially miss the second string while it is performing some
    operation after receiving the first string. In order to prevent this
    kind of condition, it temporarily sets Synchronous to True:
       crt.Screen.Synchronous = True
       crt.Screen.Send("someCommand")
       crt.Screen.WaitForString("thisString")
       ... do something else
       crt.Screen.WaitForString("thatString")
```

#### **Methods**

```
Clear
Description
Clears the screen.
Remarks
None.
VBScript
Syntax
object.Clear
Python
Syntax
object.Clear()

Get
Description
```

Returns a string of characters read for a portion of the screen.

#### Remarks

Returns a string containing the characters on the screen rectangle defined by the numeric values row1,col1 (upper-left) and row2,col2 (lower-right).

#### **VBScript**

## **Syntax**

object.Get(row1, col1, row2, col2)

### Python

### **Syntax**

object.Get(row1, col1, row2, col2)

#### Get2

#### **Description**

Returns the characters on each row requested.

#### Remarks

Returns the characters on each row requested with a \r\n, so the rows can be split by looking for the \r\n sequence. This allows the rows to be different lengths as required by the contents of the rows.

#### **Notes**

If your scripts need to work with MBCS languages, you should use the Get2 interface.

## **VBScript**

## **Syntax**

object.Get2(row1, col1, row2, col2)

## Python

### **Syntax**

object.Get2(row1, col1, row2, col2)

## <u>IgnoreCase</u>

## Description

Provides a global method to set case insensitivity. In addition, case insensitivity can be set per-function as described below in the WaitForStrings, WaitForString, and ReadString methods.

#### Remarks

If this method is true, the WaitForStrings, WaitForString, and ReadString methods are not case sensitive, otherwise, they are

```
case sensitive (default).

VBScript
Syntax
object.IgnoreCase

Python
Syntax
object.IgnoreCase()
```

#### **Print**

### **Description**

Prints the screen.

#### Remarks

If no printer is defined on your machine, an error will be returned.

**VBScript** 

**Syntax** 

object.Print

**Python** 

**Syntax** 

object.Print()

### **ReadString**

### **Description**

Captures data as it is received from the remote.

#### Remarks

ReadString is similar to the WaitForStrings function except in that ReadString captures data. By default, ReadString will capture all data received from the remote, including escape sequences. To enable or disable the inclusion of escape sequences in the data captured by ReadString, set the Screen.IgnoreEscape property to false/true, respectively. If the remote side is sending escape sequences and Screen.IgnoreEscape is set to true, ReadString will return the string "plug" when "p" was drawn in the upper left corner, "I" in the upper right corner, "u" in the bottom left corner, and "g" in the bottom right corner.

If a timeout parameter is provided, and ReadString reaches the timeout period without receiving the specified string(s) from the remote, ReadString will return an empty string.

#### **Notes**

This method takes an optional bCaseInsensitive parameter that defaults to false (case-sensitive wait string matching). It also takes an optional lWaitTimeout parameter, which defaults to "0" if not specified. To specify the bCaseInsensitive parameter, you must also explicitly specify the lWaitTimeout parameter when using this function in a script.

To retrieve a 1-based index of which string ReadString encountered, check the Screen.MatchIndex property.

### **VBScript**

### **Syntax**

[ varname = ] object.ReadString([string1[,string2 ... , stringn]] [, timeoutSeconds] [bCaseInsensitive])

### **Example**

ReadString has the following three usage scenarios:

1. Returns data as soon as it is available from the remote, one character at a time. The syntax for this scenario is:

```
varname = crt.Screen.ReadString
Example:
   char = crt.Screen.ReadString
```

2. Capture data until a specific string is detected from the remote (similar to how WaitForStrings() is used, except that it captures data). The syntax for this scenario is:

```
varname =
crt.Screen.ReadString(StringToWaitFor [,
TimeOutSeconds])
```

## Example:

```
str = crt.Screen.ReadString("home", 10)
```

 Capture data until one of a list of multiple strings is detected from the remote (similar to how WaitForStrings is used, except that it captures data). The syntax for this scenario is:

```
varname =
crt.Screen.ReadString(StringToWaitFor [,
StringToWaitFor [, ...]] [,
```

```
TimeOutSeconds])
       or
         varname =
         crt.Screen.ReadString(StringsArray [,
         TimeOutSeconds])
       Examples:
         str = crt.Screen.ReadString("home",
         "work", ..., 10)
       or
         str =
         crt.Screen.ReadString(arrayOfStrings, 10)
Python
  Syntax
    [ varname = ] object.ReadString([stringarray] [,
    timeoutSeconds] [, bcaseInsensitive])
  Example
    ReadString has the following three usage scenarios:
    1. Returns data as soon as it is available from the remote.
       one character at a time. The syntax for this scenario is:
         varname = crt.Screen.ReadString()
       Example:
         char = crt.Screen.ReadString()
    2. Capture data until a specific string is detected from the
       remote (similar to how WaitForStrings() is used, except that
       it captures data). The syntax for this scenario is:
         varname =
         crt.Screen.ReadString(StringToWaitFor [,
         TimeOutSeconds])
       Example:
         str = crt.Screen.ReadString("home", 10)
    3. Capture data until one of a list of multiple strings is
```

detected from the remote (similar to how WaitForStrings is

used, except that it captures data). The syntax for this

scenario is:

```
varname =
  crt.Screen.ReadString(StringsArray [,
  TimeOutSeconds])

Example:
  str = crt.Screen.ReadString(["home",
  "work"], 10)
```

#### Send

#### **Description**

Sends a string of characters.

#### Remarks

Attempting to send a string while no connection is open returns an error.

#### **Notes**

The Send interface works with MBCS languages, and works correctly regardless of whether the display font can represent the characters or not, as long as the select "Character Encoding" for the session can represent the characters.

```
VBScript
Syntax
object.Send string
Python
Syntax
object.Send(string)
```

## **SendKeys**

### **Description**

Sends key strokes to the active window.

#### Remarks

The SendKeys method can send more than one keystroke at a time by using compound string arguments. For example, to send the keystrokes a, b, and c, you would send the string argument "abc". The SendKeys method also uses some characters as modifiers of characters. This set of special characters consists of the plus sign (+), caret (^), percent sign (%), tilde (~), parentheses, brackets, and braces. The characters "+", "^", and "%" perform the functions of SHIFT, CTRL, and ALT, respectively.

These can be combined to affect one key as in "^%c" which is the equivalent of the CTRL+ALT+C key combination. Parenthesis characters are used to group characters for modifiers, for example, "+(ec)" will send "EC". SendKeys can use up to three nested parenthesis.

To send bracket characters, send the string argument "{[]" for the left bracket and "{[]]" for the right one. To send brace characters, send the string argument "{{}" for the left brace and "{}}" for the right one.

Some keystrokes do not generate characters (such as ENTER and TAB). Some keystrokes represent actions (such as BACKSPACE and BREAK). To send these kinds of keystrokes, send the arguments shown in the <u>SendKeys keystroke table</u>.

```
VBScript
  Syntax
    object. SendKeys string
  Example
    #$language = "VBScript"
    #$interface = "1.0"
    Sub Main()
    crt.Screen.Clear
        crt.screen.sendkeys("mc~")
        crt.Sleep 2000
        crt.screen.sendkeys("{f1}")
        crt.Sleep 2000
        crt.screen.sendkeys("{esc}0")
        crt.Sleep 2000
        crt.screen.sendkeys("{esc}0")
        crt.Sleep 2000
        crt.screen.sendkeys("y")
    End Sub
Python
  Syntax
    SendKeys is not currently supported for Python.
```

## **SendSpecial**

Sends a built-in SecureCRT command. SendSpecial can send any of the Menu, Telnet, and VT functions listed on the Map Selected Key dialog (accessed by selecting a key in the Keymap Editor and clicking on the Map Selected Key... button).

#### Remarks

The string parameter to SendSpecial should describe one of the special SecureCRT or protocol functions. Attempting to use SendSpecial while no connection is opened will result in an error.

### **VBScript**

#### **Syntax**

object.SendSpecial string

### **Example**

```
screen.SendSpecial "MENU_PASTE" screen.SendSpecial "TN_BREAK" screen.SendSpecial "VT PF1"
```

#### **Python**

#### **Syntax**

object.SendSpecial(string)

### **Example**

```
crt.Screen.SendSpecial("MENU_PASTE")
crt.Screen.SendSpecial("TN_BREAK")
crt.Screen.SendSpecial("VT_PF1")
```

#### **WaitForCursor**

### Description

Wait for the cursor to change position.

#### Remarks

The optional timeout parameter specifies the number of seconds to wait for the change. If a change of cursor position is detected WaitForCursor() returns True. If a timeout occurs the function returns False. If no timeout is specified then WaitForCursor() will not time out. An error will be returned if there is no connection open.

## **VBScript**

### **Syntax**

```
[ result = ] object.WaitForCursor [ timeout ]
```

## **Python**

## **Syntax**

## [ result = ] object.WaitForCursor([timeout])

### **WaitForKey**

### Description

Wait for a keypress event.

#### Remarks

The optional timeout parameter specifies the number of seconds to wait for a key event. If key event is detected WaitForKey() returns True. If a timeout occurs the function returns False. If no timeout is specified then WaitForKey() will not time out. An error will be returned if there is no connection open.

### **VBScript**

### **Syntax**

[ result = ] object.WaitForKey [ timeout ]

### **Python**

### **Syntax**

[ result = ] object.WaitForKey([timeout])

### **WaitForString**

## **Description**

Wait for a string.

#### Remarks

Wait for the string to appear in the input. The timeout (seconds) parameter is optional. When the string is detected in the input WaitForString() returns True. If a timeout occurs the function returns False. An error will be returned if there is no connection open.

#### Note

This method takes an optional bCaseInsensitive parameter that defaults to false (case-sensitive wait string matching). It also takes an optional IWaitTimeout parameter, which defaults to "0" if not specified. To specify the bCaseInsensitive parameter, you must also explicitly specify the IWaitTimeout parameter when using this function in a script.

## **VBScript**

## **Syntax**

[ result = ] object.WaitForString string [, timeout] [bCaseInsensitive]

```
Example
    If crt.screen.WaitForString("ogin:", 10) <> Tri
        MsgBox "Failed to detect login!"
        Exit Sub
    End If

Python
    Syntax
    [result = ] object.WaitForString(string [, timeout] [,
        bCaseInsensitive])

Example
    if (crt.Screen.WaitForString("ogin:", 10) !=
        True):
        crt.Dialog.MessageBox("Failed to detect
        login!")
```

### **WaitForStrings**

### Description

Wait for one of several strings to appear in the input.

#### Remarks

Waits for one of the strings given as arguments to appear in the input. When one of the argument strings is matched in the input, WaitForStrings() returns the argument index of the string that was found (the index of the first string given as an argument to WaitForStrings() is 1). If the optional timeout parameter is specified and a timeout occurs before any of the strings are found, WaitForStrings() returns 0. In the absence of a timeout parameter WaitForStrings() will block without timing out and will not return 0. An error will be returned if there is no connection open.

#### **Notes**

If you are using VBScript, WaitForStrings() will accept an array of strings as its first argument followed by an optional timeout. The value returned by WaitForStrings() will be the index of the string found in the array (1st element in the array = 1). A value of 0 will be returned if no strings were found within the timeout period if specified.

The WaitForString(s) interface works with MBCS languages, and

depends only on the "Character Encoding" for the session being able to represent the characters being waited for, not on the characters being displayed correctly on the screen.

This method takes an optional bCaseInsensitive parameter that defaults to false (case-sensitive wait string matching). It also takes an optional IWaitTimeout parameter, which defaults to "0" if not specified. To specify the bCaseInsensitive parameter, you must also explicitly specify the IWaitTimeout parameter when using this function in a script.

```
VBScript
```

```
Syntax
```

```
[ result = ] object.WaitForStrings string1, [string2, ..., stringn] [, timeout] [bCaseInsensitive]
```

### **Example**

# Python

### **Syntax**

[ varname = ] object.WaitForStrings(StringsArray [, timeout] [, bcaseInsensitive])

### **Example**

```
result = crt.Screen.WaitForStrings(["foo",
"bar", "quux", "gee"], 10)
crt.Dialog.MessageBox(str(result))
if (result == 3):
    crt.Dialog.MessageBox("Got quux!")
if (result == 0):
    crt.Dialog.MessageBox("Timed out!")
```



The Session object provides access to the state and properties that exist for the current connection or session .

## **Syntax**

```
session.Property [ = expression ] session.Method([arglist])
```

#### Remarks

SecureCRT's Session object is accessed through the top-level object's Session property.

# **Session Object Properties and Methods**

PropertiesMethodsConfigConnectConnectedConnectInTabLocalAddressDisconnect

<u>Log</u> <u>Log</u>

<u>LogUsingSessionOptions</u>

<u>Path</u> <u>Print</u>

RemoteAddress SetStatusText

RemotePort

## **Properties**

## **Config**

### **Description**

Returns the configuration associated with the session.

#### Remarks

This configuration will not be saveable to a new name.

## **VBScript**

```
Syntax
Set objectvarname = object.Config
Example
Set config = crt.Session.Config
Python
Syntax
objectvarname = object.Config
Example
config = crt.Session.Config
```

#### **Connected**

### **Description**

Returns a Boolean value indicating whether the current session is connected or not.

#### Remarks

Boolean read-only property.

### **VBScript**

**Syntax** 

[ varname = ] object. Connected

### **Python**

**Syntax** 

[ varname = ] object.Connected

#### LocalAddress

### Description

Returns the IP address of the local machine in the form of a string.

#### Remarks

LocalAddress is a read-only string property. The LocalAddress property should only be accessed if the session is connected. Attempting to access LocalAddress while not connected is an error.

## **VBScript**

### **Syntax**

[ varname = ] object.LocalAddress

## Python

## **Syntax**

[ varname = ] object.LocalAddress

### **LogFileName**

### **Description**

Returns or sets the name of the current log file.

#### Remarks

If filename is invalid a runtime error is generated. See also: session.Log

If the log filename contains parameter substitutions that are known when the function is called, they are filled in. Otherwise, the literal string is returned.

## **VBScript**

#### **Syntax**

[ varname = ] object.LogFileName [ = filename ]

## **Python**

#### **Syntax**

[ varname = ] object.LogFileName [ = filename ]

### **Logging**

## Description

Returns a Boolean value indicating whether the current session is logging or not.

#### Remarks

Boolean read-only property.

## **VBScript**

## **Syntax**

[ varname = ] object.Logging

### **Python**

## **Syntax**

[ varname = ] object.Logging

#### **Path**

## Description

Returns the path for the current session starting from location of the session INI files.

#### Remarks

Path is a read-only string property that returns the path for the current session starting from location of the session INI files. If no session has been selected, this function will return the name "Default".

```
VBScript
     Syntax
       [ varname = ] object.Path
     Example
       If crt.session.path = "work\server0" Then
          myprompt = "linux"
        End If
  Python
     Syntax
       [ varname = ] object.Path
     Example
       if crt.Session.Path == "work\server0":
         myprompt = "linux"
RemoteAddress
  Description
     Returns the IP address of the remote host in the form of a string.
  Remarks
     RemoteAddress is a read-only string property. The
     RemoteAddress property should only be accessed if the session
     is connected. Attempting to access RemoteAddress while not
     connected is an error.
  VBScript
     Syntax
       [ varname = ] object.RemoteAddress
  Python
     Syntax
        [ varname = ] object.RemoteAddress
RemotePort
  Description
     Returns the port number of the remote port.
  VBScript
     Syntax
       [ varname = ] object.RemotePort
  Python
     Syntax
```

[ varname = ] object.RemotePort

#### **Methods**

#### Connect

### **Description**

Connects to a session.

#### Remarks

The Connect method takes a string parameter that specifies how a connection is to be made. The format of the string parameter matches the format of the command line arguments to SecureCRT. The Connect method can accept an empty argument set.

The command crt.Session.Connect() means "connect to the current session". If a connection has not been made before, the command will fail. If a connection already exists, nothing happens, otherwise, this command is equivalent to pressing the **Reconnect** button.

#### **Examples:**

Connect using a predefined session: crt.session.Connect("/s mysession")

Connect to "myhost" on port 2345 using the  $\underline{\text{Telnet}}$  protocol and Default session parameters:

crt.session.Connect("/telnet myhost 2345")

The Connect method takes a parameter (True or False) that determines whether or not the script should wait for the connection to fully authenticate before continuing. For instance, the connect call in the examples below will wait until the connection has been fully authenticated before returning and allowing the script to execute the error check just below the connect call. This parameter only applies to SSH1 and SSH2 connections. Whereas, passing False would allow the error check to execute before the connection has a chance to finish. You will probably want to use False only in cases where you want to script the logon process for the session you are attempting to

connect to. The default value for this parameter is True.

The Connect method also takes an optional parameter (True or False) that specifies whether or not to suppress pop-up messages. This parameter defaults to False (don't suppress).

```
VBScript
  Syntax
    object.Connect [arg [, True|False] [, True|False]]
  Examples
    # $language = "VBScript"
    # $interface = "1.0"
    On Error Resume Next
    crt.Session.Connect "/SSH2 /PASSWORD password
    username@hostname", True
    If Err.Number <> 0 Then
        MsgBox "Connection Failed"
    Else
        MsgBox "Connection Successful"
    End If
    The following is an example script that uses the
    /ENCRYPTEDPASSWORD command-line option.
    # $language = "VBScript"
    # $interface = "1.0"
    set config = crt.OpenSessionConfiguration("
    <session name>")
    CRT.dialog.messagebox(config.GetOption("Passwo
    crt.Session.Connect "/SSH2 /ENCRYPTEDPASSWORD
    " & config.GetOption("Password") & "
    <username>@<hostname>", True
Python
  Syntax
    object.Connect(arg [, True|False] [, True|False])
  Examples
    # $language = "Python"
    # $interface = "1.0"
```

```
errcode = 0
try:
  crt.Session.Connect("/SSH2 /PASSWORD
password username@hostname", True)
except ScriptError:
  errcode = crt.GetLastError()
if errcode != 0:
  crt.Dialog.MessageBox("Connection Failed")
else:
  crt.Dialog.MessageBox("Connection
Successful")
The following is an example script that uses the
/ENCRYPTEDPASSWORD command-line option.
# $language = "Python"
# $interface = "1.0"
import SecureCRT
set config = crt.OpenSessionConfiguration("
<session name>")
crt.Dialog.Messagebox(config.GetOption("Passwo
crt.Session.Connect("/SSH2 /ENCRYPTEDPASSWORD
" + config.GetOption("Password") + "
<username>@<hostname>", True)
```

#### **ConnectInTab**

## **Description**

Connects to a session in a tab or tiled session window.

#### Remarks

Takes the same arguments as Connect. This also returns a Tab object.

The ConnectInTab method also takes two optional parameters (True or False). The first determines whether or not the script should wait for the connection to fully authenticate before continuing. The first parameter only applies to SSH1 and SSH2 connections. The second optional parameter specifies whether the call fails silently or raises an exception. If the second

parameter is False and the connection attempt fails, an exception is raised and the method returns an empty object. If the second parameter is True and the connection attempt fails, the method returns a valid Tab object, which can then be used to check for connection status in order to determine if the connection attempt was successful. See the Connect method above for additional examples.

## **VBScript**

### **Syntax**

Set objTab = object.ConnectInTab [arg [, True|False] [, True|False]]

## **Examples**

Connect in a tab using a predefined session:
 Set tab = crt.session.ConnectInTab("/s
 mysession")

Connect in a tab to "myhost" on port 2345 using the <u>Telnet</u> protocol and Default session parameters:

Set tab = crt.session.ConnectInTab("/telnet
myhost 2345")

Close tab if attempt to connect fails:

```
Set objNewTab =
crt.Session.ConnectInTab("Host_Does_Not_Exis")
MsqBox
    "Script Tab's index: " &
crt.GetScriptTab().Index & vbcrlf & _
    "New Tab's index: " & objNewTab.Index
    If Not objNewTab.Session.Connected Then
         ' Make sure we're not re-using
disconnected
         ' tabs, we don't want to close the
script
         ' tab, just any new tabs that get
created by
           calls to ConnectInTab(), but
which don't
         ' result in a successful
```

```
connection.
                If crt.GetScriptTab().Index <>
      objNewTab.Index Then
                     crt.Dialog.MessageBox("Closing
      failed tab")
                     objNewTab.Close()
                End If
            End If
Python
  Syntax
    objTab = object.ConnectInTab [arg [, True|False] [, True|False]]
  Example
    Connect in a tab using a predefined session:
      crt.Session.ConnectInTab("/s mysession")
    Connect in a tab to "myhost" on port 2345 using the Telnet
    protocol and Default session parameters:
      crt.Session.ConnectInTab("/telnet myhost
      2345")
    Close tab if attempt to connect fails:
      obiNewTab =
      crt.Session.ConnectInTab("Host_Does_Not_Exis"
      failSilently=True)
      crt.Dialog.MessageBox(
           "Script Tab's index: " +
      str(crt.GetScriptTab().Index) + "\n" +
           "New Tab's index: " +
      str(objNewTab.Index))
      if objNewTab.Session.Connected != True:
            # Make sure we're not re-using
      disconnected
            # tabs, we don't want to close the
      script
            # tab, just any new tabs that get
      created by
            # calls to ConnectInTab(), but which
      don't
```

```
# result in a successful connection.
   if crt.GetScriptTab().Index !=
objNewTab.Index:
        crt.Dialog.MessageBox("Closing
failed tab")
        objNewTab.Close()
```

#### **Disconnect**

#### **Description**

Disconnects the current session.

#### Remarks

If the current session is not connected, Disconnect does nothing.

#### **VBScript**

**Syntax** 

object.Disconnect

#### **Python**

**Syntax** 

object.Disconnect()

#### Log

#### Description

Enables or disables logging.

#### Remarks

Starts or stops logging depending on the Boolean state of the "start" parameter. When logging is being started the optional Boolean "append" and "raw" parameters may be set to True to open the log file for appending or to log raw characters respectively. The append and raw parameters are optional and are false if not specified (When "start" is false the values of append and raw are ignored).

#### **VBScript**

#### **Syntax**

object.Log(start[, append[, raw]])

#### Python

#### **Syntax**

object.Log(start[, append[, raw]])

#### **LogUsingSessionOptions**

#### **Description**

Turns on logging using the logging options for the current session.

#### Remarks

If the session is an ad hoc session, the Default session's logging options will be used.

#### **VBScript**

#### **Syntax**

object.LogUsingSessionOptions

#### **Python**

#### **Syntax**

object.LogUsingSessionOptions()

#### **Print**

#### Description

Starts or stops autoprint.

#### Remarks

Starts or stops autoprint depending on the Boolean *start* parameter.

#### **VBScript**

#### **Syntax**

object.Print(start)

#### Python

#### **Syntax**

object.Print(start)

#### **SetStatusText**

#### **Description**

Allows you to set the text within the status bar for a specific session.

#### Remarks

Sets the status bar message to the specified text string.

#### **VBScript**

#### **Syntax**

object.SetStatusText(text)

#### Python

#### **Syntax**

object.SetStatusText(text)



## SessionConfiguration Object

#### **Description**

The SessionConfiguration object provides methods for accessing session configuration through scripting.

#### **Syntax**

SessionConfiguration. *Method*([arglist])

#### Remarks

Can be used in the following cases:

- Change settings in an existing session before connecting.
- · Convert an Excel spreadsheet to save SecureCRT or SecureFX (if installed) sessions.

#### **SessionConfiguration Object Methods**

#### **Methods**

ConnectInTab

**GetOption** 

<u>Save</u>

**SetOption** 

#### **Methods**

#### **ConnectInTab**

#### Description

Connects to a session in a tab or tiled session window.

#### Remarks

This method returns a Tab object.

#### **VBScript**

#### **Syntax**

```
Set objTab = object.ConnectInTab
```

#### **Example**

```
Set objConfig =
crt.OpenSessionConfiguration("Test")
objConfig.SetOption "Hostname", "host"
```

```
Set objTab = objConfig.ConnectInTab

Python
    Syntax
    objTab = object.ConnectInTab()

Example
    objConfig =
    crt.OpenSessionConfiguration("Test")
    objConfig.SetOption("Hostname", "host")
    objTab = objConfig.ConnectInTab()
```

#### **GetOption**

#### **Description**

Gets the value of the specified option.

#### Remarks

OptionName is a string parameter that is the name of the option. This method returns the value of the specified option. If the option is a string type (S), a string is returned. If the option is a multi-string (Z), an array of strings is returned. If the option is a DWORD (D), an integer is returned.

#### **VBScript**

#### **Syntax**

varname = object.GetOption(OptionName)

#### Example

```
Set objTab = crt.GetScriptTab
Set objConfig = objTab.Session.Config
szUsername = objConfig.GetOption("Username")
szSessionName = objTab.Session.Path
MsgBox "Username for current session (" & _
szSessionName & ") = " & szUsername

vPortForwards = objConfig.GetOption("Port
Forward Table V2")
nElements = UBound(vPortForwards)
If nElements = -1 Then
MsgBox "No port forward configuration defined"
Else
MsgBox nElements + 1 & _
" port forward entries exist in this session
```

```
(" &
    obiTab.Session.Path & ")"
    End If
    In the following example, the current session's Description is
    displayed.
    Set objTab = crt.GetScriptTab
    Set objConfig = objTab.Session.Config
    strDescription =
    Join(objConfig.GetOption("Description"),
    vbcrlf)
    crt.Dialog.MessageBox "Current Session's
    Description: " & vbcrlf & _
    strDescription
Python
  Syntax
    varname = object.GetOption(OptionName)
  Example
    objTab = crt.GetScriptTab()
    objConfig = objTab.Session.Config
    szUsername = objConfig.GetOption("Username")
    szSessionName = objTab.Session.Path
    crt.Dialog.MessageBox("Username for current
    session (" +
    szSessionName + ") = " + szUsername)
    vPortForwards = objConfig.GetOption("Port
    Forward Table V2")
    nElements = len(vPortForwards)
    if nElements == 0:
      crt.Dialog.MessageBox("No port forward
    configuration defined")
    else:
      crt.Dialog.MessageBox(str(nElements) +
      " port forward entries exist in this session
    (" + objTab.Session.Path + ")")
```

In the following example, the current session's Description is displayed.

```
objTab = crt.GetScriptTab()
objConfig = objTab.Session.Config
formattedDescription = []
arrDescription =
objConfig.GetOption("Description")
nElements = len(arrDescription)
for index in xrange(nElements):
   formattedDescription.append(arrDescription
   formattedDescription.append("\n")
   strDescription =
''.join(formattedDescription)
   crt.Dialog.MessageBox("Current Session's
Description: \n" + strDescription)
```

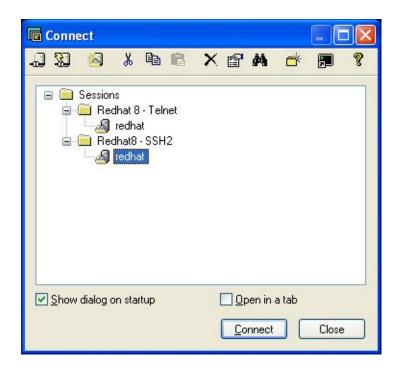
#### Save

#### **Description**

Saves the configuration.

#### Remarks

If SessionPath is not provided, the configuration will be saved to the .ini file that is associated with the Config object used. If the SessionPath provided is the same as an existing session configuration, the existing .ini file will be overwritten with the settings contained in the Config object used. If the SessionPath provided does not exist, a new session of that name will be created in the session path specified. Note that the session path is not a full file system path, rather it is the path relative to the folders as they appear within the SecureCRT Connect dialog, with the built-in "Sessions" folder being the "root" path. In the graphic below, the path to the selected session would be Redhat8 - SSH2\redhat.



```
VBScript
Syntax
   object.Save [SessionPath]
Example
   ' Save configuration to a new session.
   objConfig.Save "Redhat8 - SSH2\redhat - PF"
Python
Syntax
   object.save([SessionPath])
Example
   # Save configuration to a new session.
   objConfig.Save("Redhat8 - SSH2\redhat - PF")
```

#### **SetOption**

#### Description

Sets the specified option to the specified value.

#### Remarks

OptionName is a string parameter that is the name of the option. Value is the value to set the option to. If the option is a string type (S), a string should be specified for the value. If the option is a multi-string (Z), an array of strings should be specified for the

value. If the option is a DWORD (D), the value should be numeric. If there is an error this method will display a error message box. If the errors are not being displayed, the error message can be retrieved by using crt.GetLastErrorMessage.

#### **VBScript**

#### **Syntax**

object.SetOption OptionName, Value

#### Example

```
Config.SetOption "Terminal Protocol Name",
"SSH2"
```

In the following example, the current session's Description is changed.

```
Set objTab = crt.GetScriptTab
Set objConfig = objTab.Session.Config
strNewDescription =
  "This host is located in room 304." & vbcrlf
  & _
  "Requires SSH2 publickey authentication."
' Update the Config object's description;
' convert string to array using Split()
objConfig.SetOption "Description",
Split(strNewDescription, vbcrlf)
' Display updated configuration
strDescription =
Join(objConfig.GetOption("Description"),
vbcrlf)
crt.Dialog.MessageBox "Session's Updated
Description (New): " & _
  vbcrlf &
  strDescription
' Save the configuration
obiConfig.Save
```

The following shows how to use a password in a script. The example uses Server1's password to connect to Server.

**Note:** When set, the Password parameter must be encoded. The Session Password Saved parameter must be set to true for the Password parameter to be used.

```
Sub Main ()
      set config =
    crt.OpenSessionConfiguration("Server")
      set config1 =
    crt.OpenSessionConfiguration("Server1")
      config.SetOption "Session Password Saved", 1
      config.SetOption "Password",
    config1.GetOption("Password")
      config.ConnectInTab
    End Sub
Python
  Syntax
    object.SetOption(OptionName, Value)
  Example
    config.SetOption("Terminal Protocol Name",
    "SSH2")
    In the following example, the current session's Description is
    changed.
    objTab = crt.GetScriptTab()
    objConfig = objTab.Session.Config
    arrNewDescription = []
    arrNewDescription.append("This host is located
    in room 304.")
    arrNewDescription.append("Requires SSH2
    publickey authentication.")
    # Update the Config object's description;
    objConfig.SetOption("Description",
    arrNewDescription)
```

```
# Display updated configuration
formattedDescription = []
arrDescription =
objConfig.GetOption("Description")
nElements = len(arrDescription)
for index in xrange(nElements):
   formattedDescription.append(arrDescription[informattedDescription.append("\n")
strDescription = ''.join(formattedDescription)
crt.Dialog.MessageBox("Session's Updated
Description (New): \n" + strDescription)
# Save the configuration
objConfig.Save()
```

The following shows how to use a password in a script. The example uses Server1's password to connect to Server.

**Note:** When set, the Password parameter must be encoded. The Session Password Saved parameter must be set to true for the Password parameter to be used.

```
def main():
    config =
crt.OpenSessionConfiguration("Server")
    config1 =
crt.OpenSessionConfiguration("Server1")
    config.SetOption("Session Password Saved",
1)
    config.SetOption("Password",
config1.GetOption("Password"))
    config.ConnectInTab()
main()
```

#### **Description**

The Tab object provides access to the tab functions that exist for the current <u>connection</u> or <u>session</u>. The Tab object can also be used to work with tiled session windows.

#### **Syntax**

```
objTab.Property [ = expression ]
objTab.Method([arglist])
```

#### **Tab Object Properties and Methods**

<u>Properties</u>	<u>Methods</u>
Caption	<u>Activate</u>
<u>Index</u>	Clone
<u>Screen</u>	Close
Session	ConnectSftp

#### **Properties**

#### **Caption**

#### Description

Returns or sets the caption of the specified tab object.

#### Remarks

Setting this property sets the caption of the tab or tiled session window and sets the caption of the main application window if the tab or tiled session window is active.

#### **VBScript**

## **Syntax**

```
[ varname = ] tab.Caption
tab.Caption = newCaption
```

#### Python

#### **Syntax**

```
[ varname = ] tab.Caption
tab.Caption = newCaption
```

#### Index

#### Description

Returns the index for the tab object referenced by object.

#### Remarks

When sessions are tabbed, the index for each tab object matches its position in the tab bar. When sessions are tiled, the indexes of the tab objects may not match the indexes when tabbed, but will remain consistent while the sessions are tiled.

## **VBScript**

**Syntax** 

object.Index

#### **Python**

**Syntax** 

object.Index

#### Screen

#### Description

Returns a reference to the Screen object associated with the *tab* object.

#### Remarks

Object variables assigned from the screen property require the "Set" syntax. See the Screen object documentation for a description of its properties and methods.

#### **VBScript**

#### **Syntax**

Set object = tab.Screen

#### **Python**

#### **Syntax**

*object = tab*.Screen

#### Session

#### Description

Returns a reference to the Session object associated with the *tab* object..

#### Remarks

See the Session object documentation for a description of its

```
properties and methods.

VBScript
Syntax
Set object = tab.Session

Python
Syntax
object = tab.Session
```

#### **Methods**

#### **Activate**

#### Description

Brings the tab or tiled session window referenced by *object* to the foreground.

#### Remarks

A tab can receive and send text whether active or not. This method simulates a user clicking on a tab or tiled session window to activate.

```
VBScript
    Syntax
    object.Activate
Python
    Syntax
    object.Activate()
```

#### **Clone**

#### Description

Returns a reference to a tab object cloned from the specified *object* tab reference.

#### Remarks

Cloned sessions benefit from their parent session already having been authenticated to the server. Therefore, cloned sessions require no additional authentication.

```
VBScript
    Syntax
    Set cloneTab = object.Clone
Python
```

#### **Syntax**

cloneTab = object.Clone()

#### Close

#### **Description**

Closes the tab or tiled session window referenced by object.

#### Remarks

Any active connection associated with the tab (or tiled session window) referenced by *object* is terminated when the tab (or tiled session window) is closed. The tab or tiled session window running the script cannot be closed.

# VBScript Syntax object.Close Python Syntax object.Close()

#### **ConnectSftp**

#### **Description**

Creates an SFTP tab based on this tab. When in tiled mode, creates an SFTP session window based on this tiled session.

#### Remarks

SFTP sessions benefit from their parent session already having been authenticated to the server. Therefore, SFTP sessions require no additional authentication.

```
VBScript
   Syntax
   Set sftpTab = object.ConnectSftp
Python
   Syntax
   sftpTab = object.ConnectSftp()
```

#### **Description**

The Window object provides access to properties and methods relating to SecureCRT's window such as the window's visible state, caption, etc.

#### **Syntax**

Window.*Property* [ = expression ] Window.*Method*([arglist])

#### Remarks

SecureCRT's Window object is accessed through the top-level object's Window property.

#### **Window Object Properties and Methods**

PropertiesMethodsActiveActivateCaptionShow

<u>State</u>

#### **Properties**

#### **Active**

#### **Description**

Returns True if SecureCRT's window is the active window, otherwise False.

#### Remarks

Boolean read-only property. On Linux, False may be returned even if the SecureCRT icon is flashing.

#### **VBScript**

#### **Syntax**

[ varname = ] crt.Window.Active

#### **Python**

#### **Syntax**

[ varname = ] crt.Window.Active

#### **Caption**

#### **Description**

Returns or sets the title or caption of SecureCRT's application window and the active session tab or tiled session window.

#### Remarks

Read/write string property.

#### **VBScript**

#### **Syntax**

```
crt.Window.Caption [ = string ]
[ varname = ] crt.Window.Caption
```

#### **Python**

#### **Syntax**

```
crt.Window.Caption [ = string ]
[ varname = ] crt.Window.Caption
```

#### **State**

#### Description

Returns a number indicating the state of SecureCRT's application window.

#### Remarks

Read-only numeric property. The state may be one of the following values depending on SecureCRT's window state:

- 0 hidden
- 1 visible (normal)
- 2 minimized
- 3 maximized

#### **VBScript**

#### **Syntax**

crt.Window.State

#### **Python**

#### **Syntax**

crt.Window.State

#### **Methods**

#### **Activate**

#### **Description**

Gives focus to the SecureCRT window, bringing the window to the top of the desktop.

#### Remarks

On Linux, calling Activate() from a script may cause the SecureCRT icon to flash instead of raising SecureCRT to the top. This is the required behavior for this operating system.

#### **VBScript**

#### **Syntax**

crt.Window.Activate

#### **Python**

#### **Syntax**

crt.Window.Activate()

#### **Show**

#### **Description**

Shows, hides, minimizes, or maximizes SecureCRT's application window

#### Remarks

The state may be one of the following values:

- 0 Hide
- 1 Show (Normal)
- 2 Minimize
- 3 Maximize

### **VBScript**

#### **Syntax**

crt.Window.Show state

#### Python

#### **Syntax**

crt.Window.Show(state)

## Example Scripts

Example VBScripts can be found in the SecureCRT installation folder. If you would like to modify an example script, please modify a copy of the script file because if the original scripts are updated, the installer will overwrite the example scripts.

The following are descriptions of the example scripts:

- AddCommentToLogFile.vbs This example script allows you to add a custom line of text to your SecureCRT log file.
- AutoResponder.vbs Demonstrates how to set up a list of commands/text that get sent to the remote as soon as a corresponding "trigger" string/text is detected from the remote system.
- BasicReadStringExample.vbs This example script captures the output of a command (Is -I) sent to a remote machine by using the Screen.ReadString() method. The captured text is then displayed in a message box window.
- ConnectToMultipleSessionsAndSendCommands.vbs Reads in sessions from a file (one session per line) and connects to each one (one at a time) and sends a series of commands to the remote, then disconnects from the session and moves on to the next session.
- CopyOutputToClipboard.vbs This script example is designed to run while connected to a Cisco Pix firewall or other router device.
- GoogleSelectedText.vbs When this script is launched, the text selected within the terminal window is used as the search term for a web search using google.com.
- LogOutputOfSpecificCommand-UseReadString.vbs Sends commands one by one as listed in the g vCommands() array to the remote machine. The results of each command are captured into a variable, and then written to an individual log file (one log file for each command). Once all the commands have been run, Windows Explorer is launched, with the first command output file selected within the explorer window.
- SaveSelectedTextToFile.vbs If non-whitespace text is selected within the terminal screen, the user will be prompted for a location and filename in which to store the selected text. The selected text

- will then be saved to the file specified by the user.
- SendCommandToAllTabs.vbs This example script shows how to send the same command to all tabs that currently have an active connection.
- UseIEAsCustomDialog.vbs This sample script shows how the Internet Explorer automation object can be used to create custom dialog that can be used within a SecureCRT script.
- UseIEAsListOutput.vbs This example script shows how to create a custom dialog using the InternetExplorer.Application ActiveX object. This example also shows how data can be passed between the InternetExplorer object and SecureCRT.



## Introduction to System Administration

#### Issues

The topics included in this section contain information that may be useful to system administrators who find it necessary to configure SecureCRT for use by a large number of different users.

These topics include:

How to specify the directory in which the SecureCRT configuration files will Specifying the Configuration Directory.

be stored or loaded.

How to configure SecureCRT to

remember passphrases used with public-Caching Passphrases

private key authentication.

How to configure administrator options **Administrative Template** 

from Microsoft Management Consol.

How to enable FIPS mode. FIPS Mode



## Specifying the Configuration Folder

This topic explains how to specify the folder SecureCRT will use to store and load the initialization files containing the settings for global and session options.

#### If this is the first time you have installed SecureCRT

If this is the first installation of SecureCRT on your system, the **Browse** for Folder dialog will appear when SecureCRT is run for the first time after installation. In the **Browse for Folder** dialog, select the folder in which you would like to store the settings for global and session options and then click on the **OK** button.

#### If you are installing SecureCRT over a previous version:

If you have an earlier version of SecureCRT installed on your system as well as a 5.0 version installed, the Migrate SecureCRT Settings dialog will appear the first time you run the 5.0 version. In this dialog, you are asked if you want to migrate the options settings from your earlier version of SecureCRT. If you would like to migrate the configuration settings from the prior version to the newer version of SecureCRT, you must also specify the configuration folder at this time:

- 1. Specify the configuration folder by entering the full folder path or by clicking on the **Browse** button and selecting the path from the Browse for Folder dialog.
- 2. Once you have specified the configuration folder (if different than the default configuration folder), click on the Yes button. Your configuration settings will be migrated and stored in the configuration folder that you specified in the first step above.

**Note:** If you do not want to migrate your configuration settings from the older installation of SecureCRT, click on the **No** button in the Migrate SecureCRT Settings dialog and the Browse for Folder dialog will appear, in which you will be able to specify the SecureCRT configuration folder.

#### **Changing the Configuration Folder**

To change the configuration folder, complete the following steps:

Start SecureCRT.

- 2. Open the <u>options</u> menu and click on **Global Options...** to open the <u>Global Options</u> dialog.
- 3. Click on the **General** category.
- 4. In the **Configuration folder** entry box, enter the path to the desired configuration folder.
- 5. Click on the **OK** button to exit the **Global Options** dialog and restart SecureCRT to instate the change of folders.

**Note:** Changes to the configuration folder will not take effect until SecureCRT is restarted.

#### **Specifying a Configuration Folder from the Command Line**

The command-line option <u>IF. folder</u> allows you to specify an alternate configuration folder when you start SecureCRT from the command line. When SecureCRT is started with the /F option, the specified folder will override the default setting for the current instance of SecureCRT. Each instance of SecureCRT can use a different configuration folder.

To specify a configuration folder from the command line, enter the following command:

SecureCRT /F folder

**Note:** If the folder pathname contains spaces, you must enclose the argument inside quotation marks.

## Locking a Session

Sessions can be locked or unlocked by selecting the **Lock/Unlock Session** option on the **File** menu. When this option is selected, the **Lock** Session or Unlock Session dialog appears, as appropriate, allowing you to assign a password and lock the session, or give the original password to unlock the session. The **Lock Session** or **Unlock Session** dialogs also allow you to lock or unlock all sessions simultaneously using a single password.

When a session is locked, actions that would affect the session are disallowed, including the following:

- **Keyboard** input
- Starting and canceling scripts
- Disconnecting and reconnecting the session
- Starting a file transfer
- Closing the session
- Editing the session options

Scripts that are started before a session is locked will continue to run, but can only access their own session and any unlocked sessions. This means that a script that is running in one session cannot access a different tab that is locked.

A MENU TAB LOCK menu function is available for use in custom keymaps, toolbars, and the user button bar.



## Overview of Ymodem

Ymodem is a simple file transfer protocol that supports basic error detection methods to ensure that information is not lost or corrupted during a transfer. Ymodem is based on Xmodem and is essentially the same except that Ymodem sends the file's name, size, and timestamp before actually transferring the file.

The SecureCRT Ymodem implementation supports both 128-byte and Ymodem-1K (1024-byte) packet sizes.

Unlike Xmodem, Ymodem supports multiple-file ("batch") transfers.

Topics for this section include:

Changing the Default File Transfer Directories Downloading a File with Ymodem Uploading a File with Ymodem



## **Using an Exported Public Key**

In order to use a public key that has been exported, you will need to complete the following steps:

- 1. Configure the SSH2 server to recognize your public-key file (e.g., Identity.pub). Instructions are provided for configuring VanDyke Software's VShell® server, OpenSSH, SSH Communications, and Data Fellows Servers.
- 2. Configure SecureCRT to use the certificate store with the exported public key(s).
  - a. Open the Session Options dialog and navigate to the Connection/SSH2 category.
  - b. In the **Authentication** group, enable **PublicKey** authentication.
  - c. Select **PublicKey** from in the **Authentication** list and click on the **Properties** button.
  - d. Select **Use identity or certificate file** and enter capi::rawkey::my in the entry box.

If you would like this to be the default for all sessions, open the **Global Options** dialog, navigate to the **SSH2** category, and repeat Step d.

Return to the Tools Menu



## SSH2 Quick Connect Options

When ssh2 is selected as the **Protocol** in the Quick Connect dialog, the connection options are as follows:

#### Hostname

The hostname or IP address of the remote machine that provides the SSH2 service.

#### **Port**

The port number of the SSH2 service on the remote machine. For SSH2, the default port is 22.

#### **Firewall**

If your connection involves a firewall, select your firewall from the list of firewalls that have been configured in the Global Options/Firewall dialog.

#### Username

The username used to log on to the remote machine.

#### **Authentication group**

SecureCRT supports several authentication methods for connecting to SSH2 servers, and will attempt to connect using them in the order that you specify.

**Password** authentication transmits the user's password to the server to authenticate the connection. The transmitted password is protected from network eavesdropping, due to the cipher encryption of the data channel.

**Note**: If the remote machine supports both the SSH2 protocol and changing passwords at the protocol level, the password for an SSH2 session can be changed from the **Password Properties** dialog, which is accessed by selecting **Password** in the **Authentication** group and clicking on the **Properties** button.

PublicKey authentication uses a public/private key pair to authenticate the connection. During the authentication process, the client and the server negotiate a public key to use for the connection. Once a public key has been determined, the client uses the corresponding private key to perform a signature operation over a unique connection identifier. This signature is then sent to the server for verification. If verification is successful, the client is given

permission to connect to the server. The security of the mechanism requires that no one but the owner have access to the private key. The private key is stored locally in an <u>identity file</u>. Also, prior to using public-key authentication, the public key must be made available to the SSH2 server. For more information on generating private-public key pairs, See <u>Public-Key Authentication for SSH2</u>.

**Keyboard Interactive** authentication allows you to use the keyboard to respond to challenges put forth by the server.

**GSSAPI** (Generic Security Services Application Program Interface) is a generic API for performing client/server authentication. GSSAPI allows SecureCRT to authenticate with a server without knowing anything about the specific authentication mechanism in use. For more information about using GSSAPI, see the **GSSAPI Properties** dialog.

SecureCRT also supports the use of  $\underline{x.509}$  certificates.



## SSH1 Quick Connect Options

When ssh1 is selected as the **Protocol** in the Quick Connect dialog, the connection options are as follows:

#### Hostname

The hostname or IP address of the remote machine that provides the SSH1 service.

#### **Port**

The port number of the SSH1 service on the remote machine. For SSH1, the default port is 22.

#### **Firewall**

If your connection involves a firewall, select your firewall from the list of firewalls that have been configured in the Global Options/Firewall dialog.

#### Username

The username used to log on to the remote machine.

#### Authentication

SecureCRT supports three types of authentication for connecting to SSH1 servers: password, RSA, and TIS.

**Password** authentication transmits the user's password to the server to authenticate the connection. The transmitted password is protected from network eavesdropping, due to the cipher encryption of the data channel. For this reason, some SSH1 servers reject the use of password authentication if the cipher is set to None.

**RSA** authentication uses a public/private key pair to authenticate the connection. The general mechanism behind RSA authentication is that the SSH1 server "challenges" the client to decrypt a message encoded using the user's public key stored on the server. Upon connecting, the SSH1 server generates a random value, encrypts the value using the user's public key and sends the encrypted challenge to the client. The client authenticates the connection by successfully decrypting the challenge using the user's private key. The security of the mechanism requires that no one but the owner have access to the private key. The private key is stored locally in an identity file. The first time you connect to an SSH1 server using RSA authentication, SecureCRT will prompt you for the location of this file. Also, prior to using RSA authentication, the

public key must be made available to the SSH1 server.

**Note:** RSA authentication is only supported by the SSH1 protocol and is not an available option for the <u>SSH2</u> protocol. See <u>Public-Key</u> <u>Authentication for SSH1</u> to learn more about generating identity files and other setup issues.

**TIS** firewall authentication uses the TIS firewall server to provide a challenge phrase / response combination. SSH1 servers must be configured to offer TIS authentication

.



## Telnet Quick Connect Options

When Telnet is selected as the Protocol in the Quick Connect dialog, the connection options are as follows:

#### Hostname

The hostname or IP address of the remote host.

#### **Port**

The port number of a service on a remote machine. For Telnet , the default port is 23.

#### **Firewall**

If your connection involves a firewall, select your firewall from the list of firewalls that have been configured in the Global Options/Firewall dialog.



## Telnet/SSL Quick Connect Options

When Telnet/SSL is selected as the Protocol in the Quick Connect dialog, the connection options are as follows:

#### Hostname

The hostname or IP address of the remote host.

#### **Port**

The port number of a service on a remote machine. For Telnet/SSL , the default port is 992.

#### **Firewall**

If your connection involves a firewall, select your firewall from the list of firewalls that have been configured in the Global Options/Firewall dialog...



# RLogin Quick Connect Options

When RLogin is selected as the Protocol in the Quick Connect dialog, the connection options are as follows:

#### **Hostname**

The hostname or IP address of the remote host.

#### Username

The username used to log onto the remote machine.



## Serial Quick Connect Options

When Serial is selected as the **Protocol** in the Quick Connect dialog, the connection options are as follows:

#### **Port**

The serial port (COM1, COM2, etc.) used by the session.

#### **Baud rate**

Select the baud rate for the device. The default value is 38400. The baud rate can be changed from the session Options/Connect/Serial dialog while a session is connected.

#### **Data bits**

The number of communication data bits. The default value is 8.

#### **Parity**

Choose the parity setting supported by the connected device. The default setting is None.

#### Stop bits

The number of stop bits sent after each character. The default value is 1.

#### DTR/DSR

Select this option to use the DTR/DSR (data-terminal-ready/data-setready) signals with the connected device. By default, this is not selected.

#### RTS/CTS

Select this option to use RTS/CTS (request-to-send/clear-to-send) hardware flow control. By default, this option is selected.

#### XON/XOFF

Select this option to use XON/XOFF software flow control. By default, this option is not selected.



## TAPI Quick Connect Options

When tapi is selected as the **Protocol** in the Quick Connect dialog, the connection options are as follows:

#### Dial using

Select the modem you would like to use from the **Dial using** list of modems. If you have a modem installed, you can also configure it while in the Session Options dialog by selecting the modem and clicking on the Configure button.

#### **Country code**

Enter the country code associated with the number you are dialing.

#### Area code

Enter the area code for the number you are dialing.

#### Phone number

Enter the phone number that you are dialing. SecureCRT also allows you to specify any alternate phone numbers you wish to dial using the same session .



## Raw Quick Connect Options

When Raw is selected as the **Protocol** in the Quick Connect dialog, the options are as follows:

#### Hostname

The hostname or IP address of the remote host.

#### **Port**

The port number of a service on a remote machine.

#### **Firewall**

If your connection involves a firewall, select your firewall from the list of firewalls that have been configured in the Global Options/Firewall dialog.



## Using Compression

Data that is to be transferred over an SSH connection can be automatically compressed before it is encrypted and sent. The data is then automatically uncompressed after it is received and decrypted. Compressing data can improve transfer speed over slower network links.

To configure SecureCRT to compress session data for an SSH1 session, follow these steps:

- 1. Select the SecureCRT session for which you want to compress data.
- 2. Open the options menu and select the **Session Options...** command.
- 3. Select the connection/SSH1/Advanced Category.
- 4. In the **Options** group, select the type of **Compression** that you want (this option is None by default).
- 5. You can then choose the compression **Level** setting. The **Level** ranges from 0 (minimal compression) to 9 (maximum compression). A higher **Level** means better compression but slower performance. The default value is 5.

To configure SecureCRT to compress session data for an SSH2 session, follow these steps:

- 1. Select the SecureCRT session for which you want to compress data.
- 2. Open the options menu and select the **Session Options...** command.
- 3. Select the **Connection/SSH2/Advanced** category.
- 4. In the **Options** group, select the type of **Compression** that you want (this option is None by default).
- 5. You can then choose the compression **Level** setting. The **Level** ranges from 0 (minimal compression) to 9 (maximum compression). A higher **Level** means better compression but slower performance. The default value is 5.

# 😼 Using NTLM

To use NT LAN Manager (NTLM) (a Windows authentication protocol) with SecureCRT Telnet and Telnet/SSL sessions, you must have a valid Windows security context for NTLM authentication to succeed. Typically, in a Windows network, this means that the local machine and remote server would be members of the same (or trusted) Active Directory Domains. Once a user authenticates to Windows on the client side, the user then has a security context that can be used to authenticate the client to the remote server (assuming that the security context is valid between both the client and server).

As an example, the remote machine must have a username and password associated with the session that is attempting to access the Telnet server. The server's machine must have that same username and password listed under its **Users and Passwords** facility in the Windows Control Panel.

**Note:** If a user's password is changed on the local machine, it must also be changed on the Telnet server.

In SecureCRT, NTLM authentication is enabled by default. You can disable NTLM by unchecking the **Enable NTLM authentication** option on the <u>session Options/Connection/Telnet</u> dialog.



## 😼 Sending a Serial Break

When using a Serial connection, there are several ways to send a breakline condition.

- If the VT220 keymap is selected, you can use the F5 key.
- When using any of the keymappings supplied with SecureCRT, you can generate a break by using the CTRL plus PAUSE/BREAK key combination.
- You can also use the Button Bar or Keymap Editor to arbitrarily map the Telnet function **TN\_BREAK** to a button or to any key you like.

## Answerback Substitutions

When the Answerback option has been selected, SecureCRT will respond to the ENQ character with the specified answerback message defined in the Terminal/Emulation/Advanced Category of the Session Options dialog. You can include the following substitution variables to be sent to the remote server when the server sends the ENQ (control byte 5) command:

- \a IPv6 address of the first adapter on the local machine (the client)
- **\b** Break
- **\c** Computer Name of the local machine (the client)
- **le** Escape
- **\i** IPv4 address of the first adapter on the local machine (the client)
- **II** Windows login name associated with the SecureCRT process running on the local machine (the client)
- **\m** MAC address of the first adapter on the local machine (the client)
- **\n** Newline
- **\p** Pause
- **\r** Carriage return
- \t Tab
- \u Username used to login to the remote server
- **\v** Send the current contents of the paste buffer

#### For example:

will be sent to the remote server as:



## **Menu Configuration Command Strings**

Several example menu files can be found in the SecureCRT install folder in the Menus subfolder. The file default.mnu contains most of the currently supported commands. It is recommended that you use one of the examples as a template for creating a custom menu.

**WARNING:** Before modifying an example menu file, it should be renamed or copied to a different location because otherwise, it will be overwritten when an update of SecureCRT is installed.

The following are command strings that can be used in the menu configuration file in addition to those in the default.mnu example menu. For more information, see <u>Customizing the User Interface</u>.

MENU_WINDOW_NEXT	Changes focus to the next application window
MENU_WINDOW_PREV	Changes focus to the previous application window
MENU_WINDOW_CLOSE_ALL	Close all open application windows

MENU_WINDOW_LIST1	Show list of windows
-------------------	----------------------

MENU_TAB_LIST1	Show list of tabs in current window
----------------	-------------------------------------

	Load the default main menu
--	-------------------------------

	Load the default system menu
MENU_LOAD_DEFAULT_CONTEXT_MENU	Load the default context menu
	Load the default toolbar menu

## These items should only be used in the system menu

SYS_MENU_RESTORE	The system menu Restore item
SYS_MENU_SIZE	The system menu Size item
SYS_MENU_MOVE	The system menu Move item
SYS_MENU_MINIMIZE	The system menu Minimize item
SYS_MENU_MAXIMIZE	The system menu Maximize item
SYS_MENU_CLOSE	The system menu Close/Exit item

## Accepting a Host Key

The **New Host Key** dialog appears when SecureCRT attempts to connect to an SSH server whose host key is not contained in the host key database. This can happen when SecureCRT first encounters an SSH server or when the host name given is an alias that resolves to several different IP addresses.

**Note:** SecureCRT will warn you about aliases where possible. However, if you are connecting through a firewall, this will not be possible. because the firewall prevents SecureCRT from discovering the possibility of the existence of an alias.

For SSH2 connections, the **New Host Key** dialog displays the server's public key in a fingerprint format for easier verification. You should check this fingerprint against the actual server fingerprint to verify that the server to which you have connected is legitimate

Choose one of the options below to proceed with the connection

- **Accept once**: Choosing this option causes the host key from the SSH server to be accepted, but the host key is not saved in the database.
- **Accept & save**: Choosing this option causes the host key from the SSH server to be accepted and it is saved in the database.
- **Cancel**: Choosing this option causes the host key to not be accepted and the connection process is terminated.

#### **Host Key Database**

SecureCRT keeps the host keys that you accept in the host key database. To determine the default location for this database, SecureCRT queries the system for the Application Data folder; generally, this is the C:\Documents and Settings\<username>\Application Data folder. This is also where SecureCRT tries to install the Config folder.

The location of the host key database folder cannot be changed in the SecureCRT user interface, but can be changed manually by editing the SSH2.ini file in the Config folder. If you have any saved host keys, you will have to move them to the new location or accept them again.



## Downloading a File with Ymodem

To download a file from a remote machine, complete the following steps:

- 1. On the remote machine, issue the appropriate Ymodem command to instruct the Ymodem software to send the file that you want to download. The remote machine software should then prompt you to initiate the transfer.
- 2. To initiate the Ymodem download, open the SecureCRT Transfer menu and select **Receive Ymodem**. The file will be transferred and saved to the default download directory when you select the command.



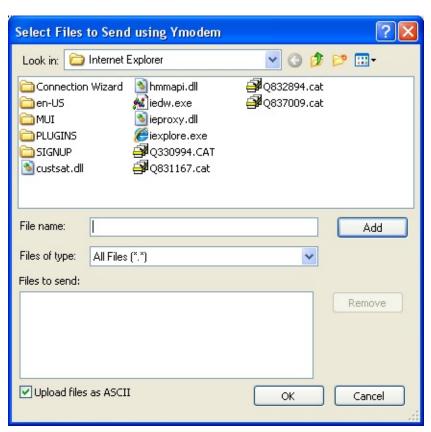
## **Uploading a File with Ymodem**

To upload a file to a remote machine, complete the following steps:

 On the remote machine, issue the appropriate Ymodem command to instruct the Ymodem software to receive the file that you want to upload. This will cause SecureCRT to open the Select Files to Send using Ymodem dialog (shown below). Note that Ymodem must be installed on the remote machine to transfer files with Ymodem.

**Note:** Another way to open this dialog is to open the <u>Transfer</u> menu and select **Send Ymodem**.

- In the Select Files to Send using Ymodem dialog, select the file(s) that you want to upload to the remote machine and click on the Add button to add the file(s) to the Files to send list. You can also choose to send your file(s) in ASCII instead of binary by checking the Upload files as ASCII check box.
- 3. When you have finished adding all the files that you want to upload, click on the **OK** button.



Select Files to Send using Ymodem Dialog