SSH Tectia® Client/Server 6.1

Migration Guide

30 November 2009

SSH Tectia® Client/Server 6.1: Migration Guide

30 November 2009

SSH Communications Security Corp.
Valimotie 17, FI-00380 Helsinki; Finland

Table of Contents

Chapter 1 About This Document

This guide describes how to upgrade an SSH Tectia 4.x version to 6.1. Readers are expected to be familiar with SSH Tectia Client and Server 4.x.

The architecture of the SSH Tectia products was changed in release 5.0. The new SSH G3 architecture allows for faster file transfer speeds and better scalability.

Along with the architecture changes, the configuration file format used by SSH Tectia Client and SSH Tectia Server was changed in release 5.0; now it uses a more flexible and robust XML format.

Migrating existing 4.x installations of SSH Tectia Client and Server will require planning and rewriting of the configuration files. This document contains instructions on planning and carrying out the migration from SSH Tectia Client and Server 4.x to 6.1

This document contains the following information:

- introduction to the SSH Tectia client/server solution 6.1
- planning the migration
- changes in the SSH Tectia Server configuration
- changes in the SSH Tectia Client configuration
- examples of migrating an SSH Tectia 4.4 configuration to SSH Tectia 6.1.

1.1 Related Documents

SSH Tectia Client/Server Product Description contains important information on SSH Tectia products, and we recommend that you read it before this document.

SSH Tectia Server 4.4 Windows Administrator's Guide and SSH Tectia Unix manual pages contain descriptions of the 4.4 configuration options.

SSH Tectia Client 6.1 User Manual and *SSH Tectia Server 6.1 Administrator Manual* contain detailed instructions on using the new SSH Tectia Client and Server, including descriptions of the configuration files.

1.2 Documentation Conventions

The following typographical conventions are used in SSH Tectia documentation:

Table 1.1. Documentation conventions

Convention	Usage	Example
Bold	Menus, GUI elements, strong emphasis	Click **Apply** or **OK**.
→	Series of menu selections	Select File → Save
`Monospace`	Filenames, commands, directories, URLs etc.	Refer to `readme.txt`
Italics	Reference to other documents or products, emphasis	See *SSH Tectia Client User Manual*
#	In front of a command, # indicates that the command is run as a privileged user (root).	`# rpm --install package.rpm`
$	In front of a command, $ indicates that the command is run as a non-privileged user.	`$ sshg3 user@host`
\	At the end of a line in a command, \ indicates that the command continues on the next line, but there was not space enough to show it on one line.	`$ ssh-keygen-g3 -t rsa \` `  -F -c mykey`

Note

A Note indicates neutral or positive information that emphasizes or supplements important points of the main text. Supplies information that may apply only in special cases (for example, memory limitations, equipment configurations, or specific versions of a program).

Caution

A Caution advises users that failure to take or to avoid a specified action could result in loss of data.

1.2.1 Operating System Names

When the information applies to several operating systems versions, the following naming systems are used:

- **Unix** refers to the following supported operating systems:
  - HP-UX
  - IBM AIX

  - Red Hat Linux, SUSE Linux
  - Linux on IBM System z
  - Sun Solaris
  - IBM z/OS, when applicable; as SSH Tectia Server for IBM z/OS is running in USS and uses Unix-like tools.

- **z/OS** is used for IBM z/OS, when the information is directly related to IBM z/OS versions.
- **Windows** refers to all supported Windows versions.

1.2.2 Directory Paths

The following conventions are used in the documentation to refer to directory paths:

$HOME
: A Unix environment variable, that indicates the path to the user's home directory.

%APPDATA%
: A Windows environment variable, that indicates the path to the user-specific Application Data folder. By default expands to:

  "`C:\Documents and Settings\<username>\Application Data`" on pre-Vista Windows versions

  "`C:\Users\<username>\AppData\Roaming`" on Windows Vista.

%USERPROFILE%
: A Windows environment variable, that indicates the path to the user-specific profile folder. By default expands to:

  "`C:\Documents and Settings\<username>`" on pre-Vista Windows versions

  "`C:\Users\<username>`" on Windows Vista.

<INSTALLDIR>
: Indicates the default installation directory on Windows:

  "`C:\Program Files\SSH Communications Security\SSH Tectia`"

1.3 Customer Support

All SSH Tectia product documentation is available at http://www.ssh.com/support/documentation/.

If the product documentation does not answer all your questions, you can find the SSH Tectia FAQ and Knowledge Base at https://support.ssh.com/.

If you have purchased a maintenance agreement, you are entitled to technical support from SSH Communications Security. Review your agreement for specific terms and log in at https://support.ssh.com/.

Information on submitting support requests, feature requests, or bug reports, and on accessing the online resources is available at http://www.ssh.com/support/contact/.

Chapter 2 Introduction

The following areas are different in the 4.x and 6.x versions of the SSH Tectia client/server solution.

Software architecture

The 6.x version uses the new SSH G3 architecture which is the third complete implementation of the Secure Shell protocol. SSH G3 is based on and fully compatible with the standard SSH2. SSH G3 allows faster secure file transfers and application tunneling without causing processing bottlenecks.

In the 6.x version, the SSH Tectia Server architecture is divided to servant and master processes for better scalability. This change is transparent to the administrator.

In the 6.x version, the SSH Tectia Client and SSH Tectia ConnectSecure architecture includes the Connection Broker, that handles all encryption and authentication tasks for these Secure Shell client components.

Product packaging

The SSH Tectia solution includes SSH Tectia Client, ConnectSecure, and Server as separate product packages. In the 6.0 version, SSH Tectia ConnectSecure has been added as a new product.

On Unix, the SSH Tectia Server package includes the basic Secure Shell client components, but the SSH Tectia Client package no longer include the Secure Shell server components (as with 4.x).

In 6.x, the advanced file transfer and tunneling features have been concentrated to SSH Tectia ConnectSecure.

SSH Tectia Connector is no longer available in release 6.0.

Software configuration

SSH Tectia Client, ConnectSecure, and Server 6.x use standard XML-based configuration files. The flexible XML configuration file format allows implementing even complex security policies easily.

Changes in SSH Tectia Client and Server are detailed in the following sections. For general information on the SSH Tectia products and features, see also *SSH Tectia Client/Server Product Description*.

2.1 SSH Tectia Server

SSH Tectia Server has been updated between releases 4.x and 6.x. The following sections detail the changes made to:

- Architecture, see Section 2.1.1
- Product packaging, see Section 2.1.2
- Configuration, see Section 2.1.3

2.1.1 Architecture

SSH Tectia Server has new distributed master/servant architecture, but it is transparent to the administrator.

2.1.2 Product Packaging

In 4.x, there were two separate versions of the product available: SSH Tectia Server (A) for system administration and SSH Tectia Server (T) for application tunneling. Starting from release 6.0, there is one generic SSH Tectia Server for Unix and Windows platforms, and the separate product SSH Tectia Server for IBM z/OS for mainframes. This manual handles only the generic SSH Tectia Server.

SSH Tectia Server installation packages are contained in bundles starting from release 6.0. There is a separate installation bundle for each supported platform. See the differences between 4.x and 6.x installation packages in the following sections.

Table 2.1 shows the SSH Tectia Server packages on the different platforms.

Table 2.1. The SSH Tectia Server installation packages

SSH Tectia Server on Unix and Linux	SSH Tectia Server on Windows
`common`	`server`
`server`	

2.1.2.1 Unix

In version 4.4 (and earlier) on Unix platforms, the SSH Tectia Server installation package included both the Secure Shell server (`sshd2`) and client (`ssh2`, `sftp2`, `scp2`) functionality and also the necessary libraries (for example crypto libraries) and auxiliary software (for example `ssh-keygen2`, `ssh-certview`). The functionality of the server (A or T) was controlled by a license file. The SSH Tectia Client package for Unix platforms included the same files as the SSH Tectia Server packages but the functionality of the server component was limited by license to accept at most two simultaneous connections.

In version 6.x, SSH Tectia Server includes the following installation packages:

- The *common* package contains auxiliary tools and libraries that are shared by client and server. The *common* files are included in the SSH Tectia Client, ConnectSecure, and Server packages, because most Unix packaging systems do not allow the same files to be shared between multiple packages. This also makes it possible to install each of SSH Tectia Client, ConnectSecure, and Server independently on separate hosts.
- The *server* package includes the Secure Shell server functionality.
- The *client* package provides the basic Secure Shell client functionalities.

2.1.2.2 Windows

On Windows platforms, SSH Tectia Client and Server were packaged separately already in 4.x versions. There was also SSH Tectia Connector for Windows. The functionality of Server (A or T) was controlled by a license file. Both SSH Tectia Client and Server on Windows installed the tools and libraries common to both packages.

SSH Tectia Server 6.x is one single package, that contains all necessary tools, libraries, and other auxiliary files.

2.1.3 Configuration File

SSH Tectia Server 6.x uses an XML-based configuration file `ssh-server-config.xml`. The configuration file can be edited with an ASCII text editor or an XML editor. On Windows, you can use the SSH Tectia Server Configuration tool.

The new XML-based configuration format allows for more flexible customization of connection settings, for example, based on user parameters or the location of the client machine.

- Access restrictions and user-specific settings are now defined using the `selector` elements in the configuration file.
- The globbing syntax used in patterns has been simplified.

New configurations are easy to generate with the SSH Tectia Server Configuration GUI, which offers two configuration modes: Simple and Advanced. The Simple mode is for making basic connection, authentication and service rule settings that are applied to all users. The Advanced mode allows you to modify all possible configuration settings.

Global server parameters are straightforward to migrate from SSH Tectia Server 4.x to 6.x, but migrating the access restrictions will in most cases require some re-planning.

For more information on changes in the SSH Tectia Server configuration, see Chapter 4.

2.2 SSH Tectia Client

SSH Tectia Client has been updated between releases 4.x and 6.x. The following sections detail the changes made.

- Architecture, see Section 2.2.1
- Product packaging, see Section 2.2.2
- Configuration, see Section 2.2.3

2.2.1 Architecture

The architecture of SSH Tectia Client has been changed to include a new component called the Connection Broker.

The Connection Broker implements the Secure Shell protocol and it is a common component for client-side tools `sshg3`, `sftpg3`, `scpg3`, SSH Tectia terminal GUI client and secure file transfer GUI client. Instead of opening the SecSh transport sessions themselves, the client programs request channels from the Connection Broker. Channels can be terminal channels, remote program execution channels, SFTP channels, or TCP tunnels, for instance.

In SSH Tectia Client 6.x, all connection-related settings, such as connection profiles, as well as the used ciphers, authentication methods, and tunnels are part of the Connection Broker configuration. In SSH Tectia Client on Windows, the settings related to the GUI appearance and SFTP directories, for example, are part of the Windows client configuration.

For more information on the Connection Broker, see *SSH Tectia Client/Server Product Description*.

2.2.2 Product Packaging

The new product packaging of SSH Tectia Client and Server is described in Section 2.1.2 above.

Starting from release 6.0, SSH Tectia Client is delivered as a bundle of installation packages, a separate bundle for each supported platform. During the installation, you can select the installation packages to implement those functions that best suit the current needs and the environment.

On Unix, SSH Tectia Client includes the following installation packages:

- The *common* package contains auxiliary tools and libraries that are shared by client and server. The *common* files are included in the SSH Tectia Client, ConnectSecure, and Server packages, because most Unix packaging systems do not allow the same files to be shared between multiple packages. This also makes it possible to install each of SSH Tectia Client, ConnectSecure, and Server independently on separate hosts.

- The *client* package includes the basic Secure Shell client functionalities, the terminal client and the file transfer client.

On Windows platforms, SSH Tectia Client is one single package, that contains all necessary tools, libraries, and other auxiliary files. You can select the components to be installed from the Installation Wizard.

Table 2.2 summarizes the SSH Tectia Client packages on different platforms.

Table 2.2. The SSH Tectia Client installation packages

SSH Tectia Client on Unix and Linux	SSH Tectia Client on Windows
`common`	`client`
`client`	

2.2.3 Configuration File and Command-line Tools

SSH Tectia Client uses the same XML-format configuration file as the Connection Broker, `ssh-broker-config.xml`. The configuration file can be edited with an ASCII text editor or an XML editor. You can also use the SSH Tectia Configuration GUI to configure the Client. The settings specific to the GUI client are stored in the file `global.dat`, which uses similar format as in SSH Tectia Client 4.x.

Connection profiles, which in SSH Tectia Client 4.x were available only with the Windows GUI client, can now be used with the command-line tools, as well as with GUI both on Windows and Unix.

The command-line tool names and some of the command-line options have been changed between SSH Tectia Client releases 4.x and 6.0.

For more information on changes in the SSH Tectia Client configuration, see Chapter 5.

For more information on changes in the command-line clients, see Section 5.3.

2.3 SSH Tectia ConnectSecure

SSH Tectia ConnectSecure is a new product in 6.0, designed for securing server-to-server connections, but it also acts as a Secure Shell client. SSH Tectia ConnectSecure provides enterprise-class secure file transfer services. Note that SSH Tectia ConnectSecure and SSH Tectia Client cannot be installed on the same host.

The following chapters give the basic information about SSH Tectia ConnectSecure.

2.3.1 Architecture

The architecture of SSH Tectia ConnectSecure is similar to the SSH Tectia Client architecture, see Section 2.2.1.

2.3.2 Product Packaging

On Unix, SSH Tectia ConnectSecure is delivered as a bundle of installation packages, a separate bundle for each supported platform. During the installation, you can select the installation packages to implement those functions that best suit the current needs and the environment. The bundles include the following installation packages:

- The *common* package contains auxiliary tools and libraries that are shared by SSH Tectia ConnectSecure and Server. The *common* files are included in the SSH Tectia Client, ConnectSecure, and Server packages, because most Unix packaging systems do not allow the same files to be shared between multiple packages. This also makes it possible to install each of SSH Tectia Client, ConnectSecure, and Server independently on separate hosts.
- The *client* package includes the basic Secure Shell client functionalities, the terminal client and the file transfer client, and the `sshg3`, `scpg3`, and `sftpg3` command-line tools.
- The *client-ft-only* package can be used instead of the *client* package. It includes the other Secure Shell client functions, but not the Secure Shell terminal functionality provided by `sshg3`.
- The *capture* package contains the connection capture component needed for FTP-SFTP conversion and transparent FTP tunneling available with SSH Tectia ConnectSecure.
- The *sdk* package contains the secure file transfer C and Java APIs available with SSH Tectia ConnectSecure.

On Windows platforms, SSH Tectia ConnectSecure is one single installation package, that contains all functionalities, and all necessary tools, libraries, and auxiliary files. You can select the components to be installed from the Installation Wizard.

Table 2.3 summarizes SSH Tectia ConnectSecure installation packages on different platforms.

Table 2.3. The SSH Tectia ConnectSecure installation packages

SSH Tectia ConnectSecure on Unix and Linux	SSH Tectia ConnectSecure on Windows
`common`	`connectsecure`
`client` **or** `client-ft-only`	
`capture`	
`sdk`	

2.3.3 Configuration File and Command-line Tools

The configuration of SSH Tectia ConnectSecure is managed through the Connection Broker which uses an XML-format configuration file `ssh-broker-config.xml`. The configuration file can be edited with an ASCII text editor or an XML editor. You can also use the Connection Broker Configuration GUI to configure the SSH Tectia ConnectSecure functions.

The configuration of SSH Tectia ConnectSecure is similar to the SSH Tectia Client configuration, see Section 2.2.3.

Chapter 3 Planning the Migration

The migration from 4.x version to 6.x can be done manually or with the help of SSH Tectia Manager. This chapter gives advice on planning the migration and lists the general migration steps.

3.1 SSH Tectia Server

The migration consists of the following basic steps:

1. Familiarize yourself with the new software architecture, packaging, and configuration.
2. Old SSH Tectia Server 4.x configurations cannot be used with SSH Tectia Server 6.x. Make new configuration(s) either by using the SSH Tectia Server Configuration GUI or manually in XML. See Chapter 4.
3. Upgrade the SSH Tectia Server software either manually or with SSH Tectia Manager. SSH Tectia Manager can complete the necessary installation steps automatically.

3.1.1 Upgrading SSH Tectia Server on Unix Using Manager

Before the upgrade on Unix, consider the following issues:

- If you earlier had SSH Tectia Server 4.x installed, SSH Tectia Manager will install *common*, *client*, and *server* packages when upgrading to 6.x. This way it attempts to preserve the same functionality as with SSH Tectia Server 4.x. SSH Tectia Server 6.x licenses on Unix include the client functionality.
- If there was an earlier SSH Tectia Client installed, SSH Tectia Manager will install only *common* and *client* packages. After that there will be no server functionality at all. This is different from SSH Tectia Client 4.x where you would also have `sshd2` which took at most two concurrent connections.
- In 6.x, there is only one SSH Tectia Server version that will replace both the administration (A) server and the tunneling (T) server of the 4.x release.

SSH Tectia Manager does not automatically migrate the changes made to the SSH Tectia Server 4.x configuration, but it uses the default 6.x configuration. You can edit the `ssh-server-config.xml` configuration file using a text editor or an XML editor and deploy the configurations using SSH Tectia Manager.

3.1.2 Upgrading SSH Tectia Server on Windows Using Manager

Before the upgrade on Windows, consider the following issues:

- On Windows, the packaging in SSH Tectia Client/Server 4.x was already similar to that in 6.x. SSH Tectia Client and Server come in different packages. So upgrading SSH Tectia Server 4.x to 6.x on Windows will only upgrade the server.
- There is no separate *common* package needed on Windows even though there will now be only one set of the common tools and libraries.

SSH Tectia Manager does not automatically migrate the changes made to the SSH Tectia Server 4.x configuration, but it uses the default 6.x configuration. You can create new configuration(s) using the SSH Tectia Server Configuration GUI and deploy the configurations using SSH Tectia Manager.

3.1.3 Upgrading SSH Tectia Server on Unix Manually

Before the upgrade, consider the following issues:

- To upgrade SSH Tectia Server 4.x (A or T) to 6.x, you have to first uninstall the old version.
- When the old version has been uninstalled, install the packages of the new version.
- To install both client and server functionality, you have to install the *common*, *client*, and *server* packages.
- The *common* package is always installed first and only after that the *client* and/or *server* package(s).

Changes made to the SSH Tectia Server 4.x configuration are not automatically migrated during the upgrade, but the default 6.x configuration is used. You can edit the `ssh-server-config.xml` configuration file using a text editor or an XML editor.

3.1.4 Upgrading SSH Tectia Server on Windows Manually

On Windows, it is possible to upgrade the currently installed software version 4.1.0 or later just by launching the appropriate installer. You do not have to uninstall the previous version first.

Changes made to the SSH Tectia Server 4.x configuration are not automatically migrated during the upgrade, but the default 6.x configuration is used. You can create new configuration(s) using the SSH Tectia Server Configuration GUI.

3.2 SSH Tectia Client

The migration consists of the following basic steps:

1. Familiarize yourself with the new software architecture, packaging, and configuration.

2. The existing connection profiles for SSH Tectia Client on Windows are automatically migrated from 4.x to 6.x during upgrade. The other 4.x configurations cannot be used as such. Make new configuration(s) either manually or with SSH Tectia Manager. See Chapter 5.

3. Upgrade the SSH Tectia Client software either manually or with SSH Tectia Manager. SSH Tectia Manager can complete the necessary installation steps automatically.

4. If you have used SSH Tectia Client 4.x command-line tools in scripts, scheduled file transfers, or remote command execution, check the scripts and command lines for any outdated options, and make the necessary changes. See Section 5.3.

3.2.1 Upgrading SSH Tectia Client on Unix Manually

If SSH Tectia Manager is not used to centrally manage your SSH Tectia installation base, the configuration and installation have to be done manually. On Unix, you have to first uninstall the earlier versions and then install version 6.x (*common* and *client* packages). Files created by the user are not removed during uninstallation and upgrade.

You can edit the `ssh-broker-config.xml` configuration file using a text editor or an XML editor.

3.2.2 Upgrading SSH Tectia Client on Windows Manually

On Windows, upgrading the current installation (4.1.0 or later) to 6.x can be done simply by installing the new version on top of the old one. Files created by the user are not removed during uninstallation and upgrade, but the configuration files need to be updated manually.

The existing connection profiles for SSH Tectia Client on Windows are automatically migrated from 4.x to 6.x during upgrade. The other 4.x configurations cannot be used as such. Create new configuration(s) using the SSH Tectia Configuration GUI.

In SSH Tectia Client 4.x, the command-line tools used a different configuration file (`ssh2_config`) than the GUI client (`global.dat` and `<profile_name>.ssh2`). In SSH Tectia Client 6.x, both the command-line tools and the terminal GUI client use the Connection Broker configuration (`ssh-broker-config.xml`). In addition, the GUI clients use `global.dat` and `<profile_name>.ssh2` for the GUI-related settings.

3.3 SSH Tectia ConnectSecure

SSH Tectia ConnectSecure is a new product in release 6.0, so a direct product upgrade is not applicable, but usually a fresh installation is made. However, SSH Tectia Client and SSH Tectia Connector can be upgraded to SSH Tectia ConnectSecure by purchasing a new license and by installing the software.

Upgrade the SSH Tectia ConnectSecure software either manually or with SSH Tectia Manager. If SSH Tectia Manager is used to perform the upgrade, it automatically upgrades the SSH Tectia Connector to SSH Tectia ConnectSecure.

SSH Tectia Manager can complete the necessary installation steps automatically, but it requests the administrator to select the features that will be present in ConnectSecure.

Note that SSH Tectia ConnectSecure and SSH Tectia Client cannot be installed on the same host. SSH Tectia ConnectSecure and SSH Tectia Server must be of the same release level.

On Unix platforms, except on Solaris, you can upgrade SSH Tectia Client and Connector 5.x to SSH Tectia ConnectSecure 6.x by installing the new version on top of the old one. All 4.x versions on Unix must be uninstalled before any 6.x versions can be installed.

On Windows, SSH Tectia Client 4.0 must be uninstalled before installing SSH Tectia ConnectSecure 6.x, but SSH Tectia Client and Connector 4.1 and later versions can be upgraded by installing the new version on top of the old one.

Chapter 4 SSH Tectia Server Configuration

This chapter describes how the SSH Tectia Server configuration file has changed between releases 4.x and 6.x and how to convert an existing SSH Tectia Server 4.x configuration to the new format.

4.1 File Names and Locations

The file names and locations have changed between SSH Tectia Server releases 4.x and 6.x. The following chapters detail the differences.

4.1.1 Executables

Table 4.1 shows the commands in SSH Tectia Server 4.4 and the corresponding commands in SSH Tectia Server 6.x.

Table 4.1. Server command names in SSH Tectia Server 4.4 and 6.x

SSH Tectia Server 4.4	SSH Tectia Server 6.x
`sshd2` (Unix)	`ssh-server-g3(.exe)`
`ssh2master.exe` (Windows)	
`ssh2admin.exe` (Windows)	`ssh-server-gui.exe` (Windows)
	`ssh-server-config-tool(.exe)`

On Unix, the default directory for server executables is `/opt/tectia/sbin`.

On Windows versions, the default directory for server executables is:

```
"C:\Program Files\SSH Communications Security\SSH Tectia\SSH Tectia Server"
```


4.1.2 Configuration Files

Table 4.2 shows the configuration files in SSH Tectia Server 4.4 and the corresponding files in SSH Tectia Server 6.x.

Table 4.2. Configuration files in SSH Tectia Server 4.4 and 6.x

SSH Tectia Server 4.4	SSH Tectia Server 6.x
`sshd2_config`	`ssh-server-config.xml`
`ssh_certd_config`	
Certificate user mapping file	

On Unix, the default configuration file location is `/etc/ssh2/ssh-server-config.xml`.

On Windows versions, the default configuration file location is "`C:\Program Files\SSH Communications Security\SSH Tectia\SSH Tectia Server\ssh-server-config.xml`".

4.1.3 Server Host Key Files

On Unix platforms, the default host key file locations have not changed and an existing host key pair is automatically used for SSH Tectia Server 6.x.

On Windows, the default host key location has been changed, but existing host key pairs are automatically moved to the new location during the upgrade. However, the permissions on the host key files are different, and the permissions need to be set manually on Windows. In SSH Tectia Server 6.x, the host key file and directory must allow full permissions only for the administrator group and for the SYSTEM account, and no other permissions.

In SSH Tectia Server 6.x, the default host key location on Windows is:

`"C:\Program Files\SSH Communications Security\SSH Tectia\SSH Tectia Server\"`

To set the host key permissions, use the **ssh-keygen-g3** tool in the command prompt:

1. Go to the installation directory of SSH Tectia Server: `"C:\Program Files\SSH Communications Security\SSH Tectia"`
2. Set the permissions for the host key by running command:

```
$ ssh-keygen-g3 --set-hostkey-owner-and-dacl hostkey
```

4.1.4 License File

On Unix, the license file name has changed between SSH Tectia Server releases 4.x and 6.x, but the file location is the same.

In SSH Tectia Server 6.x, the license file is `/etc/ssh2/licenses/sts60.dat`

On Windows, the SSH Tectia Server license file name remains the same but the location of the file has been changed to: "`C:\Program Files\SSH Communications Security\SSH Tectia\SSH Tectia AUX\licenses\sts61.dat`".

On Windows, when installing from the CD-ROM or from an installation file extracted from the online package, the license file is automatically copied to the correct directory. In other cases, the license file has to be copied manually.

4.2 Configuration File Options

SSH Tectia Server 4.x reads its configuration by default from the `/etc/ssh2/sshd2_config` file (on Unix). The configuration file format consists of lines with keyword-value pairs. For example:

```
AllowedAuthentications     publickey,password
```

In version 5.0, the SSH Tectia Server configuration file was changed to XML format, and 6.x uses the same XML configuration file type. The configuration is divided into elements and the order of the configuration elements has an effect on the server behavior. For detailed information on the XML elements, their attributes, and editing the configuration settings, study Chapter *Configuring SSH Tectia Server* in *SSH Tectia Server 6.1 Administrator Manual*.

The name of the configuration file has been changed between 4.x and 6.x. By default on Unix, SSH Tectia Server reads the `/etc/ssh2/ssh-server-config.xml` file. If an element is left empty in the `ssh-server-config.xml` file, the default hardcoded values for it are used.

The file `/etc/ssh2/ssh-server-config-default.xml` shows the default configuration that is used if `/etc/ssh2/ssh-server-config.xml` cannot be found. The defaults have been hardcoded in the server, and cannot be changed by changing the `ssh-server-config-default.xml` file.

On Windows, the configuration file is:

"`C:\Program Files\SSH Communications Security\SSH Tectia\SSH Tectia Server\ssh-server-config.xml`"

And the default settings are shown in:

"`C:\Program Files\SSH Communications Security\SSH Tectia\SSH Tectia Server\ssh-server-config-default.xml`"

4.2.1 Converting `sshd2_config` to `ssh-server-config.xml`

The `ssh-server-config.xml` file used by SSH Tectia Server 6.x is divided into four blocks:

- general server parameters (`params`)
- connection rules (`connections`)
- authentication methods (`authentication-methods`)
- allowed services (`services`)

In the first three blocks, different *selectors* can be used to set access rules to users based on the user parameters such as username or location. Users can be divided into groups dynamically, for example, based on the authentication method they used to log in. In the last block, each group can then be allowed or denied services such as tunneling, file transfer, and terminal access.

The configuration file is read in top-down order during connection setup. If a connection is denied in one of the blocks, the connection setup phase ends immediately and the rest of the configuration settings are not read.

Table 4.3 shows the correspondence between 4.x and 6.x configuration options. The left column lists all configuration options used in the `sshd2_config` file in SSH Tectia Server 4.4. The right column shows how the same thing can be configured in the `ssh-server-config.xml` file in SSH Tectia Server 6.x.

The table only contains a reference to the correct XML element and/or attribute used in the `ssh-server-config.xml` file. The references are presented using the XPath notation. Full description of the configuration can be found in Chapter *Configuring SSH Tectia Server* in *SSH Tectia Server 6.x Administrator Manual*.

Table 4.3. SSH Tectia Server 4.4 and 6.x configuration options comparison

sshd2_config configuration option	**Equivalent option in ssh-server-config.xml**
AllowAgentForwarding or ForwardAgent (Unix only)	services/rule/tunnel-agent[@action="allow"|"deny"] (Unix only)
AllowedAuthentications	authentication-methods/authentication/
AllowGroups (Unix only)	Use authentication-methods/authentication elements with attribute action="allow"|"deny" and selectors.
AllowHosts	Use connections/connection or authentication-methods/authentication elements with attribute action="allow"|"deny" and selectors.
AllowSHosts (Unix only)	.shosts not used anymore in host-based authentication.
AllowTcpForwarding	services/rule/tunnel-local[@action="allow"|"deny"] and services/rule/tunnel-remote[@action="allow"|"deny"]
AllowTcpForwardingForGroups (Unix only)	Use services/group/selector elements and tunnel-local and tunnel-remote elements with group and action attributes.
AllowTcpForwardingForUsers	See AllowTcpForwardingForGroups.
AllowUsers	See AllowGroups.
AllowX11Forwarding or X11Forwarding or ForwardX11 (Unix only)	services/rule/tunnel-x11[@action="allow"|"deny"] (Unix only)
AuthInteractiveFailureTimeout	authentication-methods/authentication/auth-keyboard-interactive[@failure-delay]
AlwaysUsePAMSessionLogging	params/pluggabble-authentication-modules/pam-calls-with-commands; available in 6.0.2 and later
AlwaysUsePAMAccountManagement	params/pluggabble-authentication-modules/pam-calls-with-commands; available in 6.0.2 and later
AuthKbdInt.NumOptional	Exactly the same functionality not available.
AuthKbdInt.Optional	Use authentication-methods/authentication/auth-keyboard-interactive/ and set different submethod child elements on the same level.
AuthKbdInt.Plugin	If SecurID authentication is needed use submethod-securid, otherwise no equivalent functionality available because line-based plugin protocol is not used anymore.
AuthKbdInt.RADIUS.NASIdentifier	authentication-methods/authentication/auth-keyboard-interactive/submethod-radius/radius-server[@client-nas-identifier="identifier"]
AuthKbdInt.RADIUS.Server	authentication-methods/authentication/auth-keyboard-interactive/submethod-radius/radius-server
AuthKbdInt.Required	Use nested authentication elements that have the auth-keyboard-interactive element.

sshd2_config configuration option	**Equivalent option in ssh-server-config.xml**
AuthKbdInt.Retries	authentication-methods/authentication/auth-keyboard-interactive[@max-tries="number_of_tries"]
AuthorizationFile	authentication-methods/authentication/auth-publickey[@authorization-file]
AuthorizedKeysFile	authentication-methods/authentication/auth-publickey[@openssh-authorized-keys-file]
AuthPassword.ChangePlugin (Unix only)	Exactly the same option not available. Instead, password changing can be done using services/group/selector/user-password-change-needed and running the passwd program or similar as a forced command for that group.
AuthPublicKey.Cert.MaxSize	Not available.
AuthPublicKey.Cert.MinSize	Not available.
AuthPublicKey.MaxSize	authentication/selector/publickey-passed[@length="min_size-max_size"]
AuthPublicKey.MinSize	authentication/selector/publickey-passed[@length="min_size-max_size"]
BannerMessageFile	authentication-methods/banner-message[@file="/path/to/banner_file"]
Cert.RSA.Compat.HashScheme	Not available.
CertdListenerPath (Unix only)	Not available. ssh-certd is not used for certificate validation anymore
CheckMail (Unix only)	Not available. Mail is not checked
ChRootGroups (Unix only)	Use services/group/selector and services/rule/subsystem|terminal|command[@chroot]. (Unix only)
ChRootUsers (Unix only)	See ChRootGroups.
Ciphers	connections/connection/cipher[@name="cipher_name"]
DenyGroups (Unix only)	See AllowGroups.
DenyHosts	See AllowHosts.
DenySHosts (Unix only)	Not available because shosts files are not used anymore in host-based authentication.
DenyTcpForwardingForGroups (Unix only)	See AllowTcpForwardingForGroups.
DenyTcpForwardingForUsers	See AllowTcpForwardingForGroups.
DenyUsers	See AllowGroups.
DisableVersionFallback	Not available.

sshd2_config configuration option	**Equivalent option in ssh-server-config.xml**
DoubleBackSpace (Windows only)	Not available.
EventLogFilter (Windows only)	params/logging/log-events with improved functionality
ExternalAuthorizationProgram (Unix only)	Not available.
ForcedPAMAccountManagementPasswordChange	params/pluggabble-authentication-modules/pam-calls-with-commands
ForwardACL	Use services/group/selector elements and tunnel-local and tunnel-remote elements with group and action attributes and src and dst elements.
ForwardAgent (Unix only)	See AllowAgentForwarding.
GSSAPI.AllowedMethods	Not configurable. Only Kerberos is supported.
GSSAPI.AllowOldMethodWhichIsInsecure	Not configurable. The old method (without MIC) is not supported.
GSSAPI.DelegateToken	authentication-methods/authentication/auth-gssapi[@allow-ticket-forwarding="yes"|"no"]
GSSAPI.Dlls	authentication-methods/authentication/auth-gssapi[@dll-path="/full/path/to/libraries"]
HostbasedAuthForceClientHostnameDNSMatch (Unix only)	authentication-methods/authentication/auth-hostbased[@require-dns-match="yes"|"no"]
HostCertificateFile	params/hostkey/x509-certificate[@file="/path/to/hostcert.cer"]
HostKeyEkInitString	params/hostkey/externalkey[@init-info="..."]
HostKeyEkProvider	params/hostkey/externalkey[@type="..."]
HostKeyEkTimeOut	Not available. Default timeout is used.
HostKeyFile	params/hostkey/private[@file="/path/to/hostkey"]
HostSpecificConfig	Use selectors in connections, authentication-methods, and services elements.
IdleTimeOut	services/rule[@idle-timeout="limit_in_seconds"]
IgnoreLoginRestrictions.NISPlusNoPermission	params/settings/ignore-nisplus-no-permission; available on Linux and Solaris platforms in 6.0.2 and later
IgnoreLoginRestrictions.PasswordExpiration	Define services/group/selector/user-password-change-needed for a group and services/rule/command application="/usr/bin/passwd" action="forced". If this entry is omitted from the configuration file, password expiration is ignored.
IgnoreLoginRestrictions.Rlogin.AIX (applicable only on AIX)	params/settings[@ignore-aix-rlogin="yes"|"no"] (applicable only on AIX)

sshd2_config configuration option	**Equivalent option in ssh-server-config.xml**
IgnoreRhosts (Unix only)	.rhosts and .shosts files are not used anymore in host-based authentication.
IgnoreRootRhosts (Unix only)	.rhosts and .shosts files are not used anymore in host-based authentication.
KeepAlive	connections/connection[@tcp-keepalive="yes"|"no"]
ListenAddress	params/listener
LoginGraceTime	authentication-methods[@login-grace-time="limit_in_seconds"]
MACs	connections/connection/mac[@name="mac_name"]
MaxBroadcastsPerSecond	Not available.
MaxConnections	params/limits[@max-connections="connection_limit"]
NoDelay	Not configurable. No delay is always on.
PasswdPath (Unix only)	Password changing can be done using services/group/selector/user-password-change-needed and running the passwd program or similar as a forced command for that group.
PasswordGuesses	authentication-methods/authentication/auth-password[@max-tries="..."]
PermitEmptyPasswords	Not available.
PermitRootLogin	Use authentication-methods/authentication/selector/user-privileged and appropriate action in authentication element
PermitUserTerminal (Windows only)	services/rule/terminal[@action="allow"|"deny"]
Port	params/listener[@port="port_number"]
PrintMotd (Unix only)	services/rule[@print-motd="yes"|"no"] (Unix only)
PrivateWindowStation (Windows only)	Not available.
ProtocolVersionString	Not available.
ProxyServer	params/settings/proxy-scheme
PublicHostKeyFile	params/hostkey/public[@file="/path/to/public_host_key"]
QuietMode	params/logging
RandomSeedFile	Not configurable. Default random seed is used.
RekeyIntervalSeconds	connections/connection/rekey[@seconds="time_in_seconds" bytes="number_of_bytes"]
RequiredAuthentications	Use nested authentication elements.
RequireReverseMapping	Use the fqdn selector.
ResolveClientHostName	params/settings[@resolve-client-hostname="yes"|"no"]
SettableEnvironmentVars	services/rule/environment[@allowed="SETTABLE, ENV, VARS"]
Sftp-AdminDirList (Windows only)	services/rule/subsystem[@type="sftp"]/attribute[@name="virtual-folder" value="dir_list"] (Windows only)

sshd2_config configuration option	**Equivalent option in ssh-server-config.xml**
Sftp-AdminUsers (Windows only)	services/group/selector
Sftp-DirList (Windows only)	services/rule/subsystem[@type="sftp"]/attribute[@name="virtual-folder" value="dir_list"] (Windows only)
Sftp-Home (Windows only)	services/rule/subsystem[@type="sftp"]/attribute[@name="home" value="home_dir"] (Windows only)
SftpLogCategory (Windows only)	params/logging
SftpSysLogFacility (Unix only)	params/logging
SocksServer	params/settings/proxy-scheme
Ssh1Compatibility (Unix only)	Not available.
Sshd1ConfigFile (Unix only)	Not available.
Sshd1Path (Unix only)	Not available.
StrictModes (Unix only)	authentication-methods/auth-file-modes[@strict="yes"|"no"] (Unix only)
StrictModes.User-DirMaskBits (Unix only)	authentication-methods/auth-file-modes[@mask-bits] (Unix only)
Subsystem-<subsystem name>	services/rule/subsystem
SysLogFacility (Unix only)	params/logging
Terminal.AllowUsers	Use services/group/selector elements and services/rule/terminal elements with group and action attributes.
Terminal.DenyUsers	See Terminal.AllowUsers.
Terminal.AllowGroups (Unix only)	See Terminal.AllowUsers.
Terminal.DenyGroups (Unix only)	See Terminal.AllowUsers.
TerminalProvider (Windows only)	ssh_shell.exe executes cmd.exe, this is not currently configurable.
UserConfigDirectory	params/settings[@user-config-dir="..."]
UserKnownHosts (Unix only)	Not available. User's known hosts are not used in host-based authentication.
UserSpecificConfig	Use selectors in authentication-methods and services elements
UseSocks5	params/settings/proxy-scheme
VerboseMode	Not available.
XauthPath (Unix only)	params/settings[@xauth-path="..."] (Unix only)

4.2.2 Converting `ssh_certd_config` to `ssh-server-config.xml`

A separate certificate validation daemon, `ssh-certd`, that handles the server-side certificate validation in a centralized manner for the Secure Shell child servers, is used in SSH Tectia Server versions 4.1 through 4.4. In SSH Tectia Server 4.x, the certificate validation daemon uses its own configuration file, `ssh_certd_config`, and a certificate user mapping file.

In SSH Tectia Server 6.x, the certificate validation configuration is done in the `ssh-server-config.xml` file and no separate configuration files are needed anymore.

Table 4.4 lists all configuration options that are used in `ssh_certd_config` and references to how the same functionality can be configured in `ssh-server-config.xml`.

Table 4.4. Certificate validation daemon configuration options comparison

`ssh_certd_config` configuration option	Equivalent option in `ssh-server-config.xml`
`CertCacheFile`	`params/cert-validation/cert-cache-file[@file="..."]`
`Cert.DODPKI`	`params/cert-validation/dod-pki[@enable="yes"|"no"]`
`CrlAutoUpdate`	`params/cert-validation/crl-auto-update`
`CrlPrefetch`	`params/cert-validation/crl-prefetch`
`ExternalMapper`	Not needed as mapping is very flexible using `authentication-methods/authentication/selector/certificate` elements.
`ExternalMapperTimeout`	Not needed as it is not possible to have an external mapper.
`HostCA`	Not needed as all CA certificates are listed using `params/cert-validation/ca-certificate` elements.
`HostCANoCRLs`	`params/cert-validation/ca-certificate[@disable-crls="yes"|"no"]`
`LdapServers`	`params/cert-validation/ldap-server`
`MapFile`	A separate certificate mapping file is not used anymore. Use the `selector/certificate` elements in `authentication-methods` and `services` elements, instead.
`OCSPResponderURL`	`params/cert-validation/ocsp-responder`
`Pki`	`params/cert-validation/ca-certificate`
`PkiDisableCrls`	`params/cert-validation/ca-certificate[@disable-crls="yes"|"no"]`
`QuietMode`	Not applicable. A separate setting is not needed anymore.
`RandomSeedFile`	Not applicable. A separate setting is not needed anymore.
`SocksServer`	`params/cert-validation[@socks-server-url="..."]` or `params/cert-validation[@http-proxy-url="..."]`
`SysLogFacility`	`params/logging/log-events`
`UseSocks5`	`params/cert-validation[@socks-server-url="socks5://..."]`
`UseSSHD2ConfigFile`	Not applicable. A separate setting is not needed anymore.
`VerboseMode`	Not applicable. A separate setting is not needed anymore.

As mentioned earlier, a separate certificate mapping file is not needed anymore. Mapping the certificate to a user account is done in the `authentication-methods` element of `ssh-server-config.xml` using selectors. Earlier, egrep regular expressions were used when matching fields to a certificate. In SSH Tectia Server 6.x, egrep is no longer used. Instead, easy-to-use general globbing patterns are used.

Chapter 5 SSH Tectia Client and ConnectSecure Configuration

This chapter describes how the SSH Tectia Client configuration file and commands have changed between releases 4.x and 6.x, and how to convert an existing SSH Tectia Client 4.x configuration to the new format.

You can upgrade the SSH Tectia Client 4.x to SSH Tectia Client or SSH Tectia ConnectSecure 6.x. In release 6.x, the configuration principles are the same for both products, so what is said here about SSH Tectia Client applies also to SSH Tectia ConnectSecure, unless otherwise mentioned.

5.1 File Names and Locations

The file names and locations have changed between SSH Tectia Client releases 4.x and 6.x. The following sections detail the differences.

5.1.1 Executables

The following table shows the commands in SSH Tectia Client 4.4 and the corresponding commands in SSH Tectia Client 6.x.

Table 5.1. Client command names in SSH Tectia Client 4.4 and 6.x

SSH Tectia Client 4.4	SSH Tectia Client 6.x
`ssh2(.exe)`	`sshg3(.exe)`
`scp2(.exe)`	`scpg3(.exe)`
`sftp2(.exe)`	`sftpg3(.exe)`
`sshclient.exe` (Windows)	`ssh-client-g3.exe` (Windows)
`ssh-keygen2(.exe)`	`ssh-keygen-g3(.exe)`
`ssh-add2` (Unix)	`ssh-broker-g3(.exe)` and `ssh-broker-ctl(.exe)`
`ssh-agent2` (Unix)	`ssh-broker-g3(.exe)`
`ssh_accession.exe` (Windows)	

On Unix, the default directory for all client executables is `/opt/tectia/bin`.

On Windows, the default directories for executables are:

- "`C:\Program Files\SSH Communications Security\SSH Tectia\SSH Tectia Client`" for the client binaries (e.g. `sshg3.exe`)
- "`C:\Program Files\SSH Communications Security\SSH Tectia\SSH Tectia Broker`" for the Connection Broker binaries
- "`C:\Program Files\SSH Communications Security\SSH Tectia\SSH Tectia AUX`" for auxiliary binaries (e.g. `ssh-keygen-g3.exe`)

5.1.2 Configuration Files

The following table shows the configuration files in SSH Tectia Client 4.4 and the corresponding files in SSH Tectia Client 6.x.

Table 5.2. Configuration files in SSH Tectia Client 4.4 and 6.x

SSH Tectia Client 4.4	SSH Tectia Client 6.x
`ssh2_config`	`ssh-broker-config.xml`
`<profile_name>.ssh2` (Windows)	
`global.dat` (Windows)	`global.dat` (Windows)

On Unix, the default configuration file locations are:

- Factory defaults: `/etc/ssh2/ssh-tectia/auxdata/ssh-broker-ng/ssh-broker-config-default.xml`
- Global configuration: `/etc/ssh2/ssh-broker-config.xml`
- User-specific configuration: `$HOME/.ssh2/ssh-broker-config.xml`

On Windows, the default configuration file locations are:

- Factory defaults: "`C:\Program Files\SSH Communications Security\SSH Tectia\SSH Tectia AUX\ssh-broker-ng\ssh-broker-config-default.xml`".
- Global configuration: "`C:\Program Files\SSH Communications Security\SSH Tectia\SSH Tectia Broker\ssh-broker-config.xml`"
- On Vista, user-specific configuration: "`C:\Users\<username>\AppData\Roaming\SSH\ssh-broker-config.xml`"
- On pre-Vista Windows versions, user-specific configuration: "`C:\Documents and Settings\<username>\Application Data\SSH\ssh-broker-config.xml`"

5.1.3 License File

In 6.x, the SSH Tectia Client license file is `stc61.dat` and the SSH Tectia ConnectSecure license file is `stcs61.dat`

On Unix, the license files are located in directory: `/etc/ssh2/licenses`

On Windows, the license files are located in folder:

`"C:\Program Files\SSH Communications Security\SSH Tectia\SSH Tectia AUX\licenses"`

On Windows, when installing from the CD-ROM or from an extracted installation package, the license file is automatically copied to the correct directory. In other cases, the license file has to be copied manually.

5.2 Configuration File Options

In SSH Tectia Client 6.x, most of the configuration options are part of the Connection Broker configuration.

5.2.1 Command-Line Clients

In SSH Tectia Client 4.x, the command-line clients were configured using the `ssh2_config` file.

In 6.x, the command-line clients do not have a separate configuration file. Instead, the main part of SSH Tectia Client configurations are done in the XML-format Connection Broker configuration file `ssh-broker-config.xml`.

5.2.2 GUI Clients

On Windows, SSH Tectia Client includes graphical user interfaces (GUI) for terminal and secure file transfer. In addition to the Connection Broker configuration (main part of configurations), the GUI clients use also other configurations to define the behavior of the GUI part. The GUI-specific configurations are done using the configuration functionality in the GUI clients. No manual editing of those configuration files is needed.

There is also a graphical user interface for managing and configuring the Connection Broker (**SSH Tectia Configuration** GUI).

5.2.3 The Connection Broker Configuration

The Connection Broker reads three configuration files (if all are available):

1. The `ssh-broker-config-default.xml` file is read first. It holds the factory default settings. It is not recommended to edit the file, but you can use it to view the default settings.

   This file must be available and correctly formatted for the Connection Broker to start.

2. Next, the Connection Broker reads the global configuration file. The settings in the global configuration file override the default settings.

   If the global configuration file is missing or malformed, the Connection Broker will start normally, and will read the user-specific configuration file, instead. A malformed global configuration file is ignored and the default settings or user-specific settings, if they exist, are used instead.

3. Last, the Connection Broker reads the user-specific configuration file, if it is available. The settings in the user-specific configuration file override the settings in the global configuration file, with the following exceptions:

   - The following settings from the user-specific configuration are combined with the settings of the global configuration file:
     - In `general` element, the `key-stores` and `cert-validation` settings
     - In `profiles` element, all settings
     - In `static-tunnels` element, all settings.
   - If a connection profile with the same name has been defined in both the global configuration file and user-specific configuration file, the latter one is used.
   - If the `filter-engine` settings have been defined in the global configuration file, and the file is valid (not malformed), those settings are used, and any `filter-engine` settings made in the user-specific configuration file are ignored.

   If the user-specific configuration file is missing, the Connection Broker will start using the previously read configuration files. However, if a user-specific configuration exists but is malformed, the Connection Broker will not start at all.

The default locations of the configuration files are listed in Section 5.1.2.

5.2.4 Converting `ssh2_config` to `ssh-broker-config.xml`

Table 5.3 shows the differences in the configuration settings. The left column lists all configuration options used in the `ssh2_config` file in SSH Tectia Client 4.4. The right column shows how the same thing can be configured in the `ssh-broker-config.xml` file in SSH Tectia Client 6.x.

The table contains a reference to the correct element and/or attribute used in the `ssh-broker.config.xml` file. Use the table together with Chapter *Configuring Connection Broker* in *SSH Tectia Client 6.x User Manual*, that describes the configuration settings in detail. The references are presented using the XPath notation.

Table 5.3. SSH Tectia Client 4.4 and 6.x configuration options comparison

ssh2_config configuration option	**Equivalent option in ssh-broker-config.xml**
AllowedAuthentications	authentication-methods/authentication-method[@name="..."], can be used in default-settings and profiles/profile.
AuthenticationSuccessMsg	default-settings/authentication-success-message="yes"|"no" **or** profiles/profile/authentication-success-message="yes"|"no"; available in 6.0.2 and later
AuthPassword.AllowFromCommandLine	Not configurable. Passwords from command line are always allowed (but not recommended).
BatchMode	Can be specified using the -B/--batch-mode command-line option with sshg3, scpg3, and sftpg3.
Cert.DODPKI	general/cert-validation/dod-pki[@enable="yes"|"no"]
Cert.EndpointIdentityCheck	general/cert-validation[@end-point-identity-check="yes"|"no"]
Cert.RSA.Compat.HashScheme	Not available.
Ciphers	ciphers/cipher[@name="..."], can be used in default-settings and profiles/profile.
ClearAllForwardings	Not available.
Compression	compression[@name="..." @level="0..9"], can be used in default-settings and profiles/profile.
DebugLogFile	Not available. Instead, when enabling debug with the -D command-line option, it is possible to redirect the debug output to file using option -l <filename>.
DefaultDomain	Not available.
DisableVersionFallback	Not available. There should be no need to disable possible compatibility code.
DontReadStdin	Can be specified using the -n command-line option with sshg3. (Unix only)
EkInitString	general/key-stores/key-store[@init="..."]
EkProvider	general/key-stores/key-store[@type="..."]
EscapeChar	Can be specified using the -e/--escape-char command-line option with sshg3.
ForcePTTYAllocation	Can be forced using the -t/--tty command-line option with sshg3.
ForwardAgent	forwards/forward[@type="agent" @state="on"|"off"|"denied"], with the -a/+a command-line option in sshg3 it is possible to disable/enable agent forwarding. If state="denied", cannot be enabled using +a.
ForwardX11	forwards/forward[@type="x11" @state="on"|"off"|"denied"], with the -x/+x command-line option in sshg3 it is possible to disable/enable x11 forwarding. If state="denied", cannot be enabled using +x.

ssh2_config configuration option	**Equivalent option in ssh-broker-config.xml**
GatewayPorts	profiles/profile/tunnels/local-tunnel[@allow-relay="yes"|"no"]
GoBackground	Can be specified using the -f command-line option with sshg3.
GSSAPI.AllowedMethods	Not configurable. Only Kerberos is supported.
GSSAPI.AllowOldMethod-WhichIsInsecure	Not configurable. The old method (without MIC) is not supported.
GSSAPI.DelegateToken	Not available.
GSSAPI.Dlls	Not configurable. DLLs are searched from default locations.
Host	profile[@host="host_address"]
HostCA	general/cert-validation/ca-certificate
HostCANoCRLs	general/cert-validation/ca-certificate[@disable-crls="yes"|"no"]
IdentityFile	Can be specified with general/key-stores/identification[@file="filename"], **or** profiles/profile/authentication-methods/user-identities/identity[@identity-file="filename"].
IdentityKeyFile	Keys can be specified with general/key-stores/key-store[@type="software" @init="key_files(path_to_key)"], **or** profiles/profile/authentication-methods/user-identities/identity[@file="keyfile"].
KeepAlive	default-settings/keepalive-interval="*seconds*" **or** profiles/profile/keepalive-interval="*seconds*"
LdapServers	general/cert-validation/ldap-server
LocalForward	profile/tunnels/local-tunnel
MACs	macs/mac[@name="..."], can be used in default-settings and profiles/profile.
NoDelay	Not configurable. No delay is always on.
NumberofPasswordPrompts	Not available. There should be no need to set number of password prompts on client side.
OCSPResponderURL	general/cert-validation/ocsp-responder
PasswordPrompt	Not available.
Port	profile[@port="port_number"]
ProxyServer	proxy[@ruleset="..."], can be used in default-settings and profiles/profile. For certificate validation, proxy settings are configured using general/cert-validation[@http-proxy-url="..." @socks-server-url="..."].
QuietMode	Can be specified using the -q command-line option with scpg3.
RandomSeedFile	Not configurable. Default random seed is used.
RekeyIntervalSeconds	rekey[@bytes="number_of_bytes"], can be used in default-settings and profiles/profile.

ssh2_config configuration option	**Equivalent option in ssh-broker-config.xml**
RemoteForward	profile/tunnels/remote-tunnel
SetRemoteEnv	remote-environment/environment, can be used in default-settings and profiles/profile.
Ssh1AgentCompatibility	SSH1 compatibility is not supported.
Ssh1Compatibility	SSH1 compatibility is not supported.
Ssh1InternalEmulation	SSH1 compatibility is not supported.
Ssh1MaskPasswordLength	SSH1 compatibility is not supported.
Ssh1Path	SSH1 compatibility is not supported.
SocksServer	See ProxyServer
StrictHostKeyChecking	server-authentication-methods/auth-server-pub-lickey[@policy="strict"], can be used in default-settings and profiles/profile; available in 6.1.4 and later.
StrictModes	Not configurable.
StrictModes.UserDirMaskBits	Not configurable.
TcpConnectionTimeout	default-settings/tcp-connect-timeout[@time="*seconds*"] and profile/profile/tcp-connect-timeout[@time="*seconds*"]
TrustX11Applications	Not available.
User	profile[@user="user_name"]
UserConfigDirectory	Not configurable.
UseSocks5	Configured directly in proxy[@ruleset="..."] in default-settings and profiles/profile. For certificate validation use cert-validation[@socks-server-url="socks5://..."]
VerboseMode	Can be specified using the -v command-line option with sshg3 and scpg3.
XauthPath	Not available.

5.3 Command-Line Options for Command-Line Clients

The names of the command-line clients have changed between releases 4.x and 6.x. In version 4.x, the command-line clients were ssh2, sftp2, and scp2. In version 6.x, they are called sshg3, sftpg3, and scpg3. However, the old command names can be used with SSH Tectia Client and ConnectSecure 6.x normally. For example, running ssh2 automatically uses sshg3.

Some of the available command-line options have changed between releases 4.x and 6.x. These changes might affect scripts and batch jobs. Check all scripts and batch jobs that use ssh2, sftp2, or scp2 and if they use any outdated options, modify them to use the new command-line options, instead.

The following tables compare the command-line options between versions 4.4 and 6.x. Only Unix command-line tools of 4.4 have been used in the below comparison. In 4.x, Windows command-line tools had slightly different options.

Table 5.4. `ssh2` and `sshg3` command-line options comparison

`ssh2` command-line option	**Equivalent `sshg3` command-line option**
`-1[ti]`	Not available, SSH1 protocol is not supported.
`-4`	Not available, IPv4 is always used.
`-6`	Not available, IPv6 not supported.
`-a`	`-a` or `--no-agent-forwarding`
`+a`	`+a`
`-c` *`cipher`*	`-c` *`cipher`*
`-C`	`-C`
`+C`	`+C`
`-D` *`debug_level`*	`-D` *`debug_level`*
`-e` *`escape_char`*	`-e` *`escape_char`*
`-E` *`provider`*	Not available, must be set in the Connection Broker configuration.
`-f[o]`	`-f` or `--fork-into-background`, argument `'o'` not available
`-F` *`file`*	Not available.
`-g`	`-g` or `--gateway`
`+g`	`+g`
`-h`	`-h` or `--help`
`-i` *`file`*	`-i` *`file`*
`-I` *`initstring`*	Not available, must be set in the Connection Broker configuration.
`-K` *`keyfile`*	`-K` or `--identity-key-file=`*`FILE`*
`-l` *`username`*	`-l` *`username`*
`-L` *`listen-port:dst-host:dst-port`*	`-L` *`listen-port:dst-host:dst-port`*
`-m` *`mac`*	`-m` or `--macs=`*`LIST`*
`-n`	Not available.
`-o` *`option`*	`-o` *`ForwardAgent`*`|`*`ForwardX11`*`|`*`AllowedAuthentications`*
`-p` *`port`*	`-p` *`port`*
`-q`	`-q`
`-R` *`listen-port:dst-host:dst-port`*	`-R` or `--remotefwd` *`listen-port:dst-host:dst-port`*
`-s`	`-s` or `--subsystem`
`-S`	`-S` or `--no-session-channel`
`-t`	`-t` or `--tty`
`-v`	`-v` or `--verbose`
`-V`	`-V` or `--version`
`-x`	`-x` or `-X` or `--no-x11-forwarding`
`+x`	`+x, +X`

ssh2 command-line option	**Equivalent sshg3 command-line option**
`--password=`*`PASSWORD`*| `file://PASSWORDFILE` (this required also `BatchMode=yes`)	`--password=`*`PASSWORD`*| `file://`*`PASSWORDFILE`*| `extprog://`*`program`*
	New command-line options in sshg3
	`-B` or `--batch-mode`
	`-w`
	`+w`
	`-z` *`broker_log_file`*
	`--abort-on-failing-tunnel`
	`--allowed-authentications=`*`methods`*
	`--compressions=`*`methods`*
	`--exclusive`
	`--identity=`*`id`*
	`--identity-key-id=`*`id`*
	`--identity-key-hash=`*`id`*
	`--keep-alive=`*`VALUE`*
	`--remote-environment-name=`*`value`*
	`--remote-environment-format=`*`value`*
	`--tcp-connect-timeout=`*`VALUE`*

Table 5.5. scp2 and scpg3 command-line options comparison

scp2 command-line option	Equivalent scpg3 command-line option
`-1`	Not available, SSH1 is not supported.
`-4`	Not available, IPv4 is always used.
`-6`	Not available, IPv6 not supported.
`-a[arg]`	`-a[arg]`
`-b` *`buffer_size`*	`-b` *`buffer_size_bytes`*
`-B`	`-B` or `--batch-mode`
`-c` *`cipher`*	`--ciphers=`*`LIST`*
`-d`	`-d`
`-D` *`debug_level`*	`-D` *`debug_level`* `--debug=`*`"level"`*
`-h`	`-h` or `--help`
`-I`	`-I` or `--interactive`
`-N` *`max_requests`*	`-N` *`max_requests`*
`-o` *`option_to_ssh2`*	Not available. `scpg3` uses the Connection Broker.
`-p`	`-p`
`-P` *`port`*	`-P` *`port`*
`-q`	`-q`
`-Q`	`-Q`
`-r`	`-r`
`-S` *`ssh2_path`*	Not available, this option is not needed as `scpg3` uses the Connection Broker.
`-u`	`-u` or `--unlink-source`
`-v`	`-v` or `--verbose`
`-V`	`-V` or `--version`
`-W`	`-W` or `--whole-file`
`--checksum[=no]`	`--checksum[=yes|no|md5|sha1|md5-force|sha1-force|checkpoint]`
`--force-lower-case`	`--force-lower-case`
`--overwrite[=no]`	`--overwrite[=yes|no]`
	New command-line options in scpg3
	`--append`
	`--binary`
	`-C`
	`+C`
	`-c` or `--compressions=`*`METHODS`*
	`-i` *`FILE`*
	`-K` or `--identity-key-file=`*`FILE`*
	`--allowed-authentications=`*`METHODS`*
	`--checkpoint=s<seconds>`
	`--checkpoint=b<bytes>`

scp2 command-line option	Equivalent scpg3 command-line option
	`--dst-site=`*PARAM*
	`--exclusive`
	`--fips`
	`--identity=`*ID*
	`--identity-key-hash=`*ID*
	`--identity-key-id=`*ID*
	`--keep-alive=`*VALUE*
	`-m fileperm [:dirperm]`
	`--macs=`*LIST*
	`-O` or `--offset=r<offset>|w<offset>|l<length>|t<length>`
	`--password=`*PASSWORD*| `file://`*PASSWORDFILE*| `extprog://`*PROGRAM*
	`--plugin-path=`*PATH*
	`--prefix=`*PREFIX*
	`--src-site=`*PARAM*
	`--statistics[=yes|no|simple]`
	`--streaming[=yes|no|force|ext]`
	`--tcp-connect-timeout=`*VALUE*

Table 5.6. `sftp2` and `sftpg3` command-line options comparison

`sftp2` command-line option	**Equivalent `sftpg3` command-line option**
`-4`	Not available, IPv4 is always used.
`-6`	Not available, IPv6 not supported.
`-b` *`buffer_size`*	`-b` *`buffer_size_bytes`*
`-B` *`batch_file`*	`-B` *`batch_file`*
`-c` *`cipher`*	`--ciphers=`*`LIST`*
`-D` *`debug_level`*	`-D` *`debug_level`*`--debug=`*`"level"`*
`-h`	`-h` or `--help`
`-m` *`mac`*	`--macs=`*`LIST`*
`-N` *`max_requests`*	`-N` *`max_requests`*
`-o` *`option_to_ssh2`*	Not available. `sftpg3` uses the Connection Broker.
`-P` *`port`*	`-P` *`port`*
`-S` *`ssh2_path`*	Not available, this option is not needed as `sftpg3` uses the Connection Broker.
`-v`	`-v` or `--verbose`
`-V`	`-V` or `--version`
	New command-line options in `sftpg3`
	`-C`
	`+C`
	`-c` or `--compressions=`*`METHODS`*
	`-i` *`FILE`*
	`-K` or `--identity-key-file=`*`FILE`*
	`--allowed-authentications=`*`METHODS`*
	`--exclusive`
	`--fips`
	`--identity=`*`ID`*
	`--identity-key-hash=`*`ID`*
	`--identity-key-id=`*`ID`*
	`--keep-alive=`*`VALUE`*
	`--max-depth=`*`VALUE`*
	`--password=`*`PASSWORD`*| `file://`*`PASSWORDFILE`*| `extprog://`*`PROGRAM`*
	`--plugin-path=`*`PATH`*
	`--sftpg3-mode`[=tectia|ftp|openssh]
	`--statistics`[=yes|no|simple]
	`--tcp-connect-timeout=`*`VALUE`*

Appendix A Example Configuration

This appendix contains an example of SSH Tectia Client and SSH Tectia Server 4.x configurations and how they are converted to SSH Tectia Client and SSH Tectia Server 6.x configurations.

A.1 Client Example

An example `ssh-broker-config.xml` file for SSH Tectia Client 6.x is shown below. The file has been annotated to show how the same settings were done in the `ssh2_config` file for SSH Tectia Client 4.4.

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE secsh-broker SYSTEM "ssh-broker-ng-config-1.dtd">

<secsh-broker version="1.0" >

<!-- General settings. -->
  <general>
    <crypto-lib mode="standard" />
    <cert-validation end-point-identity-check="YES" >
      <dod-pki enable="no"/>
    </cert-validation>
    <key-stores>
    </key-stores>
    <strict-host-key-checking enable="no"/>
    <host-key-always-ask enable="no"/>
  </general>

<!-- Setting default configuration.
This is similar to the following settings in ssh2_config for
SSH Tectia Client 4.4:

.*:
RekeyIntervalSeconds   3600
Ciphers                aes192-cbc,3des-cbc,blowfish-cbc,twofish192-cbc,arcfour
MACs                   hmac-sha1,hmac-md5
AllowedAuthentications publickey,keyboard-interactive,password
ProxyServer            socks://socks.server.com:1080
-->
```



```
 <default-settings>

    <rekey bytes="1000000000" />

    <ciphers>
      <cipher name="aes192-cbc" />
      <cipher name="3des-cbc" />
      <cipher name="blowfish-cbc" />
      <cipher name="twofish192-cbc" />
      <cipher name="arcfour" />
    </ciphers>

    <macs>
      <mac name="hmac-sha1" />
      <mac name="hmac-md5" />
    </macs>

    <authentication-methods>
      <authentication-method name="publickey" />
      <authentication-method name="keyboard-interactive" />
      <authentication-method name="password" />
    </authentication-methods>

    <proxy ruleset="socks4://socks.server.com:1080" />

    <forwards>
    </forwards>

 </default-settings>

<!-- Profile for host tunnel.example.com
which creates a few local tunnels,
allows only public-key authentication,
enables only blowfish-cbc/hmac-sha1.
X11 forwarding is disabled.

This is equivalent to the following settings
in ssh2_config for SSH Tectia Client 4.4:

Host    tunnel.example.com

LocalForward   25:smtp.example.com:25
LocalForward   143:imap.example.com:143
LocalForward   110:pop3.example.com:110

AllowedAuthentications  publickey
Ciphers    blowfish-cbc
MACs    hmac-sha1
ForwardX11   no
-->
```



```
  <profiles>
    <profile name="tunnel"
             id="id2"
             host="tunnel.example.com"
             port="22"
             user="%USERNAME%">
      <hostkey>
      </hostkey>

      <ciphers>
        <cipher name="blowfish-cbc" />
      </ciphers>

      <macs>
        <mac name="hmac-sha1" />
      </macs>

      <authentication-methods>
        <authentication-method name="publickey" />
      </authentication-methods>

      <compression name="none" />
      <forwards>
        <forward type="x11" state="off"/>
      </forwards>

      <tunnels>
        <local-tunnel type="TCP"
                      listen-port="25"
                      dst-host="smtp.example.com"
                      dst-port="25"
                      allow-relay="NO"/>
        <local-tunnel type="TCP"
                      listen-port="143"
                      dst-host="imap.example.com"
                      dst-port="143"
                      allow-relay="NO"/>
        <local-tunnel type="TCP"
                      listen-port="110"
                      dst-host="pop3.example.com"
                      dst-port="110"
                      allow-relay="NO"/>
      </tunnels>

    </profile>

  </profiles>

</secsh-broker>
```


A.2 Server Example

An example `ssh-server-config.xml` file for SSH Tectia Server 6.x is shown below. The file has been annotated to show how the same settings were done in the `sshd2_config` file for SSH Tectia Server 4.4.

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE secsh-server SYSTEM "ssh-server-ng-config-1.dtd">
<secsh-server>


<!-- Set parameters.
These are equivalent to the following settings in sshd2_config for
SSH Tectia Server 4.4:

FIPSMode                no
Port                    22
HostKeyFile             my_hostkey
PublicHostKeyFile       my_hostkey.pub
-->

  <params>
    <crypto-lib mode="standard" />
    <hostkey>
      <private file="/etc/ssh2/my_hostkey" />
      <public file="/etc/ssh2/my_hostkey.pub" />
    </hostkey>
    <listener id="sshdg3" port="22" />
  </params>


<!-- Allow all connections through the connections element and allow
ciphers and macs.

This is equivalent to the following setting in sshd2_config for
SSH Tectia Server 4.4:

# AllowHosts
Ciphers   blowfish-cbc,3des-cbc
MACs      hmac-md5,hmac-sha1
-->

  <connections>
    <connection action="allow">
      <cipher name="blowfish-cbc" />
      <cipher name="3des-cbc" />
      <mac name="hmac-md5" />
      <mac name="hmac-sha1" />
    </connection>
  </connections>
```

```
<!-- In authentication-methods, allow certain users (user1 and guest)
to log in and deny access for all others. This is similar to using
the following setting in sshd2_config for SSH Tectia Server 4.4:

AllowUsers              user1,guest,admin
AllowedAuthentications  password
PasswordGuesses         3
-->

  <authentication-methods>
<!-- In this element user names matching to the selector
     are allowed to login using password authentication.
-->
    <authentication name="allow_certain_users" action="allow">
      <selector>
        <user name="user1,guest,admin" />
      </selector>
      <auth-password max-tries="3" />
    </authentication>

<!-- The users who did not match in the previous authentication
     element are matched here, because empty selector matches
     anything. Access is denied for users who are matched
     in this element.
-->
    <authentication name="deny_the_rest" action="deny">
    </authentication>
  </authentication-methods>

<!--
Users who were allowed to log in the previous elements get
to use services as defined in the services element.
If the services element is empty, the defaults are used.
By default all services are allowed.
-->

  <services>

<!-- On Unix, a forced change of expired passwords is required to prevent
     login with expired passwords. The following selector defines a group
     of users who need to change their passwords.
-->
    <group name="passwd-change">
       <selector>
       <user-password-change-needed />
      </selector>
     </group>

<!-- The following selector defines a group of admin users who will be
     allowed all services.
-->
```



```
    <group name="admin">
      <selector>
       <user name="admin" />
      </selector>
    </group>


<!-- This rule is used to force password change on Unix for the "passwd-
     change" group defined above. The defined services will be denied.
     This is similar to using the default setting in sshd2_config for
     SSH Tectia Server 4.4:
IgnoreLoginRestrictions.PasswordExpiration    no
-->
    <rule group="passwd-change">
      <terminal action="deny" />
      <subsystem type="sftp" application="sft-server-g3" action="deny" />
      <command application="/usr/bin/passwd" action="forced" />
      <tunnel-agent action="deny" />
      <tunnel-x11 action="deny" />
      <tunnel-local action="deny" />
      <tunnel-remote action="deny" />
    </rule>

<!-- The following rule allows all actions to the "admin" group defined
     above. Note that only the listed environment variables are allowed
     and all others denied.
-->
    <rule group="admin" idle-timeout="0">
      <environment allowed-case-sensitive="TERM,PATH,TZ,LANG,LC_*" />
      <terminal action="allow" />
      <subsystem type="sftp" application="sft-server-g3" action="allow" />
      <command action="allow" />
      <tunnel-agent action="allow" />
      <tunnel-x11 action="allow" />
      <tunnel-local action="allow" />
      <tunnel-remote action="allow" />
    </rule>

<!-- The following default rule allows only SFTP access for all other users.
     In this example, this rule will apply to "user1" and "guest" who are
     not included in any group. The "chroot" attribute in the <subsystem/>
     element enforces chrooting to the user's home directory on Unix.
-->
    <rule idle-timeout="0">
      <environment allowed-case-sensitive="TERM,PATH,TZ,LANG,LC_*" />
      <terminal action="deny" />
      <subsystem type="sftp" application="sft-server-g3" action="allow"
                 chroot="/home/%username%" />
      <command action="deny" />
      <tunnel-agent action="deny" />
      <tunnel-x11 action="deny" />
```

```
      <tunnel-local action="deny" />
      <tunnel-remote action="deny" />
    </rule>

  </services>

</secsh-server>
```


Index

Symbols

A

C

D

E

F

H

I

L

P

R

S

T

U

