WPS Link
user guide and reference

Version: 4.2.9

www.worldprogramming.com

Contents

Introduction 3

Setup and Configuration 5
- Configuring WPS on the Client Machine 6

Client User Guide 9
- WPS servers (local and remote) 9
  - Creating a new remote host connection 9
  - Creating a new remote WPS server 14
- Link Explorer 15
- Working Example 19
  - Creating a dataset and program 20
  - Checking the authentication methods 20
  - Creating and starting the remote connection 21
  - Running the program on the remote server 24

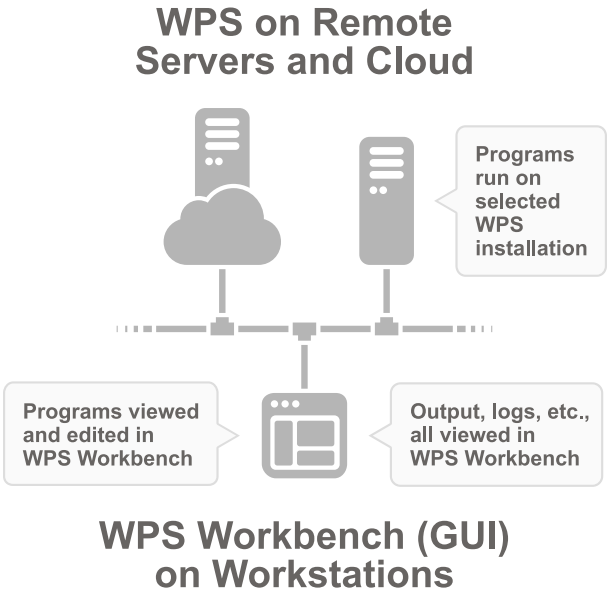
System Administration Guide 26
- SSH (Secure Shell) from a Windows client 26
  - Password authentication (using PuTTY) and WPS sign-on 28
  - Public key authentication 33
- SSH (Secure Shell) from a UNIX client 49
  - Password authentication and WPS sign-on 49
  - Public key authentication 50
- Kerberos single sign-on 59

Environment variables 61

Legal Notices 62

Introduction

WPS Link is the collective term for the technology used to provide a client/server facility. Using **WPS Workbench** on your local PC (the client), you can connect to a remote server on which WPS is installed, in order to run programs written in the language of SAS on that remote server. The resulting output - datasets, logs, and so on - can then all be viewed and manipulated from inside **WPS Workbench** (through the **Output Explorer**, **Results Explorer**, and **WPS Server Explorer**), just as if the work had been executed locally.



WPS Workbench can be installed on a completely different platform from the linked server platform. For example, you might have local **WPS Workbench** installations on Windows workstations linked to multiple Linux servers. **WPS Link** can even be used from **WPS Workbench** on one remote server to link to a further remote server.

Note:

Programs written in the language of SAS require no special language statements to use **WPS Link** technology, although paths may need to be edited to reflect data locations on the remote machines.

Benefits

The benefits of **WPS Link** include:

- Low-powered desktops and laptops can make transparent use of WPS installations on more powerful servers.
- WPS workstations can take advantage of versions of WPS installed on cloud, grid or cluster facilities.

- **WPS Workbench** users can run multiple concurrent programs on multiple local and remote WPS servers.
- Security is enhanced as programs can be developed and run without data leaving the data centre.
- Analytics can be performed on data that has been generated elsewhere, using **WPS Workbench**'s feature-rich local user interface.
- Organisations with distributed facilities can share the resources of remote servers set up in **WPS Workbench**.

Dependencies and usage

WPS Link can only be used with WPS Version 3.1 and above. **WPS Link** operates over an SSH (Secure Shell) connection and provides connectivity between installations of **WPS Workbench** and remote installations of WPS on all common server platforms except z/OS.

Layout of the manual

This manual is divided into the following parts:

- *Setup and Configuration* (page 5).
- The *Client User Guide* (page 9), which is aimed at day-to-day users of WPS Link.
- The *System Administration Guide* (page 26), which is aimed at the IT people who install and administer the system.

Setup and Configuration

There are 3 items required for a **WPS Link** client/server setup: an SSH (Secure Shell) server, the WPS software itself, and sufficient WPS license keys to cover the setup.

SSH server

WPS Link requires the use of an SSH connection between the server and client machines. This is not supplied with the WPS software. However:

- For a UNIX (Linux, Solaris or AIX) server machine, the built-in SSH daemon can be used.
- For a Windows server machine, the third party SSH facility is available separately from Bitvise (refer to the *System Administration Guide* (page 26) for details).

Note:

Bitvise SSH is the only SSH facility officially supported by World Programming for a Windows server machine.

WPS software

WPS is supplied as a single installation file that contains all the WPS features required for use with **WPS Link**, including **WPS Workbench**, **Java Runtime Environment**, and the licensable **WPS Server** component.

Note:

You will need WPS installation files that are suitable for the operating systems on both server and client machines. For example, if you have a Linux server machine and Windows client machines, you will need the Linux installation file for the server, and the appropriate Windows installation files (32-bit or 64-bit) for the clients. Refer to the relevant platform installation guide for full details of the WPS installation process.

WPS licence key(s)

In order for a WPS installation to be able to execute a program written in the language of SAS, the **WPS Server** component needs to be activated by the application of a WPS license key (supplied separately to the WPS software).

- If you want to restrict the execution of programs to the linked server, then you only need a license key for the server installation of WPS.
- If you want to be able to execute programs on either the client machine or the linked server, then you need separate licence keys for the server and the client installations of WPS.

Installation summary

A brief overview of the sequence of steps required to install and set up a **WPS Link** client/server solution is given below.

Important:

The person who installs WPS and applies the license keys must have operating system administrator privileges on those machines.

1. Outside WPS, set up and test the SSH connection between the server and client machines in accordance with the platforms and authentication methods that are active at your site (refer to the *System Administration Guide* (page 26) for details).
2. Install WPS fully on the server machine if you are going to be executing programs on that machine.

   Note:

   Ensure that you have your licence key for the server installation. When you launch WPS on the server, you will automatically be prompted to apply the licence.
3. If you have not already done so, install WPS on the individual client machine(s).

   Note:

   Ensure that you have your workstation licence key, so that it can be applied to each client in the event that you may also wish to execute programs on the client(s) itself. (When you launch WPS on the client(s), you will automatically be prompted to apply the licence.) However, if you are only going to be running programs on the linked server, then you do not require a licensed installation of WPS on the client. In this case, refer to *Configuring WPS on the Client Machine* (page 6).
4. Inside WPS, create a link to the remote computer in the following two-step operation:
   a. Create a connection to the remote host.
   b. Add a WPS server to this remote host so that you can run your programs remotely.

      Note:

      This is fully demonstrated in the *Working Example* (page 19).

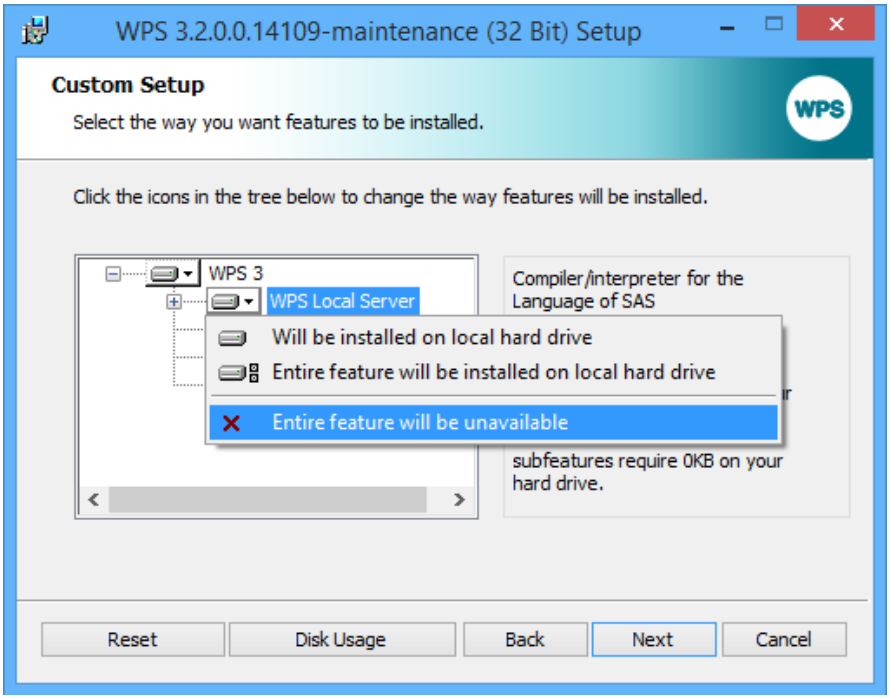
Configuring WPS on the Client Machine

A WPS installation is made up of the following components on Windows: **WPS Server** (which is attached to the licence and includes database engine capabilities), **WPS Workbench**, **Java Runtime Environment** (the version of JRE required for the WPS Workbench), and **Documentation**. Because it is not mandatory to use the **WPS Server** component on the client machine if programs are not being executed on it (in that they are being run only on the linked server), you can isolate the **WPS Server** component from the installation as described below.

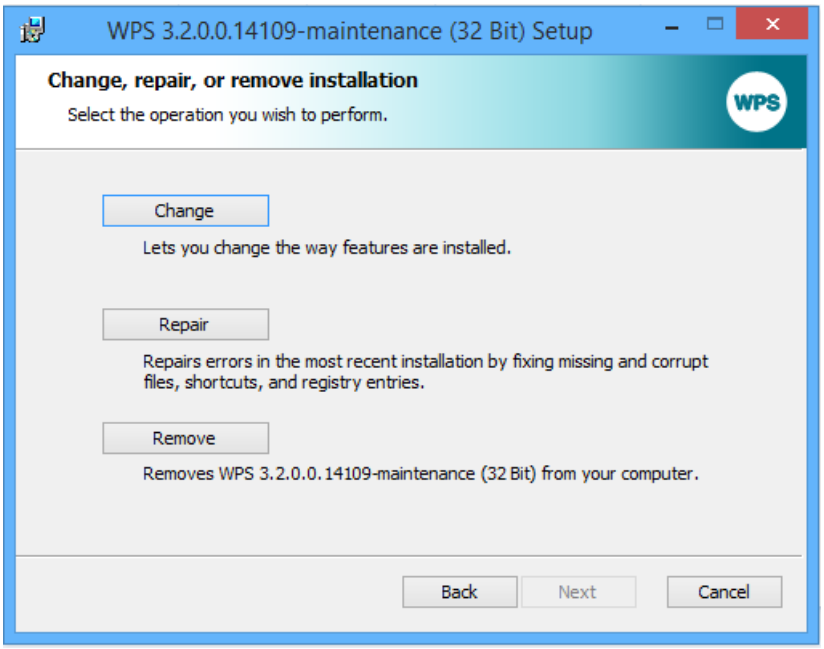
Note:

You can only do this on a workstation that is running on Windows. You will still require the **WPS Workbench** and **Java Runtime Environment** components as a minimum.

- If you have not already installed WPS, then run the relevant `wps msi`, and, when the following **Custom Setup** screen is displayed, deselect **WPS Local Server**, as shown below, before going on to click **Next**:



- If you have already installed WPS, then select **WPS** from the **Programs and Features** within the Windows **Control Panel**, click **Change** at the top of the window, followed by **Change** again in the following window:



  You will then come to the **Custom Setup** shown previously and can deselect **WPS Local Server** as shown, before completing the amended installation.

Caution:

You should be aware that, if you uninstall **WPS Local Server** from a machine using the step immediately above, your WPS installation will no longer be licensed on that machine.

Note:

If you are using a non-Windows platform, and wish to run programs on a remote server only, then you can go into *Link Explorer* (page 15) in **WPS Workbench** and either **Stop** or **Delete** the **Local Server**, without your licence being affected.

Client User Guide

WPS servers (local and remote)

In order to run a program written in the language of SAS, **WPS Workbench** requires a connection to a *WPS server*. The term *server* here is used to refer to a server process. The process may be running on the local workstation machine (a local host connection), or on an installation of WPS on a remote machine (a remote host connection). Servers running under the local connection are termed *local servers*, whilst servers running under a remote connection are termed *remote servers*.

With a full workstation installation of WPS there will be a single host connection called `Local` and a server called `Local Server`. This server will be started by default when **WPS Workbench** is started.

The term *remote server* refers to a WPS server running under a host connection on a remote machine. Multiple servers can run under a single remote host connection. Once a connection to a remote machine is authenticated, one or more WPS servers can be started without the need for further user authentication. Multiple users can have remote connections configured to the same machine, with the only restriction being the resources on that machine. For the creation of a new connection to a remote server, refer to *Creating a new remote host connection* (page 9).

The list of defined servers is stored within the workspace, and so changing your workspace will load a new list of servers, which may be different. The servers are maintained using *Link Explorer* (page 15).

For convenience, when moving between workspaces, or to share server definitions within a work group, server definitions can be exported to an external file, and imported from a definition file. For further information, refer to the **WPS Workbench** user guide.

The **WPS Server Explorer** shows the list of servers currently running in **WPS Workbench**, and the *Link Explorer* (page 15) shows the list of both connections and servers. Through the latter, you can define new connections, new servers, edit their properties, delete connections and servers, and export and import connections and servers.

Creating a new remote host connection

You will need to have access to a machine that has a licensed **WPS Local Server** installation on a supported platform. You will also need to ensure that you have SSH access to the machine, and that you know the location of the WPS installation on that machine. You may need to contact your system administrator to obtain this information and ensure that you have access.

WPS Link supports a variety of means of SSH authentication, including simple password sign-on and public key authentication. The available methods, and any additional software required, are all described in the *System Administration Guide* (page 26). Before creating a connection, you should verify that the required authentication methods are selected in **WPS Workbench**.

Note:

While you are learning about **WPS Link** and are not yet running it in secure production environments, you may wish to choose simple password sign-on as your authentication method. Subsequently, your system administrator may consider other methods such as public key authentication which might be more appropriate to your business needs. You do not need any extra software installed for this. In particular, you do not need a separate SSH client installation. It can, however, be useful to have an SSH client installed, in order to verify independently that you can indeed connect to the remote machine using SSH, for example PuTTY, which is available for download at *http://www.chiark.greenend.org.uk/%7Esgtatham/putty/download.html*.

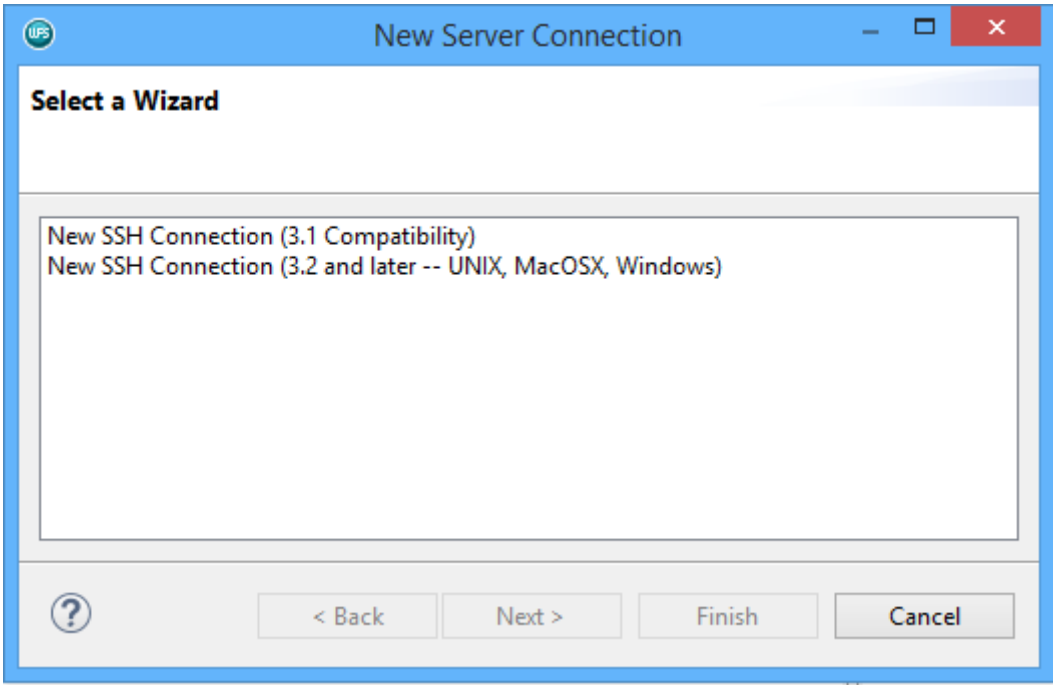
Note:

If you already have a private key, and wish to apply it, then, before starting, load it in Workbench, as described in the introduction to **SSH from a Windows client** in the *System Administration Guide* (page 26).

To create a new remote host connection, open **WPS Workbench**, and then:

1. Open the Remote Host Connection Wizard by doing either of the following:
   - Ensure that the *Link Explorer* (page 15) is visible, and click on the **New Remote Connection**  button.
   - From the main menu, select **WPS ➤ Links ➤ New Remote Host Connection**.

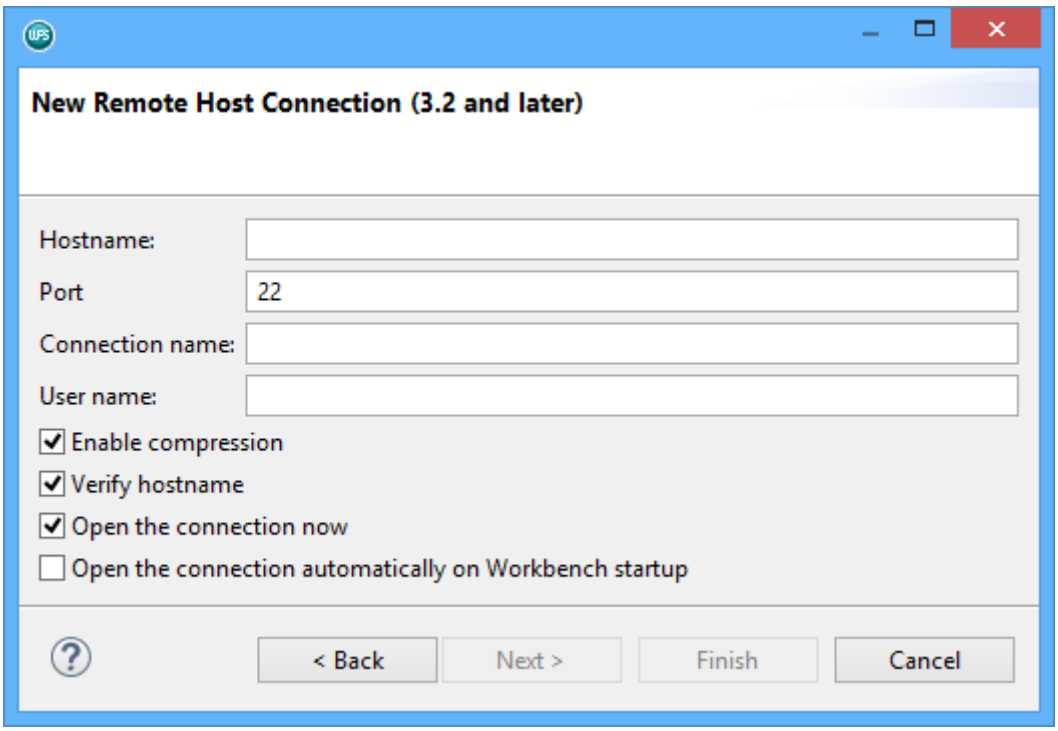
2. In the **New Server Connection** dialog, choose one of:



- **New SSH Connection (3.1 Compatibility)**
- **New SSH Connection (3.2 and later -- UNIX, MacOSX, Windows)**

The first of the above options offers backwards compatibility, having identical behaviour to that present in WPS version 3.1. The second option includes support for establishing remote connections to Windows servers running **Bitvise SSH Server**. Regardless of whether or not you require Bitvise support, you should choose this option if there is no special reason not to. You should be aware that, if you select **3.2 and later**, and then revert to WPS Version 3.1, you will lose the remote host connections that were set up in WPS Version 3.2.

3. Click **Next** and complete the following dialog:

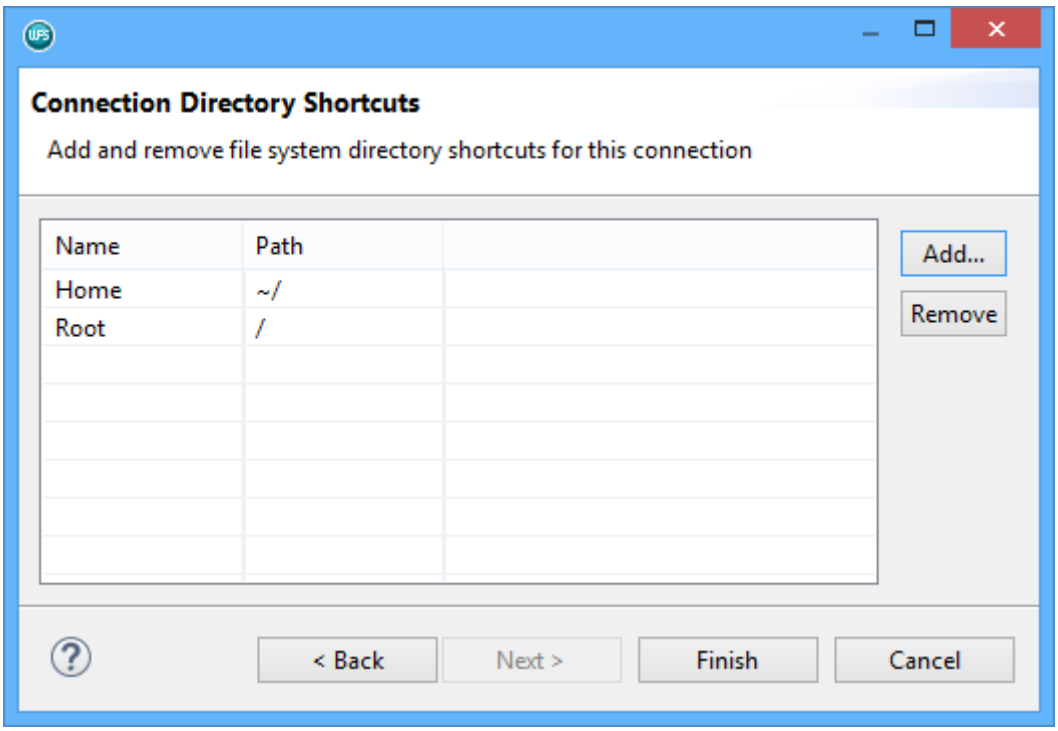


- For **Hostname**, enter the name or IP address of the remote machine to which you wish to connect.
- For **Port**, you can specify a port other than the default **22** if required, but, if so, you should verify it first with your system administrator.
- The **Connection name** defaults to the **Hostname**, but you can overwrite this to enter a descriptive name for the connection - for example, **wps-link-remote**. Remember that connection names must be unique, so, if you are creating multiple connections to the same host, you will need to modify the connection name each time.
- For **User name**, enter your user ID on the remote host.
- In the last two check boxes, ensure that **Open the connection now** is ticked if the connection is to be opened now, and that **Open the connection automatically on Workbench startup** is ticked if the connection is to be opened automatically whenever **WPS Workbench** is started.

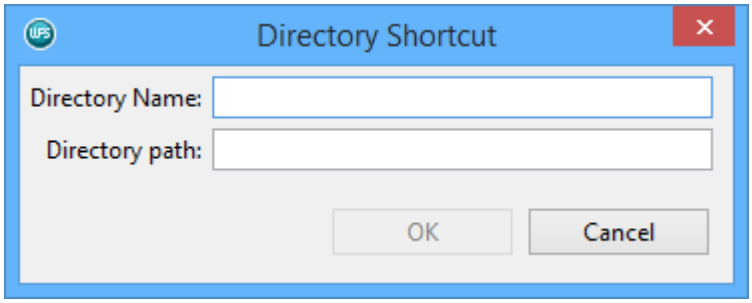
Tip:

If you are going to be using the connection regularly, then you are strongly advised to select **Open the connection automatically on Workbench startup** so that, once the associated server starts, the connection is already open.

4. Press **Next** to define the connection's directory shortcuts:

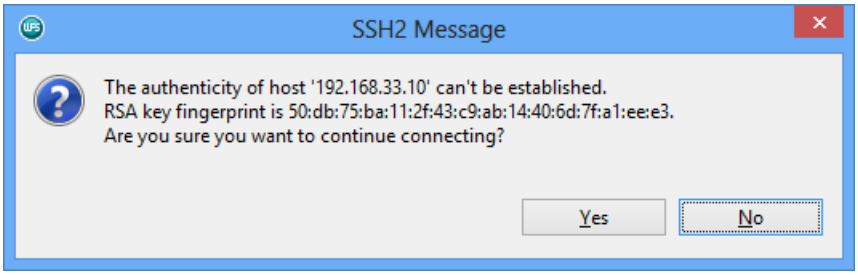


A directory shortcut is a shortcut to a path on the host's file system. By default, there are directory shortcuts defined for Home (~/) and Root (/). To set up a new shortcut, click **Add** to display the following dialog:



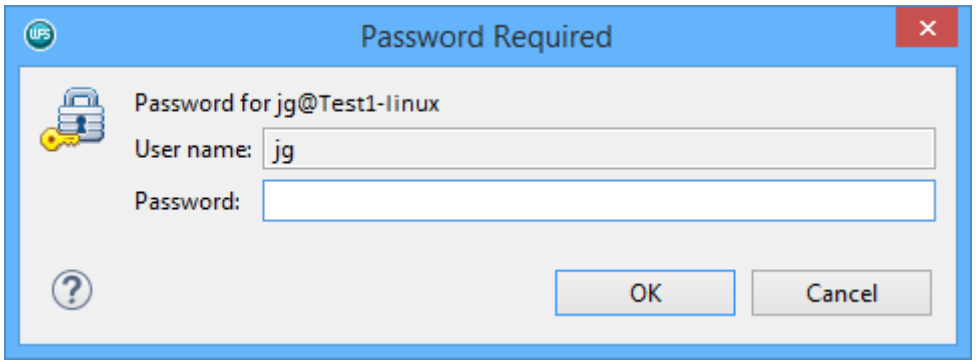
After **Directory Name**, enter the shortcut name you want to use, and, after **Directory Path**, the full path (as shown when you hover over the location in **File Explorer**), and click **OK**. The shortcut name then appears in the host file system in **File Explorer**. You can double-click a shortcut to change its definition.

5. Click **Finish**. If you have not previously validated the authenticity of the remote host, an **SSH2 Message** dialog of the following kind shown is displayed:



   If you are certain that the identity of the host is correct, click **Yes**. If you are setting this up in a production environment, you should explicitly confirm with a system administrator that the displayed RSA key fingerprint of the remote host matches that of the host to which you intend to connect. Once the host key has been confirmed, it is stored in the **WPS Workbench** and subsequent attempts to connect to the same host machine will not require this to be reconfirmed.

6. If you elected to open the connection immediately, you will now be prompted for a password to authenticate yourself on the host machine. Assuming that your system administrator has provided you with the required password, enter it in the **Password Required** dialog and click **OK**.

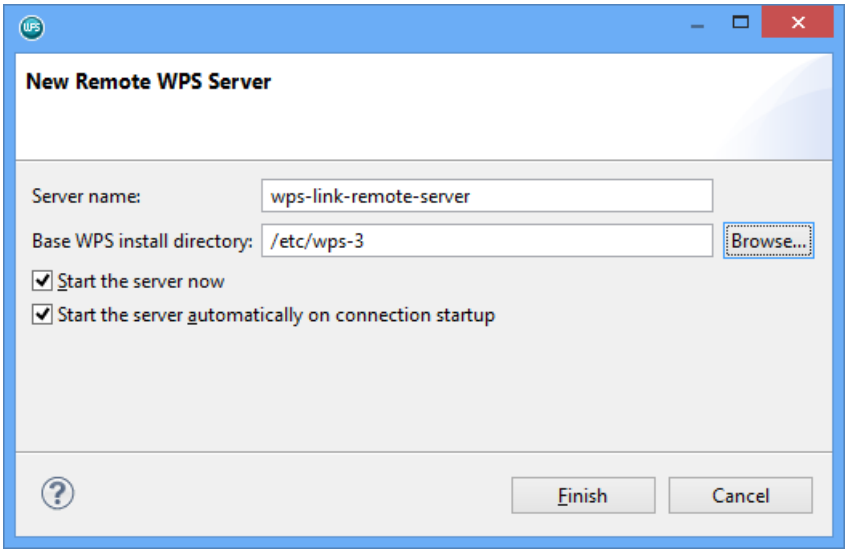


There will now be an entry in the *Link Explorer* (page 15) for your new connection. You will now need to add WPS server definitions to this connection in order to use WPS. This is covered in *Creating a new remote WPS server* (page 14).

Creating a new remote WPS server

As explained in *WPS servers (local and remote)* (page 9), WPS Workbench allows you to have multiple remote WPS servers defined on a remote host connection.

1. Having ensured that your connection is open, open the Remote WPS Server Wizard by doing one of the following:
   - Ensure that *Link Explorer* (page 15) is visible, and right-click on the host connection to which you wish to add a server, and click on the **New WPS Server** option in the context menu.
   - From the main menu, choose **WPS ➤ Links ➤ New WPS Server ➤ *Host Connection Name***.
2. In the resulting **New Remote WPS Server** dialog, enter the **Server name** (this must be unique across all connections), and, in the **Base WPS install directory** field, specify the location of the remote WPS installation:



   In the last two check boxes, ensure that **Start the server now** is ticked if the server is to be started immediately, and that **Start the server automatically on connection startup** is ticked if the server is to be started each time the connection is opened.
3. Click **Finish** to complete the wizard and create the new server definition.

There will now be an entry in the *Link Explorer* (page 15) for your new WPS server connection, and also in the **WPS Server Explorer** (for this view, refer to the **WPS Workbench** user guide).

Link Explorer

The **Link Explorer** allows you to manage the host connections that you have configured, and also any WPS servers on those connections.

To Display this View

To add this view to the current perspective, select **Window ➤ Show View ➤ Link Explorer**.

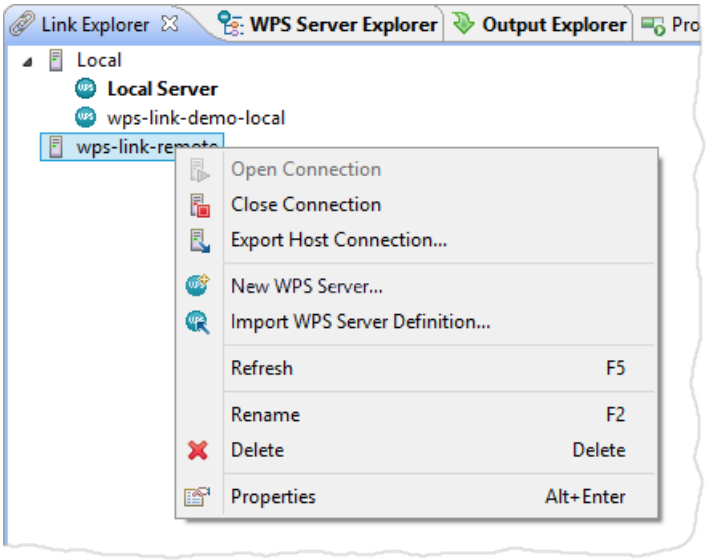
Objects Displayed

There are only two node types visible in this view:

- **Connection node:** The root node that represents a connection to a host machine. One or more servers can be running on a connection. The connection can either be a local connection or a connection to a remote machine. There can only be one local connection, but many remote connections.
- **Server node:** This node represents the WPS installation where you are running your programs.

Managing the Connections and Servers

If you right-click on an open connection, the following options are displayed:

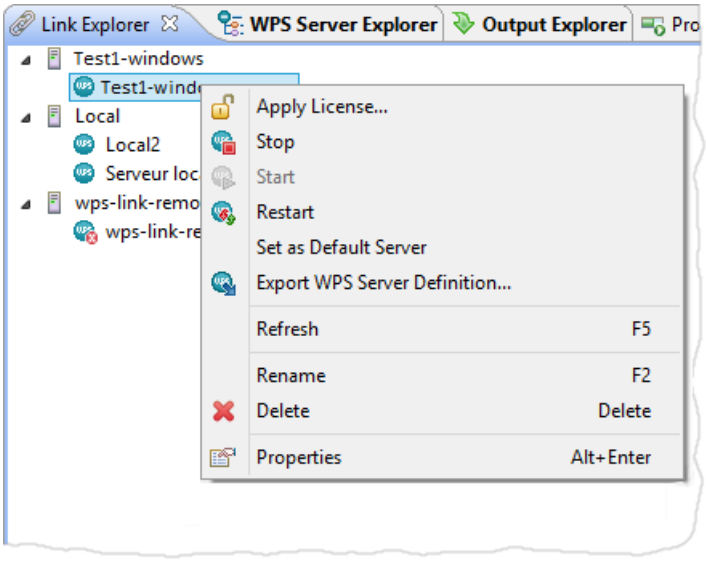


The options are largely self-explanatory. To create a **New WPS Server**, proceed as in *Creating a new remote WPS server* (page 14). To **Export Host Connection**, refer to **Exporting Host Connection Definitions to a File** in the **WPS Workbench** user guide. To **Import WPS Server Definition**, refer to **Importing WPS Server Definitions from a File** in the **WPS Workbench** user guide.

Note:

Servers can also be imported by dragging Server Definition Files (with the suffix `*.sdx`) onto a **Host Connection** node in the **Link Explorer**.

If you right-click on a server that is running, the following options are displayed:



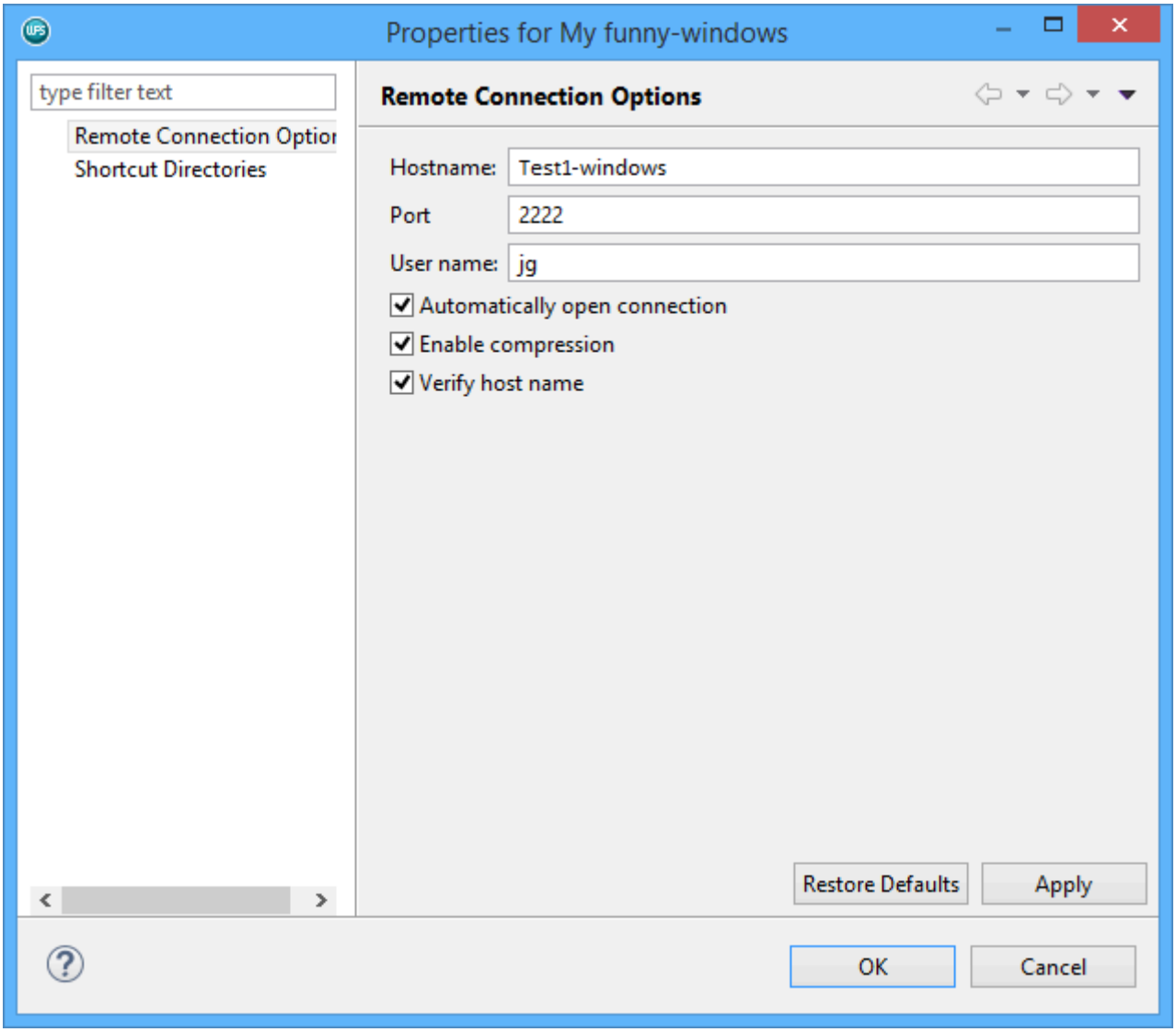
The options are once again largely self-explanatory. To **Export WPS Server Definition**, refer to **Exporting WPS Server Definitions to a File** in the **WPS Workbench** user guide. Selecting **Set as Default Server** means that this is the server that will be shown in bold, and that will always appear first, in the **Link Explorer**. It is also the server that is active when you simply click the **Run** button, **Clear Log** button, **Clear Results** button, or **Restart Server** button, as opposed to selecting a server via the drop-down associated with the button.

Properties

With the exception of a local host connection, if you select any other item in the **Link Explorer**, you will see information about that item displayed in the **Properties** view.

You can also get more detailed information about certain items in the **Link Explorer**. Right-click on an item and, from the popup menu, select the **Properties** option if it is enabled, to gain access to the following.

Host Connection Properties



- **Remote Connection Options** (for a remote host): **Hostname**, **Port** and **User name**.

> **Tip:**
>
> If you are going to be using the connection regularly, then you are strongly advised to ensure that **Automatically open connection** is selected, so that, once the associated server starts, the connection is already open.

- **Shortcut Directories**: A **Shortcut Directory** is a shortcut to a path on the host's file system. To set up a new shortcut, click **Add** and enter the shortcut name you want to use, followed by the full path (as shown when you hover over the location in **File Explorer**), and click **OK**. The shortcut name then appears in the host file system in **File Explorer**.

WPS Server Properties

- **Environment**: Details of the server's environment (working directory, process ID and environment variables).

- **Macro Variables**: The full list of automatic and global macro variables used by the server.
- **Startup**: Flag to indicate whether or not the server is to be started automatically on connection startup, the initial current directory for the server process (for remote servers), startup environment variables (for local servers), and startup system options.
- **System Options**: The currently applied system options. For more information about this topic, please refer to the **Configuration Files** section of the **WPS Workbench** user guide.
- **WPS License Information**: Full details about your licence key.
- **WPS Software Information**: Details about the WPS software, including the version number.

Commands (in the top right hand corner of the window)

- **Collapse All**: Collapse the view to show just the root nodes.
- **Create a new remote Host Connection**: Add a new remote connection (only one local connection is possible). Refer to *Creating a new remote host connection* (page 9).
- **Import a Host Connection definition**: Import a connection definition file. For further information on importing connections, refer to **Importing Host Connection Definitions from a File** in the **WPS Workbench** user guide.

  Note:

  Host connections can be also imported by dragging Connection Definition Files (with the suffix `*.cdx`) onto the **Link Explorer** view.

Dragging and Dropping a WPS Server

A WPS server can be dragged and dropped onto a different host connection. A simple drag and drop operation on a WPS server will move the server from its current host connection to the target host connection. Holding down the **Control** key while dragging will copy the WPS server. To maintain unique names, the copied server will be renamed automatically. The server can be renamed manually by right-clicking the server and selecting **Rename** from the context menu.

Working Example

In the following example, **WPS Link** is used to create a link from a Windows client to a remote UNIX server, which will execute a short program that was originally created on the client.

Note:

The following example assumes that **WPS Local Server** and **WPS Workbench** are installed and licensed on both the client Windows machine and the remote UNIX machine.

Creating a dataset and program

1. Copy the following data, saving it in, say, `C:\wps-link-demo\uk-fires.dat`:

```
2010/11 25000 7400 17600
2011/12 24200 7400 16900
2012/13 19900 5200 14600
```

2. On your client's **WPS Workbench**, create the following program, saving it as `wps-link-demo.sas`:

```
/* Fires in dwellings and other buildings, Great Britain, 2010/11-2012/13
Source: http://www.gov.uk */
DATA uk_fires;
infile "c:\wps-link-demo\uk-fires.dat"; INPUT Year $ Total Deliberate Accidental;
 run;
PROC PRINT data=uk_fires;
run;
```

Note:

Ensure that the path referenced in the `infile` statement corresponds to where you saved `uk-fires.dat`.

3. Execute the program on your local host.
4. The program reads the data file we created into a dataset, printing it out to the default HTML output, which resembles the following:

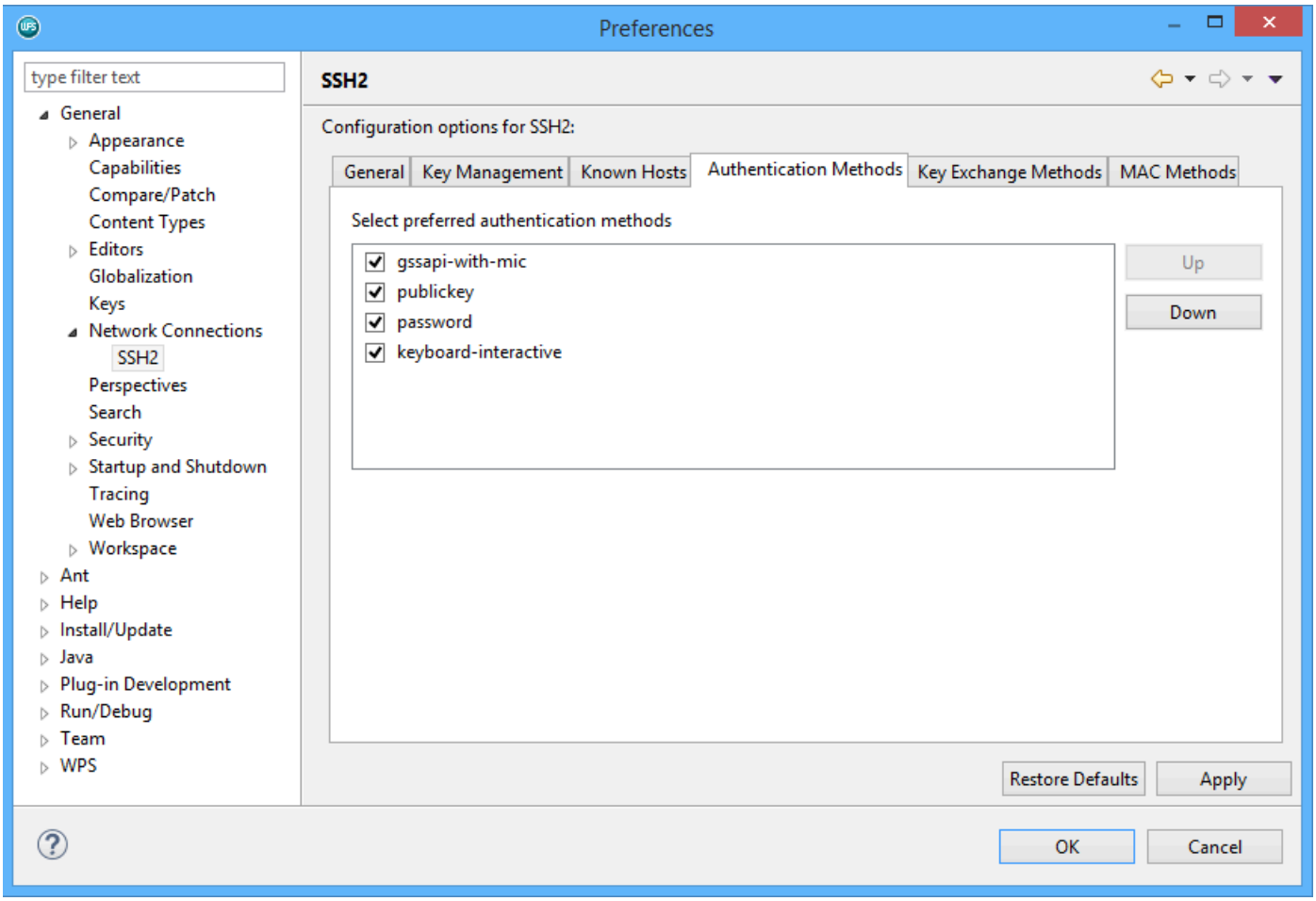
Obs	Year	Total	Deliberate	Accidental
1	2010/11	25000	7400	17600
2	2011/12	24200	7400	16900
3	2012/13	19900	5200	14600

Checking the authentication methods

We need to confirm that WPS Workbench is set up to use password sign-on.

1. On the main WPS Workbench menu, select **Window ➤ Preferences**.

2. In the left-hand pane, expand the **General** ➤ **Network Connection** ➤ **SSH2** items, select the **Authentication Methods** tab in the right-hand pane and ensure that the relevant items are selected (including **password**), before clicking **OK**.



Note:

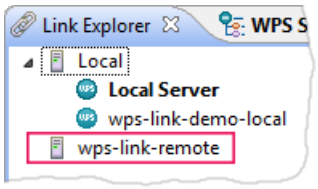
Password sign-on is not the most secure method, and the use of `Public key` - that is to say, `password-less` - authentication is advised for long-term use (refer to the *System Administration Guide* (page 26)). However, password sign-on does serve to verify that **WPS Link** is working in its most basic configuration.

Creating and starting the remote connection

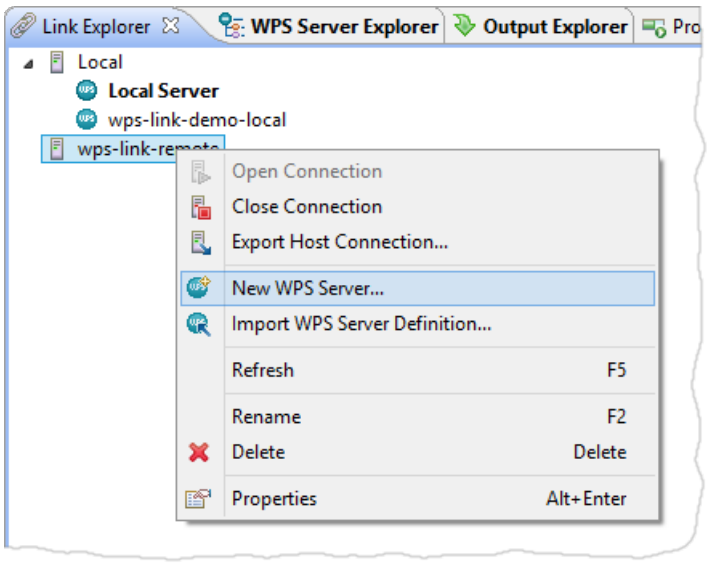
Creating a link to a remote computer is a two-step operation. Firstly, you create a connection to a remote host. Secondly, you add a WPS server to your new remote host, and it is this server that will run your programs remotely.

Before you start, check that you can sign on to the remote host via an SSH client, for example PuTTY.

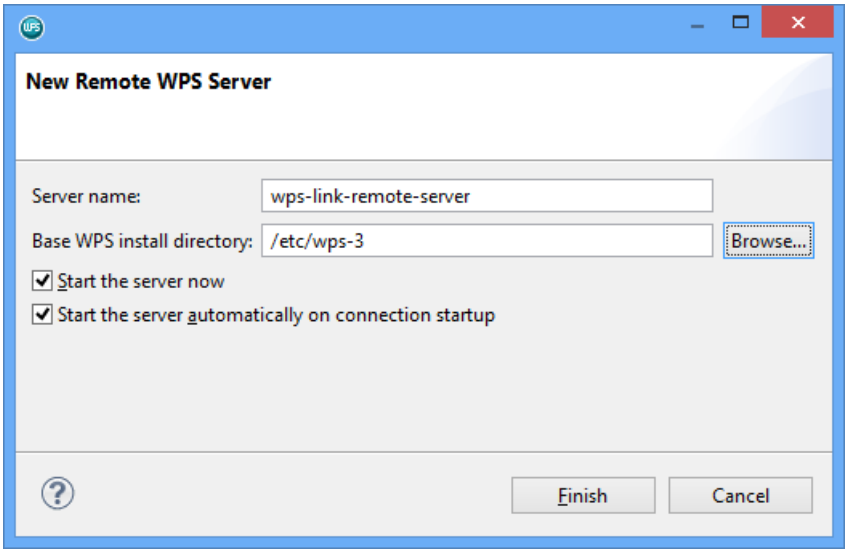
1. Go through step *1* (page 10) of *Creating a new remote host connection* (page 9).
2. Go through step *2* (page 11) of *Creating a new remote host connection* (page 9).
3. Go through step *3* (page 12) of *Creating a new remote host connection* (page 9).
4. Click **Finish** and proceed in accordance with step *5* (page 14) of *Creating a new remote host connection* (page 9).
5. In the subsequent **Password Required** dialog, type in your password on the remote server and click **OK**. The new connection appears in the **Link Explorer**:



6. In the **Link Explorer**, right-click on the new connection node and select **New WPS Server**:

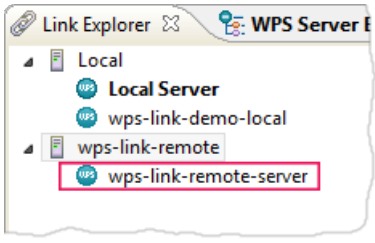


The **New Remote WPS Server** dialog appears:

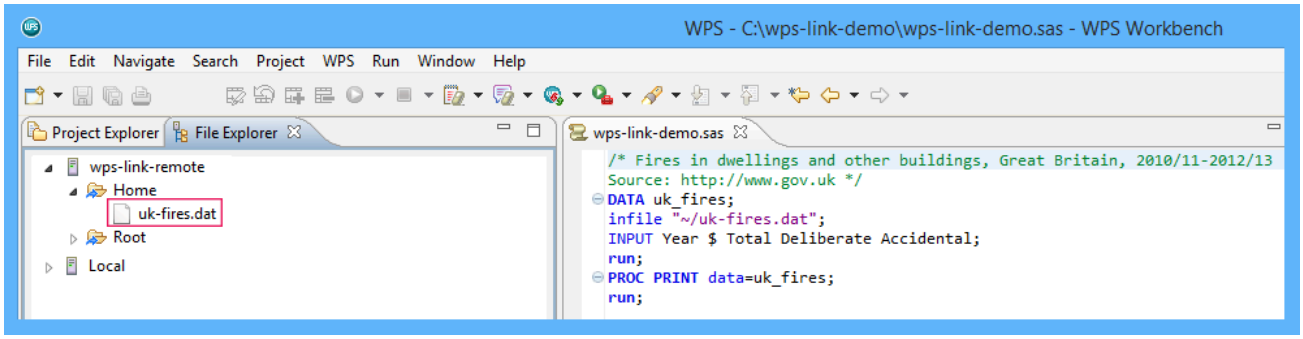


7. Complete the fields as shown above, providing the location of the remote WPS installation in the **Base WPS install directory** field. Click **Finish** when you are ready.

   If connection is successful, the server is created and appears in the *Link Explorer* (page 15):



8. Assuming that both the local and remote servers are connected, copy our data file - `uk-fires-dat` - from the local server to the `Home` directory on the UNIX server. You can either use the file transfer protocol of your choice, or copy and paste using **File Explorer**:

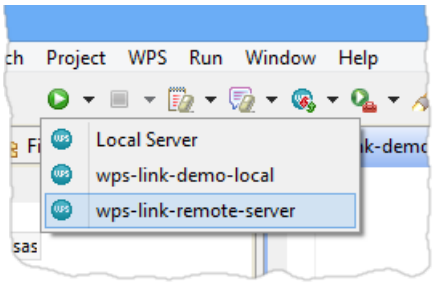


9. Click on `wps-link-demo.sas` in the **WPS Workbench** editor on the client, to bring the script into focus, and change the `infile` statement to `infile "~/uk-fires.dat";`, as shown above. This is because the path needs to be set to refer to `uk-fires.dat` in the home directory on the remote UNIX server.

Running the program on the remote server

1. Click and hold the black triangle next to the **Run** button .

   You are offered the additional choice of running the program on our new remote server:



   Note:

   If you set the remote server to be the default server in *Link Explorer* (page 15), then all you need to do in future is to click the **Run** button .

2. Run the program on the remote server and examine the output HTML and log.

   The HTML is identical to how it was on the local server, and the log will contain an indication that the program ran on a remote machine.

Note:

Once you have set up a new remote WPS server, it is as easy to run programs remotely as it is to run them locally - the only changes required are to ensure that file paths referenced are relative to the target computer.

System Administration Guide

This part of the guide is intended for system administrators who are responsible for the configuration of **WPS Communicate** and **WPS Link**, and, more specifically, for server authentication and the generation, deployment and verification of any required public and private keys.

Note:

WPS Communicate and **WPS Link** are separate features that can run independently. That is to say, you do not need one in order to run the other. You would use **WPS Communicate** to run selective code on remote server hosts or a z/OS mainframe, and **WPS Link** to run entire programs, via the Workbench GUI, on remote server hosts only. The two features are described together here as they both require means of remote authentication.

The following is a summary of the authentication methods as they apply to both **WPS Communicate** and **WPS Link**.

Authentication Method	WPS Communicate	WPS Link
Password	Yes	Yes
Public key with passphrase and keychain agent	Yes	Yes
Public key with passphrase and no keychain agent	No	Yes
Kerberos	Yes	Yes
Telnet on z/OS	Yes	No

SSH (Secure Shell) from a Windows client

This section covers the use of SSH with both **WPS Communicate** and **WPS Link** to create and maintain the connections between server and client machines.

Note:

The differences in use between **WPS Communicate** and **WPS Link** are highlighted where appropriate.

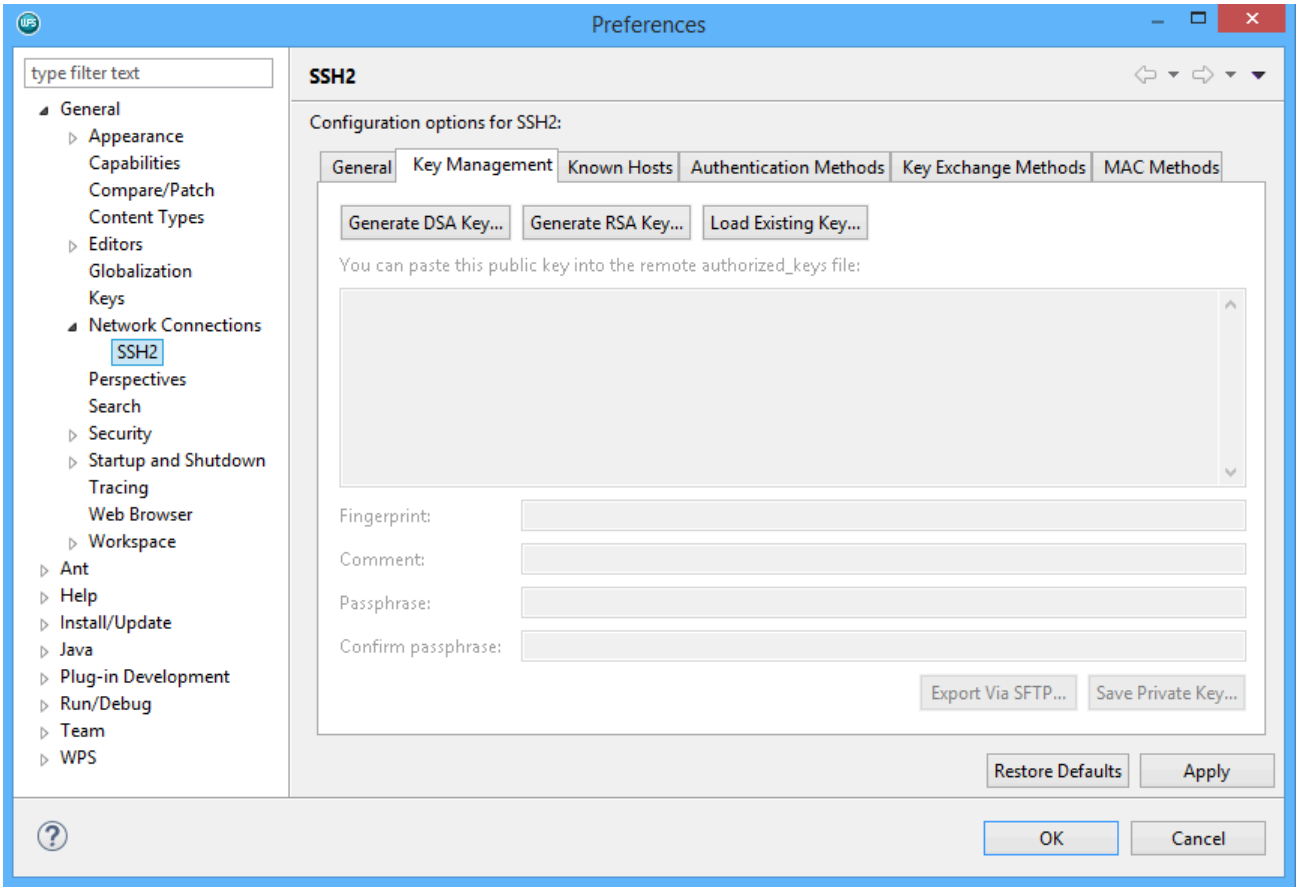
Before you access a remote host via **WPS Communicate** or **WPS Link**, it is important to ensure that you can access the remote host manually via an external SSH client such as PuTTY. This demonstrates that you can at least connect to the machine using the SSH protocol and that your user ID and password are valid.

Note:

If you intend to use *Public key authentication* (page 33), and keys have not already been generated on the server, then you may also wish to download PuTTYgen (refer to *Key generation using PuTTYgen* (page 34)). If you intend to use public keys with a **passphrase**, and you are using **WPS Communicate**, you will also need to download a keychain agent such as Pageant (refer to *Passphrase authentication using Pageant* (page 48)). The use of such an agent for **passphrases** is not necessary with **WPS Link**, although it may be desirable if you are connecting to multiple servers.

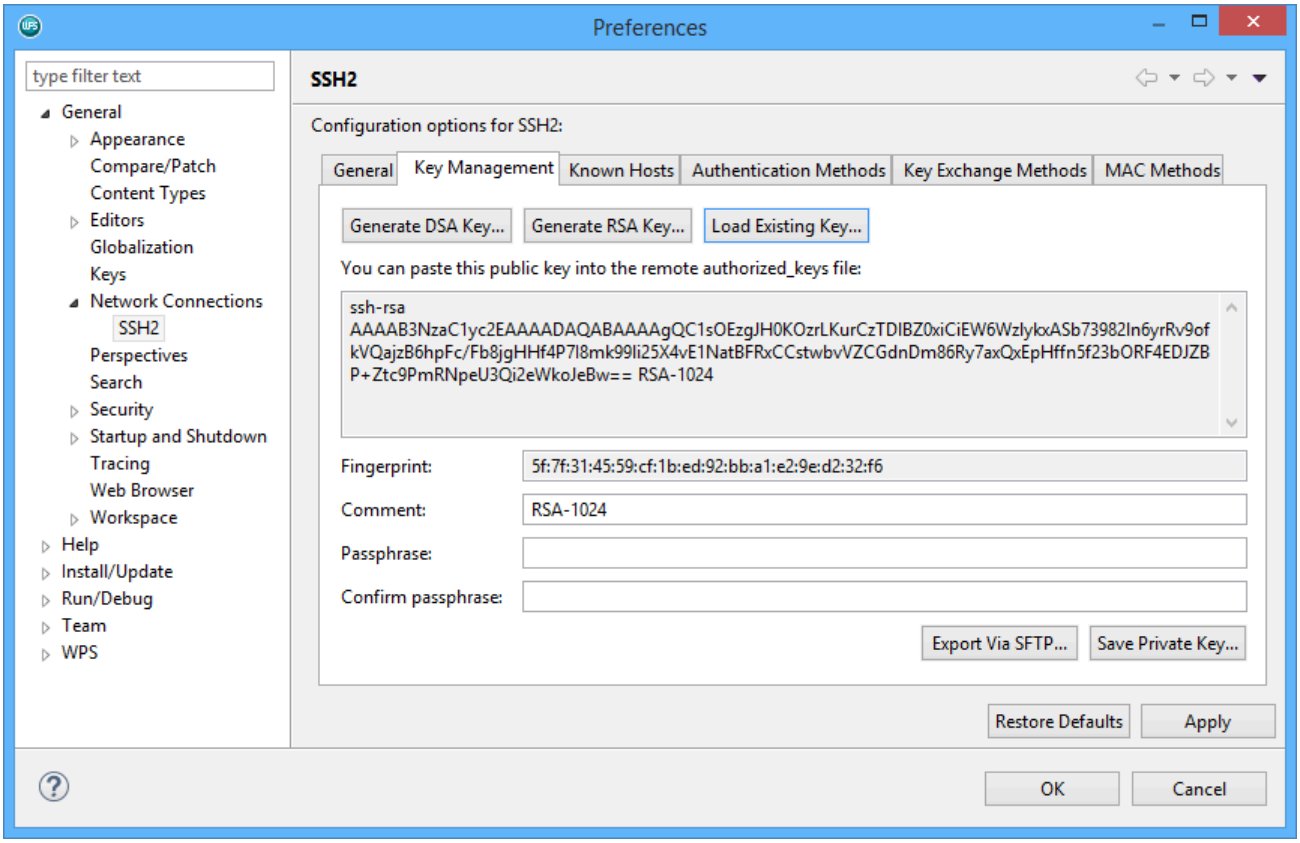
If you are using **WPS Link** and already have a private key, and wish to apply it, then proceed as follows:

1. On the WPS Workbench main menu, select **Window ➤ Preferences** and, in the left-hand pane of the subsequent **Preferences** dialog, expand the **General ➤ Network Connections ➤ SSH2** nodes.
2. Select the **Key Management** tab of the **Preferences** dialog:



3. Click **Load Existing Key...**.

4. Browse to the required private key and select it, to display the screen shown in the following example:



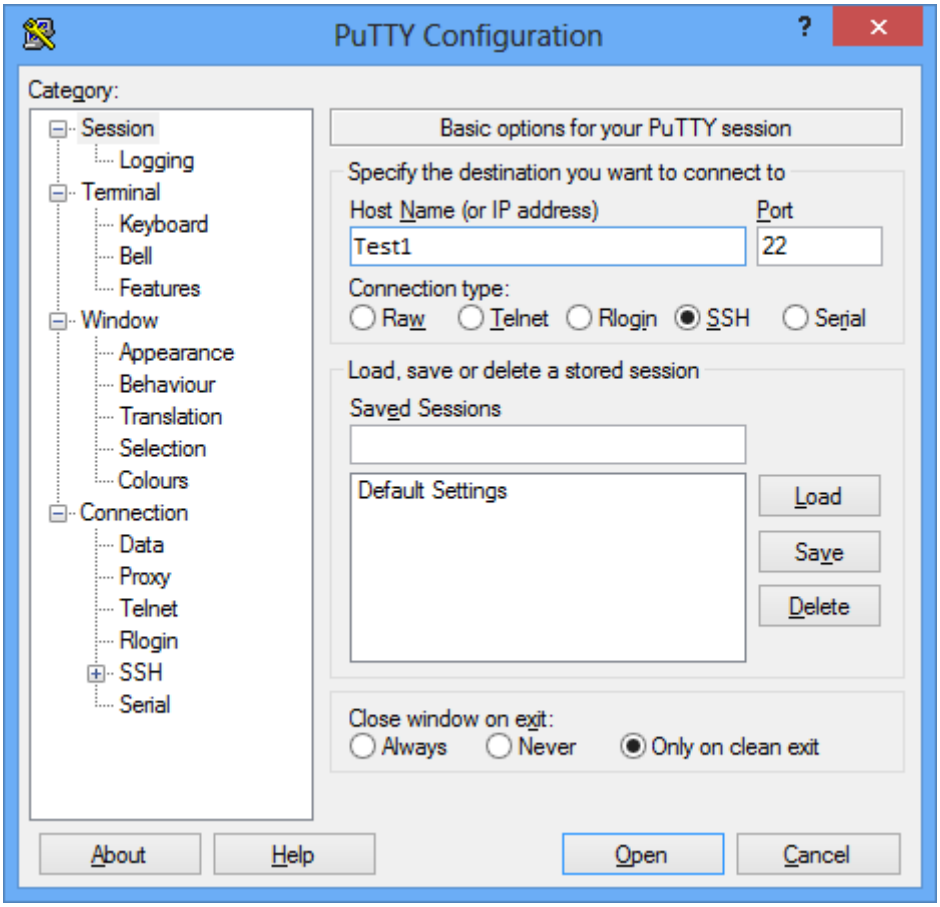
5. If you wish to apply a passphrase to your private key file, complete the **Passphrase** and **Confirm passphrase** fields.
6. Click **OK** to save your changes and dismiss the **Preferences** window.

Password authentication (using PuTTY) and WPS sign-on

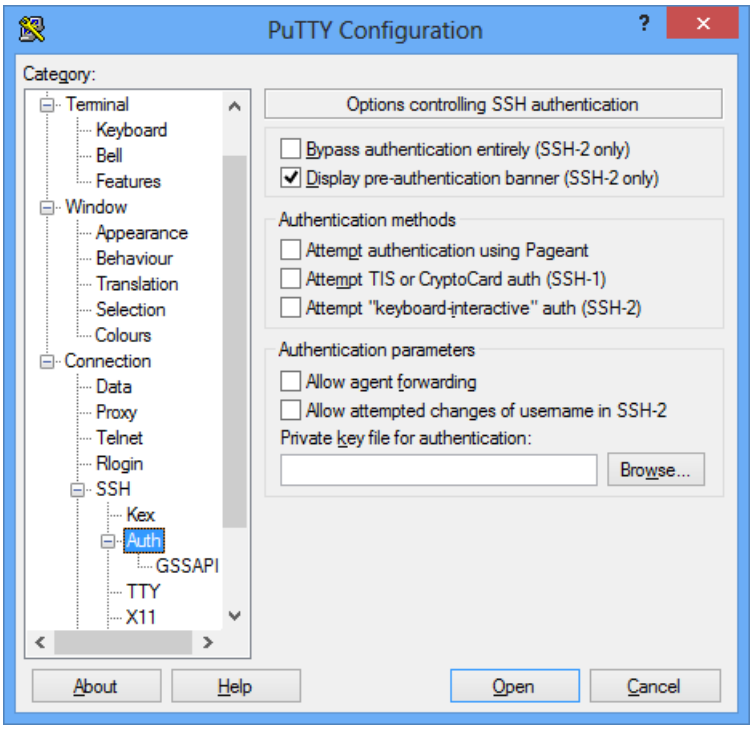
Manual SSH sign-on provides the opportunity to perform host key validation. For increased security, WPS performs host key validation during SSH sign-on. However, WPS has no mechanism for interacting with the user to accept new host keys, or to prompt about an apparent change of key. Instead, WPS relies upon host key acceptance having already been performed by an external SSH client, and will validate the host key it receives against the same database as is used by the external SSH client. On Windows clients, WPS will, by default, use the PuTTY host key database stored in the Windows registry,

so it is necessary to log onto the remote host using the PuTTY SSH client to validate the host key and add it to the host key database before attempting to make a connection with WPS.

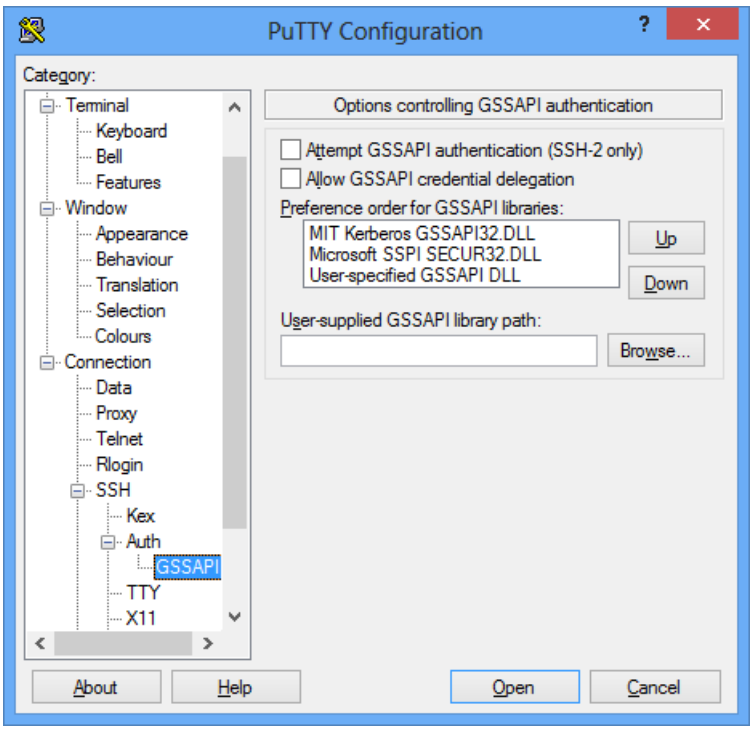
1. Launch the PuTTY client and enter the host name in the main **Host Name (or IP address)** entry field:



2. Expand the **SSH ➤ Auth** configuration page from the category list on the left, and ensure that nothing is selected under **Authentication methods** and that the **Private key file for authentication** field is empty:



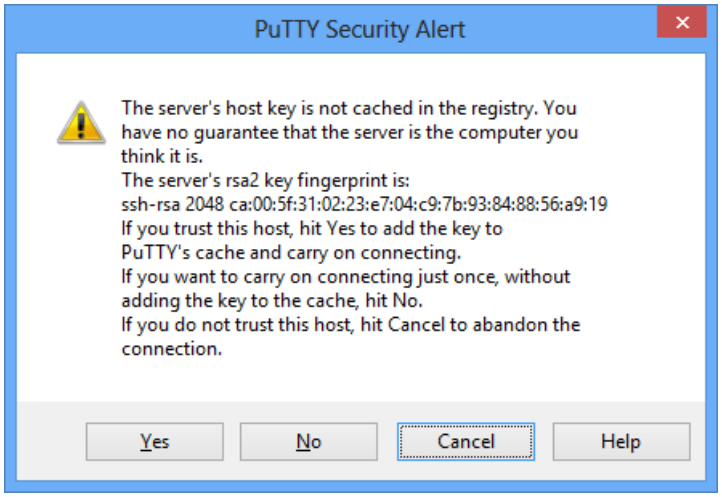
3. Select the **GSSAPI** page and ensure that **Attempt GSSAPI authentication** is not selected:



These checks ensure that only password authentication is being used, and that we have not inadvertently selected some more involved authentication mechanism.

4. Click on **Open** and when prompted, type in your password and press `Enter`.

   If this is the first time you have signed on to this particular host, an alert of the following kind will be displayed:



   At this point you should confirm that this is indeed the correct fingerprint for the host, and, assuming that it is, click **Yes** to accept the key permanently. This will later allow WPS to perform host key validation using the same cached key. If everything has worked, you will be logged in via a terminal session to your remote host. The host key will have been validated and stored in the Windows registry which is where WPS components will look for it. You can now log out safe in the knowledge that, when you launch WPS, it will be able to access the same remote server, automatically extracting the validated host key from the Windows registry to perform validation.

Note:

Do not mistake this host key validation for public key authentication - they are two entirely separate things. Host key validation simply gives you an opportunity to confirm that the host to which you are connecting is, indeed, the host to which you intended to connect.

---

5. If you are using **WPS Link**, create the required host connection and remote host server through **WPS Workbench**. If you are using **WPS Communicate**, sign onto WPS via the `SIGNON` statement - you need to specify either the IDENTITYFILE statement option or the SSH_IDENTITYFILE system option, for example:

```
SIGNON <servername> SSH
USERNAME="<username>"
password="<password>"
LAUNCHCMD="/home/installs/wps-3.2/bin/wps -dmr ";

RSUBMIT;
%PUT &SYSHOSTNAME;
ENDRSUBMIT;
SIGNOFF;
```

Alternatively:

```
OPTIONS SSH_IDENTITYFILE="C:\Users\techwriter\.ssh\wpscommunicate.ppk";
SIGNON <servername> SSH
password="<password>"
LAUNCHCMD="/home/installs/wps-3.2/bin/wps -dmr";

RSUBMIT;
%PUT &SYSHOSTNAME;
ENDRSUBMIT;
SIGNOFF;
```

Note:

You cannot use either **IDENTITYFILE** or **SSH_IDENTITYFILE** if you are using *Passphrase authentication using Pageant* (page 48).

Public key authentication

This method is more secure than using a simple password, and is sometimes called *password-less* authentication.

With SSH, authentication using keys not only improves security in general, but also, in the case of **WPS Communicate**, it avoids having user names and passwords committed to source code (even in obfuscated or encrypted form).

This authentication method relies upon a cryptographic key-pair, where the private key resides on (and never leaves) the client machine, and the public key is installed on the SSH server to which the client needs to connect, or, in the case of Windows, on **Bitvise SSH Server** (from WPS 3.2 onwards). The SSH protocol uses the key-pair to establish the identity of the client and perform the authentication.

Two methods are described by which the keys can be generated on a Windows client:

- *Key generation using PuTTYgen* (page 34)
- *Key generation using WPS Workbench* (page 37)

Following the generation of the public keys, they should be placed on the remote server in accordance with *Deploying public keys on the remote SSH server* (page 39), or, in the case of a Windows server, *Deploying public keys on Bitvise SSH Server* (page 41).

If you wish to connect to multiple servers, without having to remember or enter your password for each system, then you should also use *Passphrase authentication using Pageant* (page 48).

The validity of the key-pairs should then be checked in accordance with *Remote host access verification (using PuTTY) and WPS sign-on* (page 46).

Note:

Public key authentication can be used with both **WPS Communicate** and **WPS Link**. However, for **WPS Communicate**, if you are not using a keychain agent such as Pageant, there cannot be a **passphrase** on the private key file, as there is currently no interactive mechanism to prompt for it during WPS authentication.

Note:

You need to ensure that public key authentication is not disabled on the client machine.

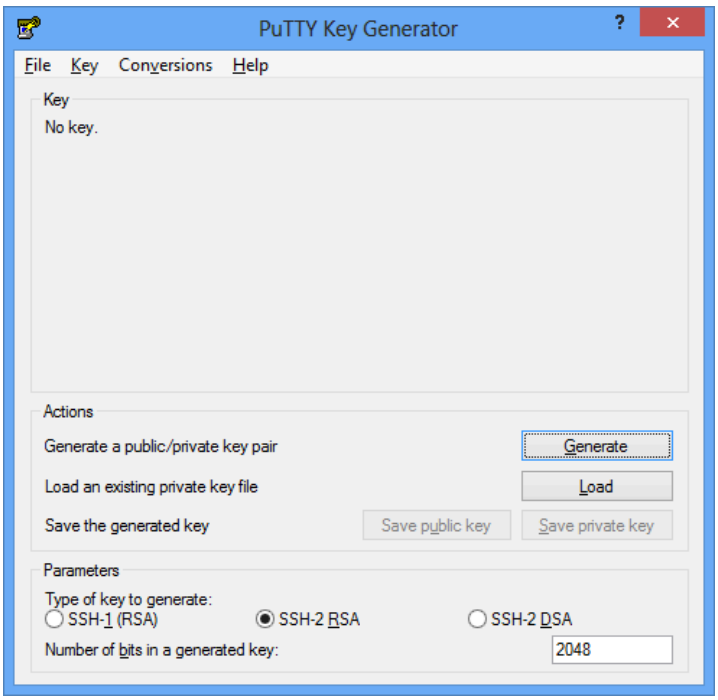
Key generation using PuTTYgen

To generate a key-pair, which is the combination of the private key and the public key for asymmetric encryption, proceed as follows:

Note:

You should be aware that, as an alternative, you can also use WPS Workbench to generate key-pairs (refer to *Key generation using WPS Workbench* (page 37)).

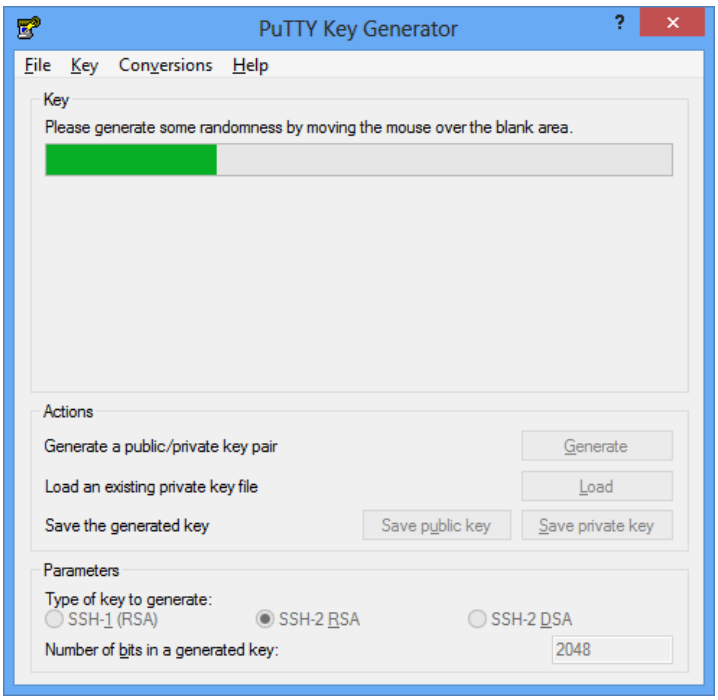
1. Launch the PuTTYgen tool (available via the same sources as PuTTY).



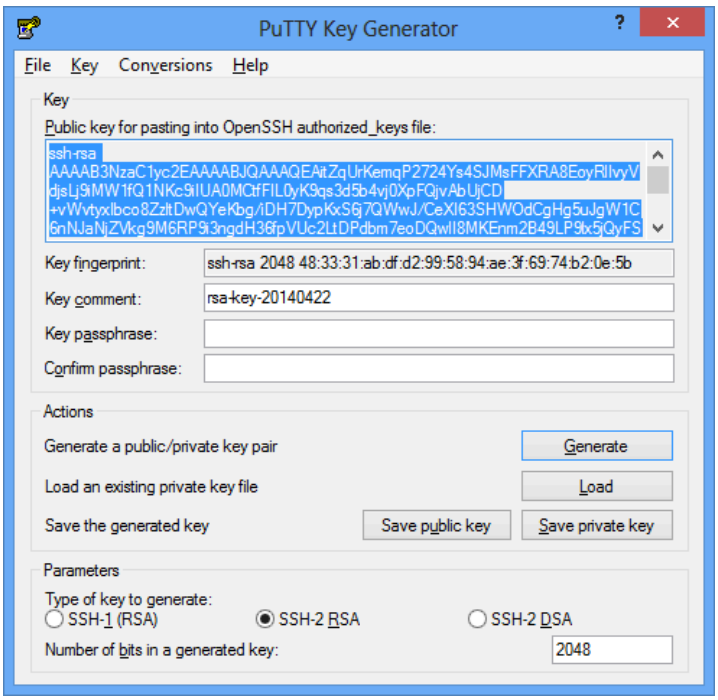
Important:

It is recommended that you use the parameter **SSH-2 RSA** with a minimum key length of **2048**.

2. Click the **Generate** button and move the mouse within the indicated area to generate some randomness - this will act as a seed for your key-pair:



3. The system generates a key-pair:



4. If you wish to apply a passphrase to your private key file, complete the **Key passphrase** and **Confirm passphrase** fields. You will need to do this if you are going to be using *Passphrase authentication using Pageant* (page 48).

Note:

If you are using **WPS Communicate**, do not enter a **passphrase** unless you are going to be using *Passphrase authentication using Pageant* (page 48). If you are using **WPS Link**, you can enter a **passphrase** and use it either with or without *Passphrase authentication using Pageant* (page 48).

---

5. Click **Save private key**. If you did not enter a **passphrase**, you will be asked to confirm that you wish to save the key without a **passphrase**. The resultant file is in PuTTY's native format (`*.PPK`), and, when you are prompted to save the file to a folder, you should ensure that it is stored in the `.ssh` folder in your user profile.

Note:

Ensure that the permissions on your private key file are such that only you can read it. This file is essentially your password, so it is important that no-one else can access the file.

---

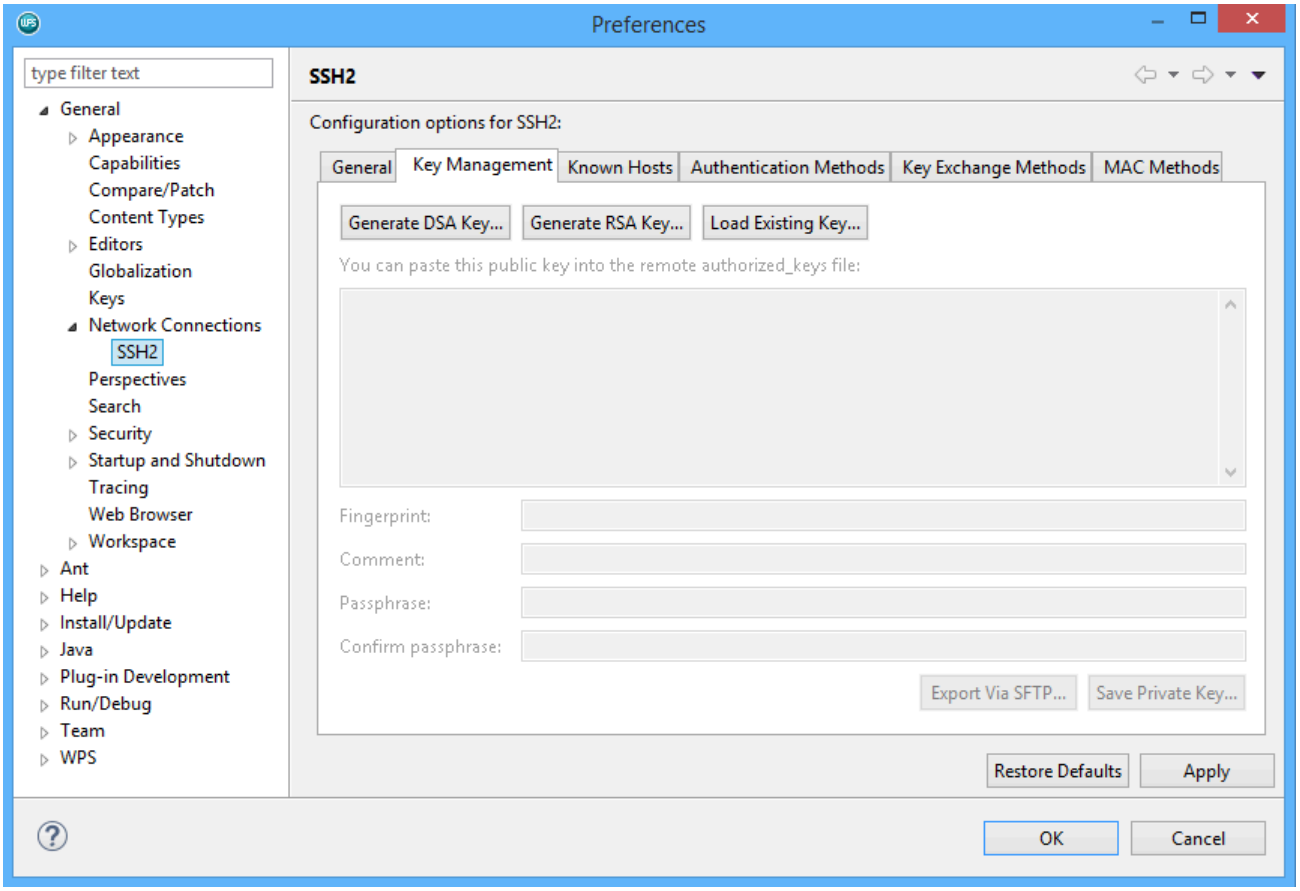
6. The public key that is highlighted in step *3* (page 36) contains the information needed to allow a user to verify that another party is in possession of the corresponding private key. The public key does not need to be kept secure, but you should save it by either copying it to your pasteboard and pasting it to a plain text file, or selecting **Save public key** in order to save it to the `.ssh` folder in your user profile. You should then proceed as in *Deploying public keys on the remote SSH server* (page 39) or *Deploying public keys on Bitvise SSH Server* (page 41).

Note:

If you are going to use **WPS Link** in conjunction with **WPS Communicate**, then, in order to avoid the need for two separate key-pairs, you should have a consistent strategy - that is to say, either avoid a **passphrase** in both cases, or else create a single **passphrase** and associate it with a single key-pair via *Passphrase authentication using Pageant* (page 48).

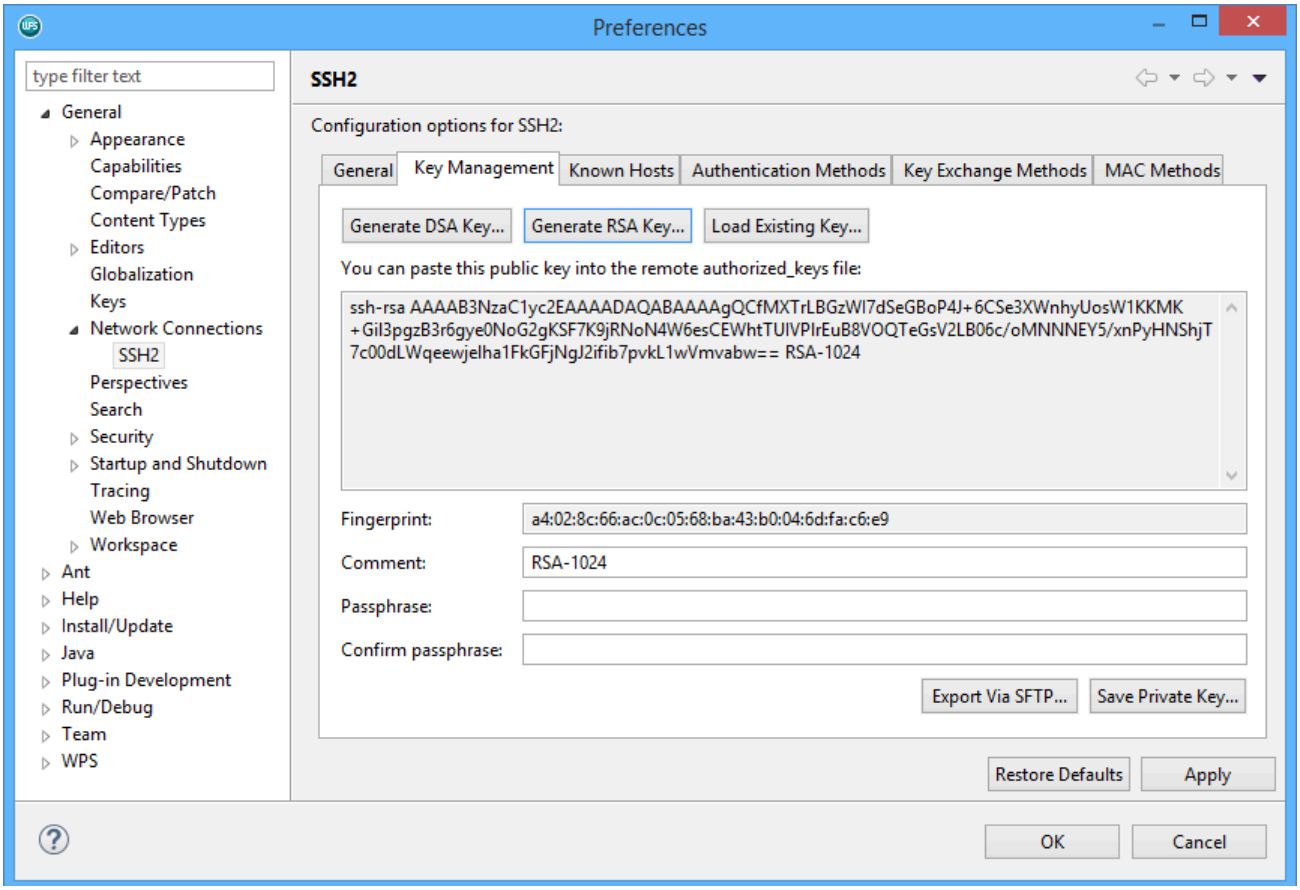
Key generation using WPS Workbench

1. On the WPS Workbench main menu, select **Window ➤ Preferences** and, in the left-hand pane of the subsequent **Preferences** dialog, expand the **General ➤ Network Connections ➤ SSH2** nodes.
2. Select the **Key Management** tab of the **Preferences** dialog:



3. Click **Generate RSA Key...**.

   A key pair is generated - the public key is displayed in a text box in the centre of the dialog, for example:



4. If you wish to apply a passphrase to your private key file, complete the **Passphrase** and **Confirm passphrase** fields. You will need to do this if you are going to be using *Passphrase authentication using Pageant* (page 48).

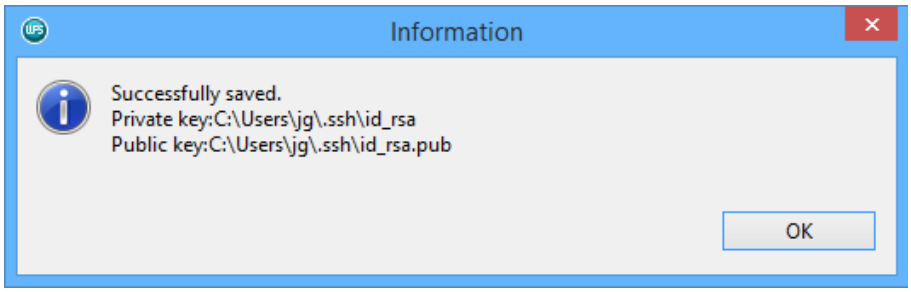
Note:

If you are using **WPS Communicate**, do not enter a **passphrase** unless you are going to be using *Passphrase authentication using Pageant* (page 48). If you are using **WPS Link**, you can enter a **passphrase** and use it either with or wihout *Passphrase authentication using Pageant* (page 48).

---

5. Click **Save Private Key**. If you did not enter a **passphrase**, you will be asked to confirm that you wish to save the key without a **passphrase**. When you are prompted to save the resultant key file to a folder, you should ensure that it is stored in the `.ssh` folder in your user profile. If you do not wish to use the default name of `id_rsa`, give the file a more meaningful name.

   WPS Workbench displays an information dialog confirming that it has saved your private key file, together with the corresponding public key file. It gives the public key file the same prefix as your private key file, but appends `.pub` to it, for example:



Note:

Ensure that the permissions on your private key file are such that only you can read it. This file is essentially your password, so it is important that no-one else can access the file.

6. Click **OK** to dismiss the **Information** dialog.
7. Click **OK** to save your changes and dismiss the **Preferences** window.
8. The public key that is both displayed on screen (for copying and pasting if required), and saved to a file, contains the information needed to allow a user to verify that another party is in possession of the corresponding private key. You should then proceed as in *Deploying public keys on the remote SSH server* (page 39) or *Deploying public keys on Bitvise SSH Server* (page 41).

Note:

If you are going to use **WPS Link** in conjunction with **WPS Communicate**, then, in order to avoid the need for two separate key-pairs, you should have a consistent strategy - that is to say, either avoid a **passphrase** in both cases, or else create a single **passphrase** and associate it with a single key-pair via *Passphrase authentication using Pageant* (page 48).

Deploying public keys on the remote SSH server

1. Log into the remote machine.

2. Once logged in, you must configure the server to accept your public key for authentication, so change into the `.ssh` directory and open the file `authorized_keys`.

   If this is the first public key to be put into the file, then you may need to create the directory and file first, by, for example, running the following commands:

   ```
   mkdir -p .ssh
   touch ~/.ssh/authorized_keys
   ```

3. Set the right permissions, for example:

   ```
   chmod 600 ~/.ssh/authorized_keys
   ```

   Note:

   You also need to ensure that your `$HOME` directory and `.ssh` directory have the permissions that are are appropriate to both the server and your particular operation.

4. Now you can add the public key to the `authorized_keys` file, as in the following example:

   ```
   cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
   ```

   If you currently have password-based SSH access configured to your server, and you have the `ssh-copy-id` utility installed, then you can simply transfer your public key by typing:

   ```
   ssh-copy-id username@remote_host
   ```

   You will then be prompted for the user account's password on the remote system. After typing in the password, the contents of your `~/.ssh/id_rsa.pub` key will be appended to the end of the user account's `~/.ssh/authorized_keys` file. You can then log into that account without a password:

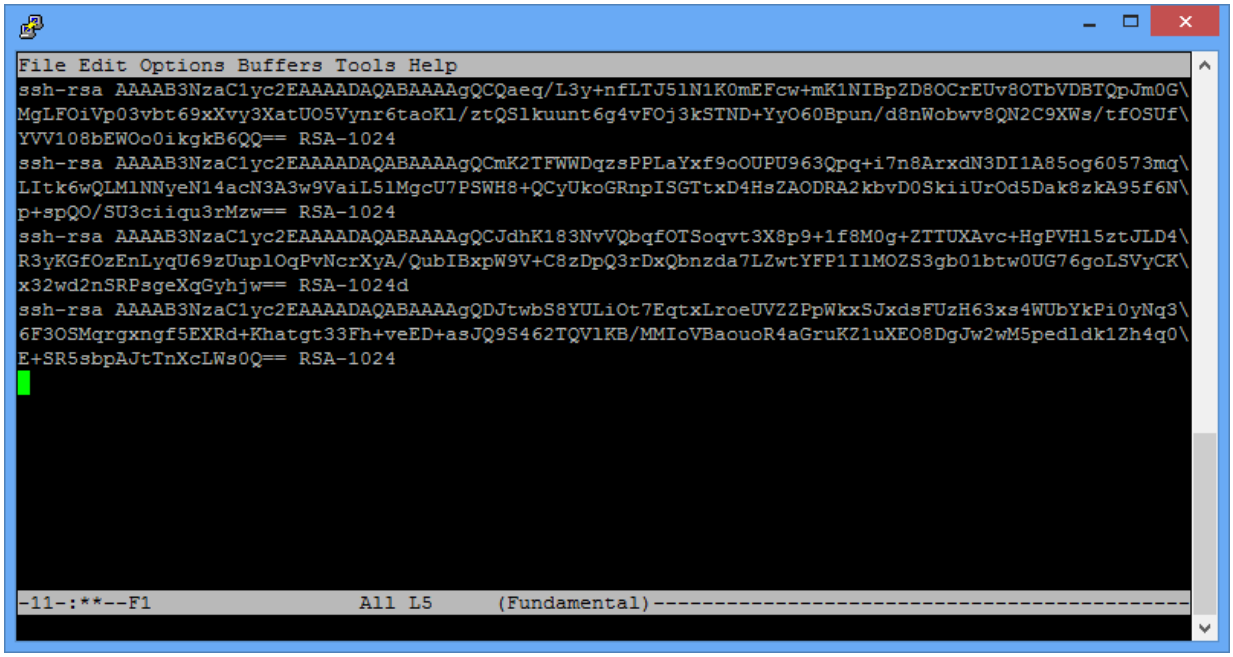
   ```
   ssh username@remote_host
   ```

   Alternatively, you can copy and paste the public key from PuTTYgen or **WPS Workbench** into the `authorized_keys` file, ensuring that it ends up on a single line.

5. Verify the contents of `~/.ssh/authorized_keys` to ensure that your public key was added properly, by entering the following on the command line:

```
more ~/.ssh/authorized_keys
```

The contents of a typical `~/.ssh/authorized_keys` file might resemble:



Note:

If you look carefully, you can see that the above file contains four public keys - each begins with `ssh-rsa` and ends with a phrase similar to `RSA-1024`.

Deploying public keys on Bitvise SSH Server

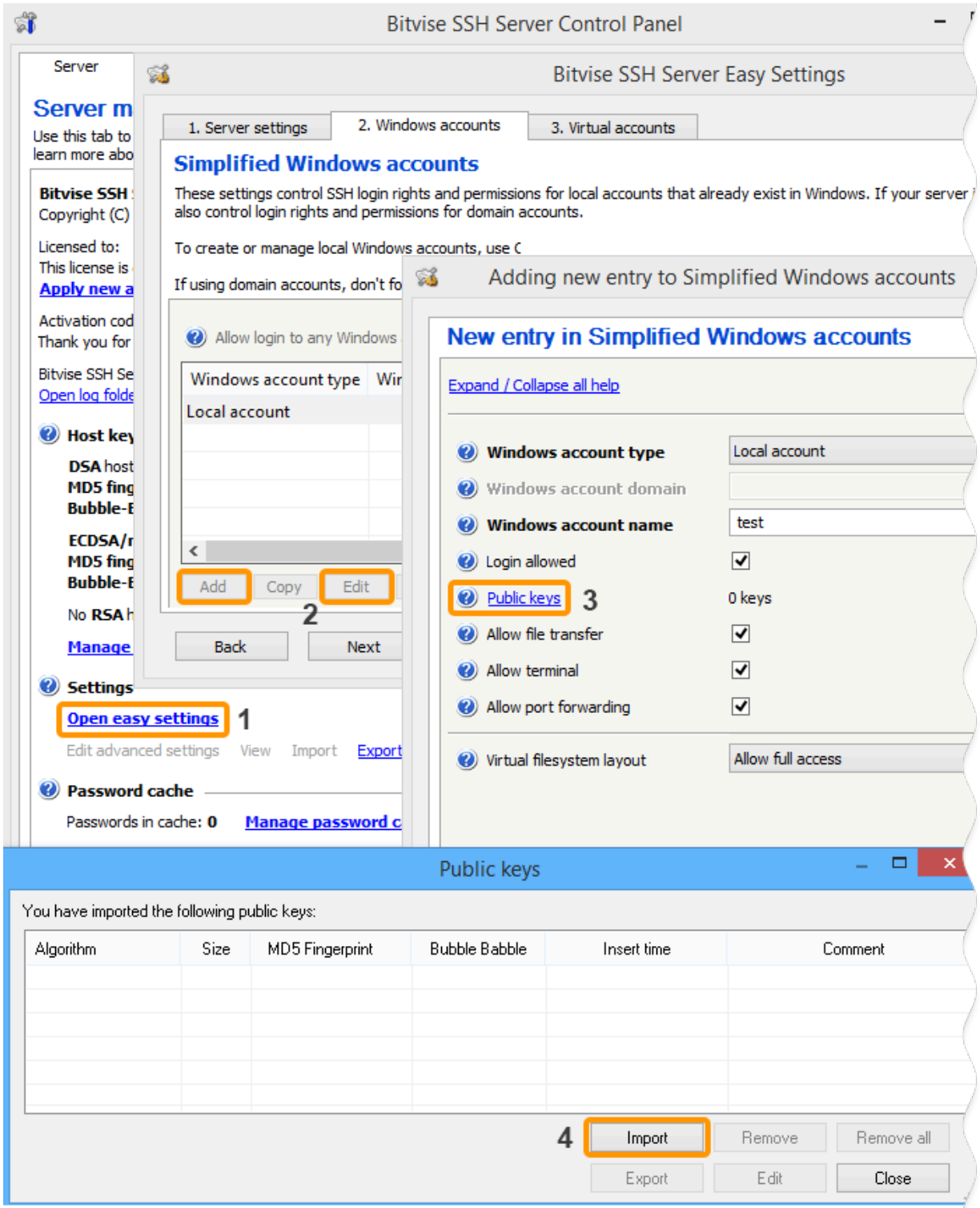
If you are new to Bitvise SSH Server (refer to the Bitvise documentation for specific configuration details), we highly recommend that you first make sure that you can establish a working SSH connection before you change any settings on the server. If you cannot connect to the SSH server using its default configuration, this is most likely due to a network or firewall problem that you will need to resolve before you are able to connect. In its default configuration, Bitvise SSH Server accepts connections on the often-used port number for SSH servers, 22. This is the only port that you need to open in your firewall in order to connect to the SSH server. If you use port forwarding to tunnel other applications through SSH, you should **not** open any additional ports for the tunnelled connections. All tunnelled connections are forwarded through the SSH session, established through port 22.

1. When connecting to Bitvise SSH Server with an SSH client for the first time, log in with the username and password of a Windows account that exists on the machine where the SSH server is running. To log into a Windows domain account, specify it in the `domain\account` format.

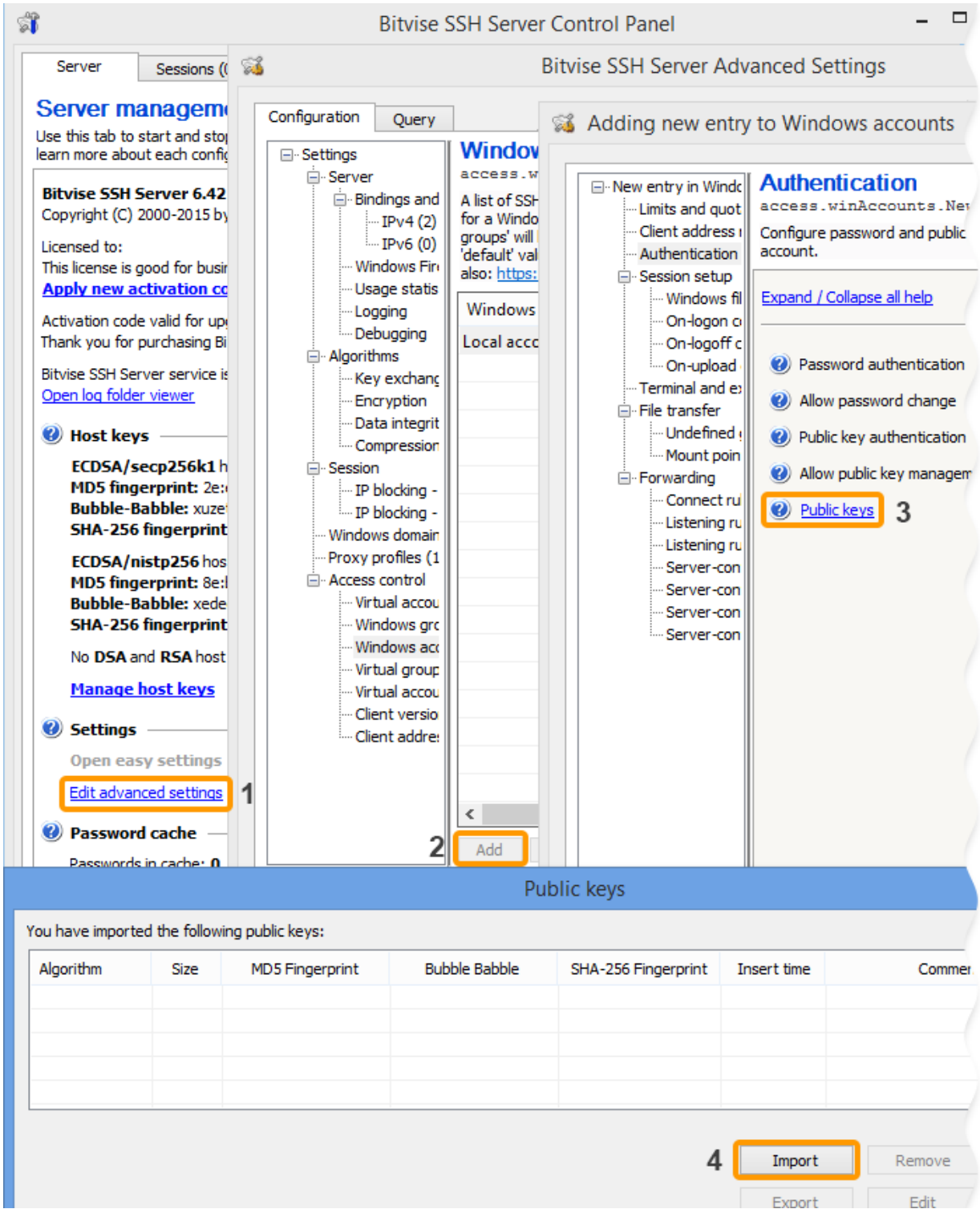
   You can use any SSH client to log into Bitvise SSH Server, as long as it supports SSH protocol version 2.

2. Having ensured that the public key has been saved to a file, transfer it to the machine where Bitvise SSH Server is installed, or to the machine from which you manage the SSH Server remotely using Bitvise SSH Client.

3. Open the **SSH Server Control Panel**, and then, to import the public key into the SSH user's account settings, use either **Open easy settings**:



or **Edit advanced settings**:


Bitvise SSH Server Control Panel
Server
Sessions
Server manageme
Use this tab to start and sto
learn more about each confi
Bitvise SSH Server 6.42
Copyright (C) 2000-2015 by
Licensed to:
This license is good for busir
Apply new activation co
Activation code valid for up
Thank you for purchasing Bi
Bitvise SSH Server service is
Open log folder viewer
Host keys
ECDSA/secp256k1 h
MD5 fingerprint: 2e:
Bubble-Babble: xuze
SHA-256 fingerprint
ECDSA/nistp256 hos
MD5 fingerprint: 8e:
Bubble-Babble: xede
SHA-256 fingerprint
No DSA and RSA host
Manage host keys
Settings
Open easy settings
Edit advanced settings
1
Password cache
Passwords in cache: 0
Bitvise SSH Server Advanced Settings
Configuration
Query
Settings
Server
Bindings and
IPv4 (2)
IPv6 (0)
Windows Fir
Usage statis
Logging
Debugging
Algorithms
Key exchang
Encryption
Data integrit
Compressior
Session
IP blocking -
IP blocking -
Windows domair
Proxy profiles (1
Access control
Virtual accou
Windows grc
Windows acc
Virtual group
Virtual accou
Client versio
Client addre
Windov
access.w
A list of SSH
for a Windo
groups' will
'default' val
also: https:
Windows
Local acco
2
Add
Adding new entry to Windows accounts
New entry in Windc
Limits and quot
Client address
Authentication
Session setup
Windows fil
On-logon c
On-logoff c
On-upload
Terminal and e
File transfer
Undefined
Mount poin
Forwarding
Connect ru
Listening ru
Listening ru
Server-con
Server-con
Server-con
Server-con
Authentication
access.winAccounts.Ne
Configure password and public
account.
Expand / Collapse all help
Password authentication
Allow password change
Public key authentication
Allow public key manager
Public keys
3
Public keys
You have imported the following public keys:
Algorithm
Size
MD5 Fingerprint
Bubble Babble
SHA-256 Fingerprint
Insert time
Commer
4
Import
Remove
Export
Edit

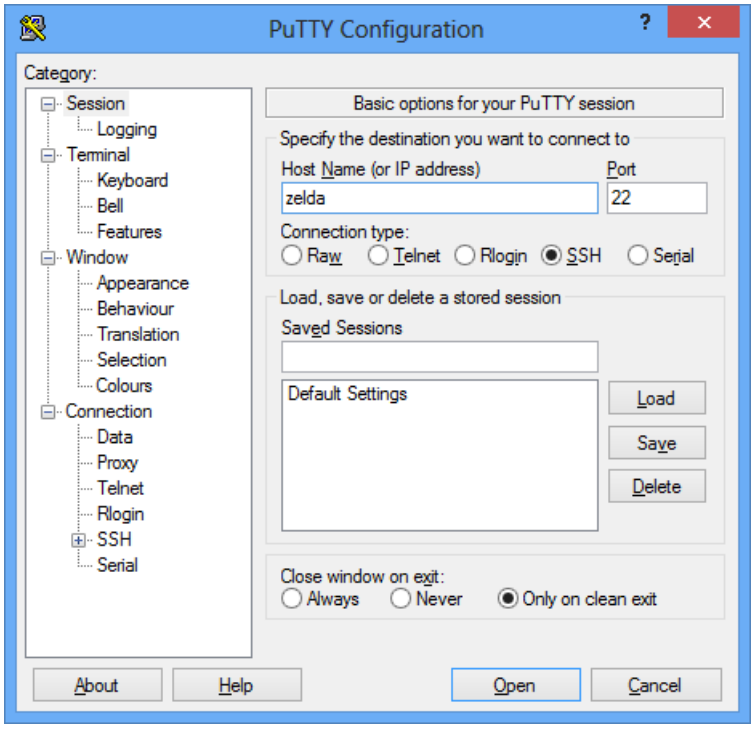
Note:

For Windows accounts, Bitvise SSH Server also supports synchronisation with `~/.ssh/authorized_keys`, provided that this feature is enabled in **Advanced SSH Server settings**, under **Access control**. If this feature is enabled, Bitvise SSH Server will check for the existence of the `authorized_keys` file when the user logs out. If the file exists, Bitvise SSH Server will replace all the public keys configured for the user with the keys found in this file.

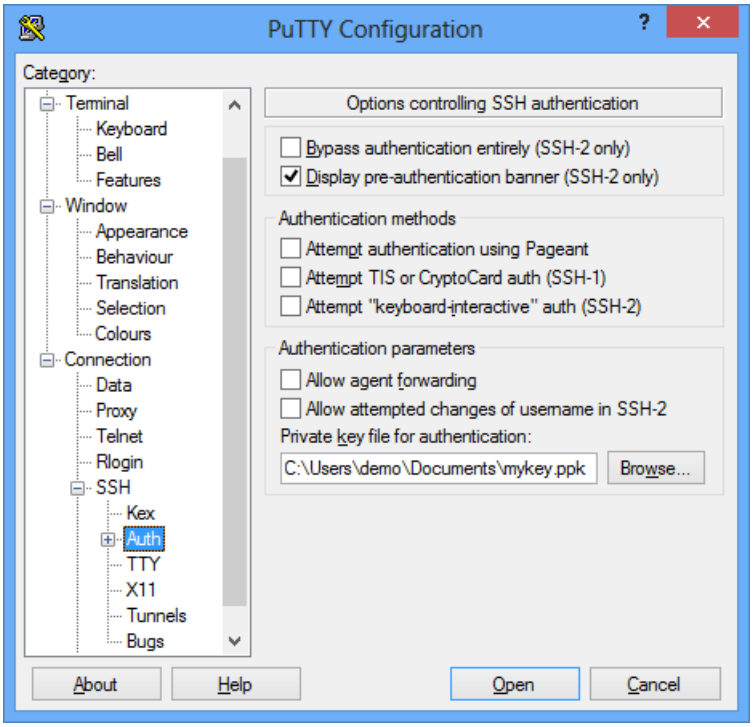
Remote host access verification (using PuTTY) and WPS sign-on

1. Verify authentication using PuTTY, as follows.

   a. Launch the PuTTY client and enter the host name in the **Host Name** field:



   b. Select the **SSH** ➤ **Auth** configuration page from the category list on the left and ensure that nothing is checked under **Authentication methods** (unless you are verifying *Passphrase authentication using Pageant* (page 48), in which case **Attempt authentication using Pageant** needs to be selected).



c. In the **Private key file for authentication** field, enter the name of the private key file you generated via either *Key generation using PuTTYgen* (page 34) or *Key generation using WPS Workbench* (page 37).

d. Click **Open** - you will be authenticated via your key-pair combination and a console window will open. The window displays a notification regarding the authentication method.

> **Note:**
>
> If you are prompted for a password, then public key authentication has failed.

2. If public key authentication is successful, then, if you are using **WPS Link**, create the required host connection and remote host server through **WPS Workbench**. If you are using **WPS Communicate**, sign onto WPS using your private key, via the `SIGNON` statement, for which you need to specify either the **IDENTITYFILE** statement option or the **SSH_IDENTITYFILE** system option, for example:

```
SIGNON <servername> SSH
USERNAME="<username>"
IDENTITYFILE="C:\Users\techwriter\.ssh\wpscommunicate.ppk"
LAUNCHCMD="/home/installs/wps-3.2/bin/wps -dmr ";
```

Alternatively:

```
OPTIONS SSH_IDENTITYFILE="C:\Users\techwriter\.ssh\wpscommunicate.ppk";
SIGNON <servername> SSH
username="<username>"
LAUNCHCMD="/home/installs/wps-3.2/bin/wps -dmr";
```

> **Note:**
>
> You cannot use either **IDENTITYFILE** or **SSH_IDENTITYFILE** if you are using *Passphrase authentication using Pageant* (page 48).

Passphrase authentication using Pageant

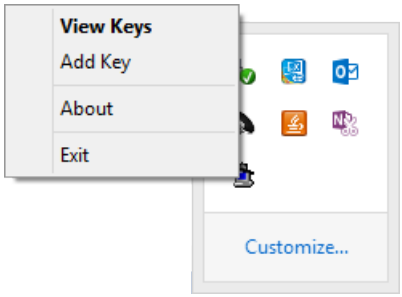
WPS Communicate does not support the reading of private key files that have been saved in encrypted form with a **passphrase**. However, it is still possible to use private key-pairs of this form if you use a keychain agent. We will assume that you are using Pageant on Windows.

A keychain agent is a utility that runs on the client machine and stores the decrypted public key file. The SSH client (or WPS when performing an SSH sign-on) contacts the agent for a public key to use when connecting to a given host. Although a password is still required to decrypt the private key, this is only required once - when the keychain agent first opens the private key.

This procedure assumes that you have already generated a key-pair with a **passphrase**, and deployed the public key on the remote SSH server (for UNIX/Linux), or on Bitvise SSH Server (for Windows).

Proceed as follows:

1. Run the Pageant tool, right-click the **system tray** icon and choose **View Keys** or **Add Keys** to add your private key file.



   At this point you will be prompted for the **passphrase** for the private key file.

2. Pageant opens and decrypts it, allowing clients (such as the normal PuTTY program or WPS) to request the loaded identity list for authentication.

> **Note:**
> WPS automatically detects whether or not Pageant is running.

SSH (Secure Shell) from a UNIX client

Before you access a remote host via **WPS Communicate** or **WPS Link**, it is important to ensure that you can access the remote host manually via SSH. This demonstrates that you can at least connect to the machine using the SSH protocol and that your user ID and password are valid.

Note:

If you intend to use *Public key authentication* (page 50), and keys have not already been generated on the server, then you may wish to use **ssh-keygen** (refer to *Key generation using ssh-keygen* (page 51)). If you intend to use public keys with a **passphrase**, and you are using **WPS Communicate**, you will also need to download a keychain agent such as **ssh-agent** (refer to *Passphrase authentication using ssh-agent* (page 58)). The use of such an agent for **passphrases** is not necessary with **WPS Link**, although it may be desirable if you are connecting to multiple servers.

Password authentication and WPS sign-on

WPS will use the OpenSSH host key database stored in the `~/.ssh/known_hosts` file by default. It is necessary to log onto the remote host using the OpenSSH client's command line to validate and accept the host key, and ensure that it is added it to the `known_hosts` file before attempting to make a connection with WPS. With OpenSSH, it is also possible for a system administrator to add keys manually to the `/etc/ssh/ssh_known_hosts` file, in preference to the `~/.ssh/known_hosts` file.

To sign on via SSH to a remote host, and ensure that password authentication is employed:

1. Issue the following command:

```
ssh -o PreferredAuthentications=password <hostname>
```

2. Verify that you receive a password prompt:

```
<user>@<hostname>'s password:
```

Note:

It is important to ensure that password authentication is being used, and that the host is not, for example, configured to accept only public key sign-on.

3. If you are using **WPS Link**, create the required host connection and remote host server through **WPS Workbench**. If you are using **WPS Communicate**, sign onto WPS via the `SIGNON` statement - you need to specify either the **IDENTITYFILE** statement option or the **SSH_IDENTITYFILE** system option, for example:

```
SIGNON <servername> SSH
USERNAME="<username>"
password="<password>"
LAUNCHCMD="/home/installs/wps-3.2/bin/wps -dmr ";

RSUBMIT;
%PUT &SYSHOSTNAME;
ENDRSUBMIT;
SIGNOFF;
```

Alternatively:

```
OPTIONS SSH_IDENTITYFILE="C:\Users\techwriter\.ssh\wpscommunicate.ppk";
SIGNON <servername> SSH
password="<password>"
LAUNCHCMD="/home/installs/wps-3.2/bin/wps -dmr";

RSUBMIT;
%PUT &SYSHOSTNAME;
ENDRSUBMIT;
SIGNOFF;
```

Note:

You cannot use either **IDENTITYFILE** or **SSH_IDENTITYFILE** if you are using *Passphrase authentication using ssh-agent* (page 58).

Public key authentication

The method described by which the keys can be generated on a UNIX client, is *Key generation using ssh-keygen* (page 51).

Following the generation of the public keys, they should be placed on the remote server in accordance with *Deploying public keys on the remote SSH server* (page 39), or, in the case of a Windows server, *Deploying public keys on Bitvise SSH Server* (page 41).

If you wish to connect to multiple servers, without having to remember or enter your password for each system, then you should also use *Passphrase authentication using ssh-agent* (page 58).

The validity of the key-pairs should then be checked in accordance with *Remote host access verification and WPS sign-on* (page 57).

Note:

Public key authentication can be used with both **WPS Communicate** and **WPS Link**. However, for **WPS Communicate**, if you are not using a keychain agent such as **ssh-agent**, there cannot be a **passphrase** on the private key file, as there is currently no interactive mechanism to prompt for it during WPS authentication.

Note:
You need to ensure that public key authentication is not disabled on the client machine.

Key generation using ssh-keygen

The **ssh-keygen** program allows you to create RSA keys for use by SSH protocol version 2. The type of key to be generated is specified by the `-t` option. If invoked without any arguments, **ssh-keygen** will generate an RSA key for use in SSH protocol 2 connections.

1. Generate the key-pair. In the following example, we have logged onto `hostA` as `wplusr`:

```
ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/wplusr/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/wplusr/.ssh/id_rsa.
Your public key has been saved in /home/wplusr/.ssh/id_rsa.pub.
The key fingerprint is:
f6:61:a8:27:35:cf:4c:6d:13:22:70:cf:4c:c8:a0:23 wplusr@hostA
```

Note:
Do not enter a **passphrase** if you are using **WPS Communicate**. You need to use a keychain agent for this (refer to *Passphrase authentication using ssh-agent* (page 58)). In the above example, the private key was saved in `.ssh/id_rsa` (this file is read-only and only for you, and no-one else must see the contents of that file, as it is used to decrypt all correspondence encrypted with the public key), and the public key in `.ssh/id_rsa.pub`. This is the file that needs to be added to the `~/.ssh/authorized_keys` file on the remote machine.

2. To deploy the public key, proceed as in *Deploying public keys on the remote SSH server* (page 39) or *Deploying public keys on Bitvise SSH Server* (page 41).

Deploying public keys on the remote SSH server

1. Log into the remote machine.
2. Once logged in, you must configure the server to accept your public key for authentication, so change into the `.ssh` directory and open the file `authorized_keys`.

   If this is the first public key to be put into the file, then you may need to create the directory and file first, by, for example, running the following commands:

```
mkdir -p .ssh
touch ~/.ssh/authorized_keys
```

3. Set the right permissions, for example:

```
chmod 600 ~/.ssh/authorized_keys
```

Note:

You also need to ensure that your `$HOME` directory and `.ssh` directory have the permissions that are are appropriate to both the server and your particular operation.

4. Now you can add the public key to the `authorized_keys` file, as in the following example:

```
cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```

If you currently have password-based SSH access configured to your server, and you have the `ssh-copy-id` utility installed, then you can simply transfer your public key by typing:

```
ssh-copy-id username@remote_host
```

You will then be prompted for the user account's password on the remote system. After typing in the password, the contents of your `~/.ssh/id_rsa.pub` key will be appended to the end of the user account's `~/.ssh/authorized_keys` file. You can then log into that account without a password:

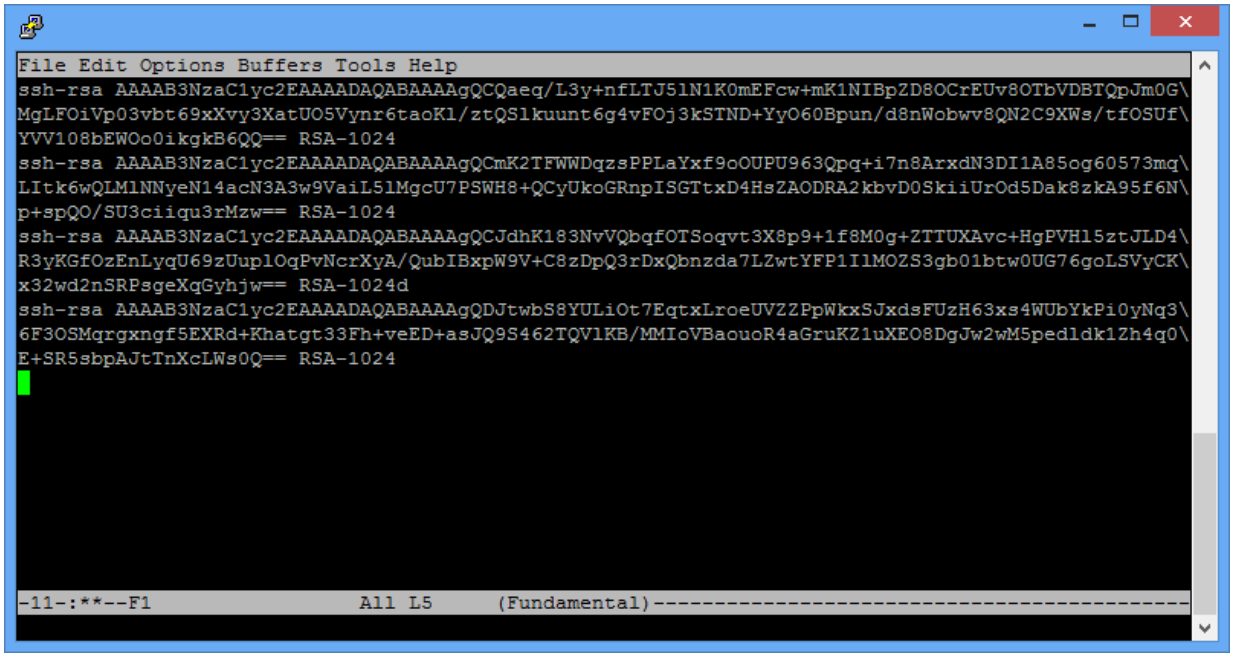
```
ssh username@remote_host
```

Alternatively, you can copy and paste the public key from PuTTYgen or **WPS Workbench** into the `authorized_keys` file, ensuring that it ends up on a single line.

5. Verify the contents of `~/.ssh/authorized_keys` to ensure that your public key was added properly, by entering the following on the command line:

```
more ~/.ssh/authorized_keys
```

The contents of a typical `~/.ssh/authorized_keys` file might resemble:



Note:

If you look carefully, you can see that the above file contains four public keys - each begins with `ssh-rsa` and ends with a phrase similar to `RSA-1024`.

Deploying public keys on Bitvise SSH Server

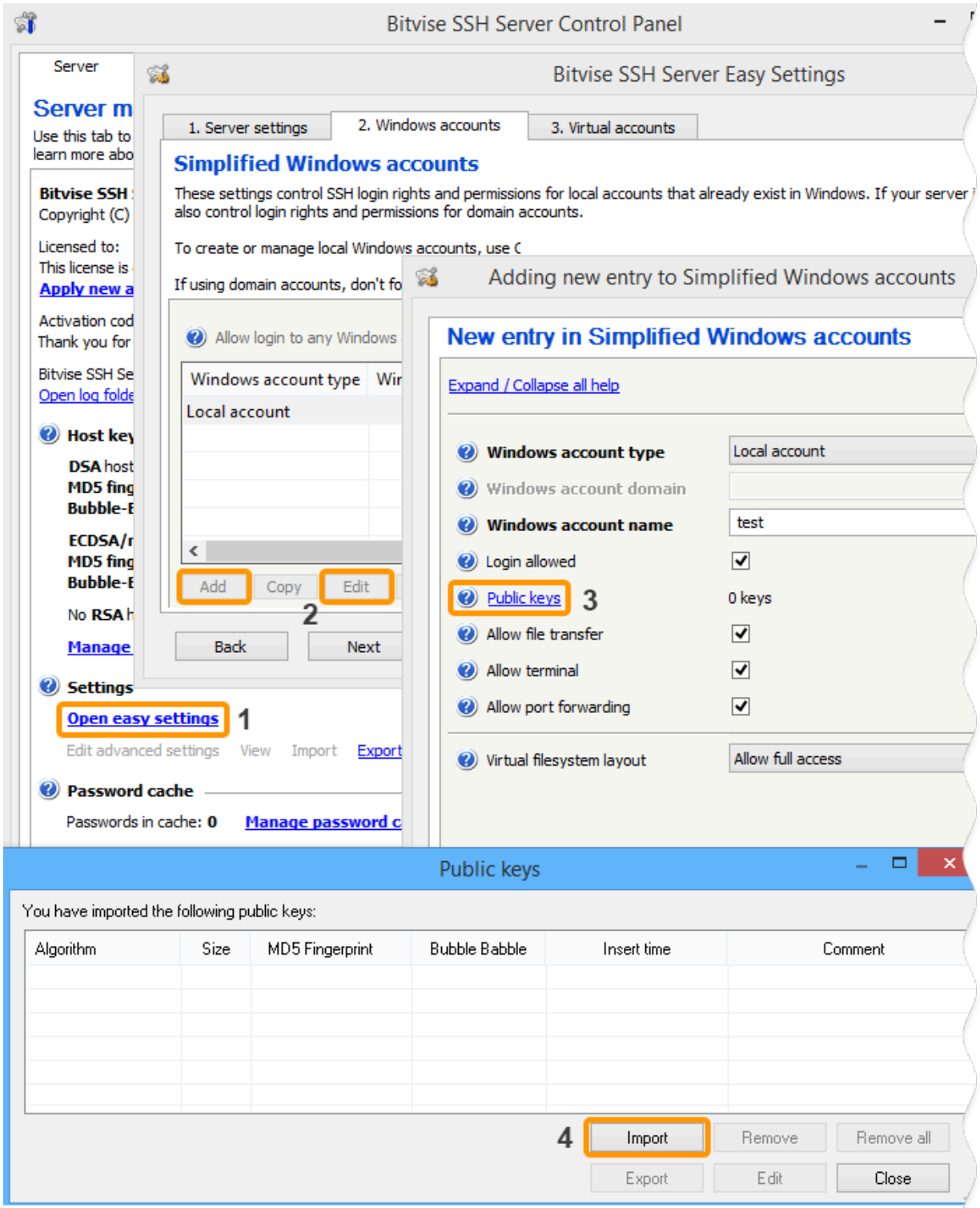
If you are new to Bitvise SSH Server (refer to the Bitvise documentation for specific configuration details), we highly recommend that you first make sure that you can establish a working SSH connection before you change any settings on the server. If you cannot connect to the SSH server using its default configuration, this is most likely due to a network or firewall problem that you will need to resolve before you are able to connect. In its default configuration, Bitvise SSH Server accepts connections on the often-used port number for SSH servers, 22. This is the only port that you need to open in your firewall in order to connect to the SSH server. If you use port forwarding to tunnel other applications through SSH, you should **not** open any additional ports for the tunnelled connections. All tunnelled connections are forwarded through the SSH session, established through port 22.

1. When connecting to Bitvise SSH Server with an SSH client for the first time, log in with the username and password of a Windows account that exists on the machine where the SSH server is running. To log into a Windows domain account, specify it in the `domain\account` format.

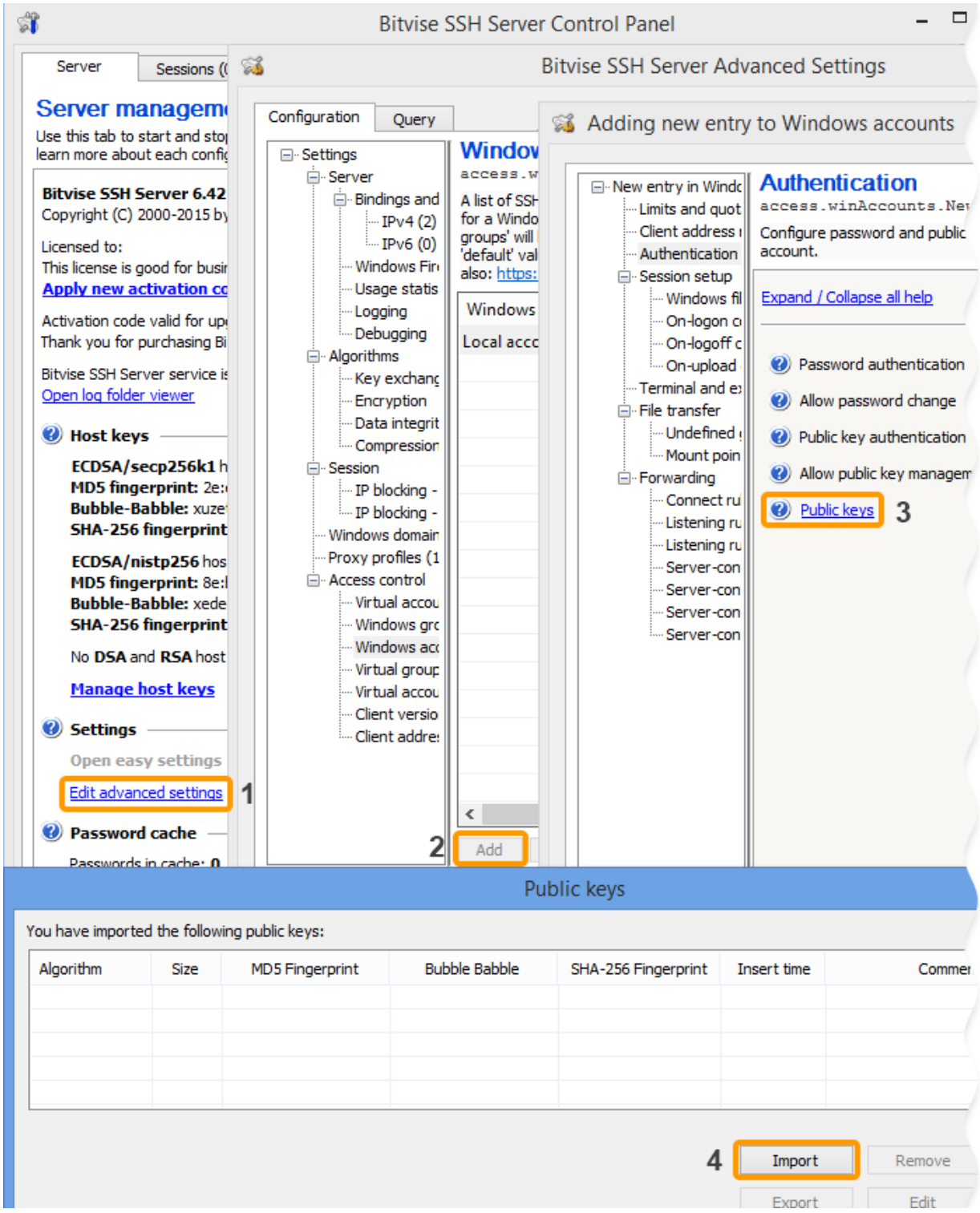
   You can use any SSH client to log into Bitvise SSH Server, as long as it supports SSH protocol version 2.

2. Having ensured that the public key has been saved to a file, transfer it to the machine where Bitvise SSH Server is installed, or to the machine from which you manage the SSH Server remotely using Bitvise SSH Client.

3. Open the **SSH Server Control Panel**, and then, to import the public key into the SSH user's account settings, use either **Open easy settings**:



or **Edit advanced settings**:


Bitvise SSH Server Control Panel
Server
Sessions
Server managem
Use this tab to start and sto
learn more about each confi
Bitvise SSH Server 6.42
Copyright (C) 2000-2015 by
Licensed to:
This license is good for busir
Apply new activation co
Activation code valid for up
Thank you for purchasing Bi
Bitvise SSH Server service is
Open log folder viewer
Host keys
ECDSA/secp256k1 h
MD5 fingerprint: 2e:
Bubble-Babble: xuze
SHA-256 fingerprint
ECDSA/nistp256 hos
MD5 fingerprint: 8e:
Bubble-Babble: xede
SHA-256 fingerprint
No DSA and RSA host
Manage host keys
Settings
Open easy settings
Edit advanced settings
1
Password cache
Passwords in cache: 0
Bitvise SSH Server Advanced Settings
Configuration
Query
Settings
Server
Bindings and
IPv4 (2)
IPv6 (0)
Windows Fir
Usage statis
Logging
Debugging
Algorithms
Key exchang
Encryption
Data integrit
Compressior
Session
IP blocking -
IP blocking -
Windows domair
Proxy profiles (1
Access control
Virtual accou
Windows gr
Windows ac
Virtual group
Virtual accou
Client versio
Client addre
Windov
access.w
A list of SSH
for a Windo
groups' will
'default' val
also: https:
Windows
Local acc
2
Add
Adding new entry to Windows accounts
New entry in Windc
Limits and quot
Client address
Authentication
Session setup
Windows fil
On-logon c
On-logoff c
On-upload
Terminal and e
File transfer
Undefined
Mount poin
Forwarding
Connect ru
Listening ru
Listening ru
Server-con
Server-con
Server-con
Server-con
Authentication
access.winAccounts.Ne
Configure password and public
account.
Expand / Collapse all help
Password authentication
Allow password change
Public key authentication
Allow public key manager
Public keys
3
Public keys
You have imported the following public keys:
Algorithm
Size
MD5 Fingerprint
Bubble Babble
SHA-256 Fingerprint
Insert time
Commer
4
Import
Remove
Export
Edit

Note:

For Windows accounts, Bitvise SSH Server also supports synchronisation with `~/.ssh/authorized_keys`, provided that this feature is enabled in **Advanced SSH Server settings**, under **Access control**. If this feature is enabled, Bitvise SSH Server will check for the existence of the `authorized_keys` file when the user logs out. If the file exists, Bitvise SSH Server will replace all the public keys configured for the user with the keys found in this file.

Remote host access verification and WPS sign-on

1. Verify that login to the remote server can take place, for example:

```
jsmith@local-host$ ssh jsmith@remote-host
Last login: Wed Oct 21 17:22:33 2015 from 192.168.1.2
[Note: SSH did not ask for password.]

jsmith@remote-host$ [Note: You are on remote-host here]
```

Note:

If you are prompted for a password, then public key authentication has failed.

2. If public key authentication is successful, then, if you are using **WPS Link**, create the required host connection and remote host server through **WPS Workbench**. If you are using **WPS Communicate**, sign onto WPS using your private key, via the `SIGNON` statement, for which you need to specify either the **IDENTITYFILE** statement option or the **SSH_IDENTITYFILE** system option, for example:

```
SIGNON <servername> SSH
USERNAME="<username>"
IDENTITYFILE="C:\Users\techwriter\.ssh\wpscommunicate.ppk"
LAUNCHCMD="/home/installs/wps-3.2/bin/wps -dmr ";
```

Alternatively:

```
OPTIONS SSH_IDENTITYFILE="C:\Users\techwriter\.ssh\wpscommunicate.ppk";
SIGNON <servername> SSH
username="<username>"
LAUNCHCMD="/home/installs/wps-3.2/bin/wps -dmr";
```

Note:

You cannot use either **IDENTITYFILE** or **SSH_IDENTITYFILE** if you are using *Passphrase authentication using ssh-agent* (page 58).

Passphrase authentication using ssh-agent

WPS Communicate does not support the reading of private key files that have been saved in encrypted form with a **passphrase**. However, it is still possible to use private key-pairs of this form if you use a keychain agent. We will assume that you are using OpenSSH's **ssh-agent** for UNIX/Linux.

The **ssh-agent** program runs on the client system, acting as a temporary store of private keys in decrypted form. When the SSH client authenticates with a remote host, it can fetch the private key from the agent without needing to prompt the user.

On startup, **ssh-agent** does not hold any keys. These are loaded from disk using the **ssh-add** command, at which point the user enters each key's passphrase to decrypt it. The agent can store multiple keys simultaneously, from which the system will automatically choose the correct key for the remote server.

The following procedure assumes that you have already generated a key-pair with a **passphrase**, and deployed the public key on the remote SSH server (for UNIX/Linux), or on Bitvise SSH Server (for Windows).

Proceed as follows:

1. Start **ssh-agent** by typing the following into your local terminal session:

   ```
   eval $(ssh-agent)
   ```

   This will start the agent program and place it into the background.

   Note:

   The `eval` command tells the shell to run the output of **ssh-agent** as shell commands. Thereafter, processes run by this shell inherit its environment variables and have access to the agent.

2. Now, you need to add your private key to the agent, so that it can manage your key:

   ```
   ssh-add
   ```

   Note:

   When run without arguments, **ssh-add** automatically adds the files `~/.ssh/id_rsa`, `~/.ssh/id_dsa`, `~/.ssh/id_ecdsa`, `~/.ssh/id_ed25519` and `~/.ssh/identity`.

3. You are prompted to enter your **passphrase**:

```
Enter passphrase for /home/demo/.ssh/id_rsa:
Identity added: /home/demo/.ssh/id_rsa (/home/demo/.ssh/id_rsa)
```

Note:

If you wish to be able to connect without a password to one server from within another server, you will need to forward your SSH key information. This will allow you to authenticate to another server through the server to which you are connected, using the credentials on your local host. To start this, **ssh-agent** must be running, and your private key must have been added to the agent (see above). You then need to connect to your first server using the `-A` option. This forwards your credentials to the server for this session:

```
ssh -A username@remote_host
```

From here, you can SSH into any other host that your SSH key is authorised to access. You will connect as if your private SSH key was located on this server.

Note:

WPS automatically detects whether or not **ssh-agent** is running.

Kerberos single sign-on

You can set up Kerberos single sign-on so that you can sign on from WPS to a remote server without directly providing a password or SSH identity file.

This typically requires modifications to the configuration of the SSH daemon to accommodate Kerberos, details of which are outside the scope of this document. You will need to discuss the required modifications with the system administrator of the remote host, as setting up Kerberos authentication is a job for an experienced system administrator.

Kerberos configuration is also required on the client machine.

Before attempting to perform a Kerberos sign-on using WPS, you must be able to perform a Kerberos sign-on to a remote host using a conventional SSH client.

Once you can perform a Kerberos sign-on to a remote host using an external SSH client, you can perform the same sign-on from within WPS. You do not need to specify any authentication information with the `SIGNON` statement, but you should ensure that neither the **IDENTITYFILE** statement option nor the **SSH_IDENTITYFILE** system option is used.

Example sign-on code might resemble:

```
SIGNON <servername> SSH
LAUNCHCMD="/home/installs/wps-3.2/bin/wps -dmr";
```

Linux clients

The `kinit` command will perform a Kerberos sign-on and prompt you for your Kerberos password. This action acquires a ticket which is cached for a period during which the ticket can be securely passed to other hosts as proof of identity, therefore allowing for authentication to those hosts without the need to provide further authentication information:

```
kinit
ssh -o PreferredAuthentications=gssapi-with-mic <hostname>
```

Specifying the `PreferredAuthentications=gssapi-with-mic` option ensures that SSH only attempts GSSAPI authentication and does not, for example, attempt to use public key authentication which might otherwise proceed without prompting you for a password.

Windows clients

It is possible to configure Kerberos single sign-on from WPS on Windows, but there are some restrictions:

- The first is that you **cannot** be a local administrator on your machine. If you are, Windows will not issue the Kerberos Ticket-Granting Ticket necessary for WPS to perform the Kerberos authentication.
- The second is that you need to set a registry key to allow Windows to give out the Ticket-Granting Ticket, even if you are not in the local administrators group. The value `allowtgtsessionkey` under the following key needs to be set to 1:

  ```
  HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Lsa\Kerberos\Parameters
  ```

  Note:

  You may need to add this value to the registry if it is not already present.

  Note:

  The above restrictions do not apply to PuTTY, so, if you can log on using PuTTY but not WPS, then one of the above restrictions is likely to be the cause.

Environment variables

Environment variables may need to be configured to provide access to third-party software being used with a particular WPS installation.

Using WPS with third-party applications such as database servers, requires that the environment information is available at WPS start up, such as:

- `LD_LIBRARY_PATH` or `LIBPATH` on Linux or Unix systems pointing to client libraries
- `ODBCSYSINI` pointing to the unixODBC client libraries
- `PATH`, for example pointing to the client libraries on Windows, or application directories on Linux or Unix systems.

The WPS recommended method for setting these environment variables is described in the `install-EN`.

Once the required environment variables have been set up by your system administrator, you can test the link server connection using `PROC OPTIONS`:

```
PROC OPTIONS;
RUN;
```

Legal Notices

Trademarks

WPS and World Programming are registered trademarks or trademarks of World Programming Limited in the European Union and other countries. (r) or ® indicates a Community trademark.

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

All other trademarks are the property of their respective owner.

General Notices

World Programming Limited is not associated in any way with the SAS Institute.

WPS is not the SAS System.

The phrases "SAS", "SAS language", and "language of SAS" used in this document are used to refer to the computer programming language often referred to in any of these ways.

The phrases "program", "SAS program", and "SAS language program" used in this document are used to refer to programs written in the SAS language. These may also be referred to as "scripts", "SAS scripts", or "SAS language scripts".

The phrases "IML", "IML language", "IML syntax", "Interactive Matrix Language", and "language of IML" used in this document are used to refer to the computer programming language often referred to in any of these ways.

WPS includes software developed by third parties. More information can be found in the THANKS or acknowledgments.txt file included in the WPS installation.