◆ ◆ ◆ CHAPTER 2

# Setting Up SSH for Centralized Administration

GlassFish Server uses secure shell (SSH) to ensure that clusters that span multiple hosts can be administered centrally. To perform administrative operations on GlassFish Server instances that are remote from the domain administration server (DAS), the DAS must be able to communicate with those instances. If an instance is running, the DAS connects to the running instance directly. For example, when you deploy an application to an instance, the DAS connects to the instance and deploys the application to the instance.

However, the DAS cannot connect to an instance to perform operations on an instance that is not running, such as creating or starting the instance. For these operations, the DAS uses SSH to contact a remote host and administer instances there. SSH provides confidentiality and security for data that is exchanged between the DAS and remote hosts.

**Note –** The use of SSH to enable centralized administration of remote instances is optional. If SSH is not practicable in your environment, you can administer remote instances locally.

The following topics are addressed here:

# About SSH for Centralized Administration

In a typical GlassFish Server deployment, the DAS acts as the SSH client, and hosts where instances reside act as SSH servers. The SSH Server Daemon `sshd` must be running on hosts where instances reside, but is not required to be running on the DAS host. The DAS uses its own SSH client for communicating with hosts where instances reside. However, to generate keys and test SSH setup, a native SSH client must be installed on the DAS host.

## Determining Whether to Use SSH for Centralized Administration

The use of SSH to enable centralized administration of remote instances is optional and is required only for specific operations. Instances local to the DAS can be administered without SSH. If SSH is not practicable in your environment, you can administer remote instances locally.

The requirements for SSH configuration and user management are different for each operating system on which GlassFish Server is supported. Therefore, the use of SSH for centralized administration involves using SSH tools to configure SSH on the operating system that you are using. Before setting up a GlassFish Server cluster, use the following considerations to determine whether to use SSH:

- If you are planning a large cluster of many instances, consider setting up SSH to enable centralized administration of the cluster. SSH simplifies the administration of the cluster by enabling you to perform all administrative operations on the cluster and its instances from the DAS. However, setting up SSH might require much effort, especially on Windows systems.
- If you are planning a small cluster of few instances, consider whether setting up SSH requires more effort than logging in to individual hosts to administer remote instances locally.

How you administer instances and the nodes on which they resides varies depending on whether SSH is available. The following table provides cross-references to instructions for administering nodes and instances depending on whether SSH is available.

| SSH Availabilty | Node Administration Instructions | Instance Administration Instructions |
|---|---|---|
| Available | "Creating, Listing, Testing, and Deleting `SSH` Nodes" on page 50 | "Administering GlassFish Server Instances Centrally" on page 78 |
| Not available | "Creating, Listing, and Deleting `CONFIG` Nodes" on page 54 | "Administering GlassFish Server Instances Locally" on page 88 |

## Obtaining SSH Software

On UNIX and Linux systems, SSH software is typically installed as part of the base operating system.

However, on Windows systems, you must install one of the following SSH providers:

- Cygwin (`http://www.cygwin.com/`) release 1.7.6
- MKS Toolkit for Developers (`http://www.mkssoftware.com`) release 9.2

## Determining the SSH User

Before setting up SSH, decide which SSH user GlassFish Server will use when connecting to remote systems. For the following reasons, administration is simplest if the SSH user is the user that starts the DAS:

- For public key authentication, the user that starts the DAS must be able to read the SSH user's private key file.
- Remote instances are started as the SSH user.
- By default, the DAS assumes that the SSH user is the user that is running the DAS.

# Setting Up Cygwin SSH on Windows

Set up Cygwin SSH on the DAS host and on all hosts where instances in your cluster will reside.

The following topics are addressed here:



## ▼ To Download and Install Cygwin

For centralized GlassFish Server administration, a basic Cygwin installation that includes the SSH client and the SSH server daemon `sshd` is sufficient. The default installation options are sufficient to create such a basic installation.

1. **Log in as a user with Administrator privileges.**
2. **Create the folder `C:\cygwin`.**
3. **From the Cygwin home page (`http://www.cygwin.com/`), download and save the `setup.exe` file to your desktop.**

4. **Run the `setup.exe` file.**

5. **Select the default for the following options:**
   - Install from Internet
   - Install Root Directory: `C:\cygwin`
   - Install for All Users

6. **Specify a folder for the local package directory that is not the Cygwin root folder, for example, `C:\cygwin\packages`.**

7. **Specify the connection method.**
   For example, if the host is connected to the Internet through a proxy server, specify the proxy server.

8. **Select the mirror site from which to download the software.**

9. **Select the `openssh` package for installation.**
   a. **Under the Net category, search for `openssh`.**
   b. **Locate the `openssh` package and click Skip.**
      The package is selected.
   c. **Click Next.**
      Any unsatisfied dependencies are listed.

10. **Leave the Select Required Packages option selected and click Next**
    The packages are installed.

11. **Click Finish.**

**See Also** For detailed information about installing Cygwin, see"Internet Setup" in *Cygwin User's Guide* (`http://cygwin.com/cygwin-ug-net/setup-net.html#internet-setup`).

## ▼ To Set the Path for Windows and for the Cygwin Shell

To enable GlassFish Server tools to find commands for SSH, each user's path for Windows and for the Cygwin shell must contain the following directories:

- The Cygwin `bin` directory, for example `C:\cygwin\bin`
- The `bin` directory of the JDK software

1. **Log in as a user with Administrator privileges.**

   Logging in as a user with Administrator privileges ensures that the change applies to all users.

2. **In the System Information control panel, click Advanced→Environment Variables.**

3. **Add the following directories to the Path environment variable:**

   - The Cygwin `bin` directory, for example `C:\cygwin\bin`
   - The `bin` directory of the JDK software

## ▼ To Set the Home Directory for the Cygwin User

To ensure that all GlassFish Server commands can run without errors, each user's home directory for Cygwin and Windows must be the same directory.

1. **Log in as a user with Administrator privileges.**

2. **In the `/etc/passwd` file, edit the home directory setting for the SSH user to specify the user's home directory for Windows.**

## ▼ To Configure and Start the Cygwin SSH Server Daemon `sshd`

**Before You Begin** Ensure that the following prerequisites are met:

- A user account is created for each user that will log in to the host through SSH.
- A password is set for each user account.

  The SSH server daemon `sshd` disallows authentication of any user for whose account a password is not set.

1. **Double-click the Cygwin icon.**

   A Cygwin terminal is started.

2. **If necessary, set the password for your user account.**

   a. **Run the `passwd` command as follows:**

      `$ passwd` *user-name*

      *user-name*
      The user name for your account.

   b. **Type a password.**

The password for your Windows account is also set.

3 **Configure SSH on the host.**

a. **Run the `ssh-host-config` command.**

```
$ ssh-host-config
```

b. **Set the following options to `yes`:**

- `StrictModes`
- `PubkeyAuthentication`

c. **For the value of `CYGWIN` for the daemon, specify `C:\cygwin`.**

The settings for the SSH server daemon `sshd` are found in the file `/etc/ssh_config`, which can also be accessed as `/cygdrive/c/cygwin/etc/sshd_config`.

4 **Start the SSH server daemon `sshd`.**

```
$ net start sshd
```

5 **Confirm that the SSH server daemon `sshd` is running.**

```
$ cygrunsrv --query sshd
 Service             : sshd
 Display name        : CYGWIN sshd
 Current State       : Running
 Controls Accepted   : Stop
 Command             : /usr/sbin/sshd -D
```

**Next Steps** After you have completed the setup of SSH on a host, test the setup on the host as explained in "Testing the SSH Setup on a Host" on page 36.

# Setting Up the MKS Toolkit on Windows

Set up the MKS Toolkit on the DAS host and on all hosts where instances in your cluster will reside.

The following topics are addressed here:

## To Install the MKS Toolkit

For centralized GlassFish Server administration, the default installation of the MKS Toolkit is sufficient.

- **Follow the instructions in the MKS Toolkit product documentation to install OpenSSH from the MKS Toolkit with default installation options.**

**See Also** For detailed information about installing MKS Toolkit, see"Installing MKS Toolkit " in *MKS Toolkit v9.4 Release Notes* (`http://www.mkssoftware.com/docs/rn/relnotes_tk94.asp#install`).

## To Set the Path for Windows and for the MKS Toolkit Shell

To enable GlassFish Server tools to find commands for SSH, each user's path for Windows and for the MKS Toolkit shell must contain the following directories:

- The MKS Toolkit `bin` directory, for example `C:\Program Files\MKS Toolkit\mksnt`
- The `bin` directory of the JDK software

The MKS Toolkit installer automatically adds the MKS Toolkit `bin` directory to the path. However, you must add the `bin` directory of the JDK software to the path yourself.

1. **Log in as a user with Administrator privileges.**

   Logging in as a user with Administrator privileges ensures that the change applies to all users.

2. **In the System Information control panel, click Advanced→Environment Variables.**

3. **Add the `bin` directory of the JDK software to the Path environment variable.**

## To Set the Home Directory for the MKS Toolkit User

The SSH Server Daemon `sshd` locates a user's home directory from the configuration in the user database, not from environment variables such as `HOME`. To ensure that all GlassFish Server commands can run without errors, each SSH user must be configure to have a home directory.

Each user on a Windows host where the MKS Toolkit is set up potentially has two home directories:

- **Windows home directory.** GlassFish Server commands, which are run in a Windows command window, use the Windows home directory.

- **MKS home directory.** SSH commands, which are run in an MKS Toolkit shell such `bash` or `ksh`, use the MKS Toolkit home directory.

If these home directories are different, GlassFish Server and SSH each locate a user's `.ssh` directory in different directories. To simplify the set up of SSH, configure each user's home directory for the MKS Toolkit and Windows to be the same directory. A disadvantage of this approach is that the MKS Toolkit home directory has spaces in its path name. Spaces in path names are cumbersome in the UNIX environment.

**1 Compare the pairs of settings for Windows and the MKS Toolkit that are listed in the following table.**

| Windows Environment Variable | MKS Toolkit Field |
|---|---|
| `HOMEPATH` | Home Directory |
| `HOMEDRIVE` | Home Directory Drive |

**a. In a Windows command window, determine the values of the following environment variables:**

- `HOMEPATH`
- `HOMEDRIVE`

**b. In an MKS Toolkit shell, determine the current settings of the following fields for the user:**

- Home Directory
- Home Directory Drive

```
$ userinfo user-name
```

*user-name*
The user name for the user whose home directory you are setting, for example `Administrator`.

**2 If the settings do not match, update setting of each MKS Toolkit field to match its corresponding Windows environment variable.**

If the settings match, no further action is required.

To update the settings, run the following command in an MKS Toolkit shell:

```
$ userinfo -u -fHomeDirDrive:"drive" -fHomeDir:"path" user-name
```

*drive*
The drive identifier of the disk drive on which the user's Windows home directory resides, for example, `C:`.

*path*
The path to the user's Windows home directory, for example, `\Documents and Settings\Administrator`.

*user-name*
The user name for the user whose home directory you are setting, for example `Administrator`.

---

**Note –** Do not set the `HOME` environment variable explicitly. If Home Directory and Home Directory Drive are set correctly, the `HOME` environment variable specifies the correct path by default.

---

**3 In an MKS Toolkit shell, confirm that the settings were updated.**

```
$ userinfo user-name
```

*user-name*
The user name for the user whose home directory you are setting, for example `Administrator`.

**4 Log out of the host and log in to the host again.**

**5 Confirm that the home directories are the same as explained in Step 1.**

**Example 2–1** Setting the Home Directory for the MKS Toolkit User

This example sets the home directory for the MKS Toolkit user `Administrator` to `C:\Documents and Settings\Administrator`.

```
$ userinfo -u -fHomeDirDrive:"C:"
-fHomeDir:"\Documents and Settings\Administrator" Administrator
```

## ▼ To Configure and Start the MKS Toolkit SSH Server Daemon `sshd`

---

**Note –** Do *not* set the command shell to `cmd.exe`. The use of SSH for centralized GlassFish Server administration requires a shell in the style of a UNIX shell.

---

**1 From the Programs menu, choose MKS Toolkit →Configuration→Configuration Information.**

**2 Enable password authentication and strict modes.**

**a. Click the Secure Shell Service tab.**

b. Select the Password Authentication option.

c. Click Advanced settings.

d. Click the Login tab.

e. Deselect the Strict Modes option.

3 If you are using SSH key-file authentication, enable `MKSAUTH` password authentication.

a. Click the Authentication tab.

b. Under Enable/Disable Password using MKSAUTH, type the user's password and click the Enable.

4 Start the SSH server daemon `sshd`.

5 Confirm that the SSH server daemon `sshd` is running.

```
$ service query MKSSecureSH
Name:           MKS Secure Shell Service
Service Type:   WIN32_OWN_PROCESS
Current State:  RUNNING
Controls Accepted:      ACCEPT_STOP
Check Point:    0
Wait Hint:      0
Start Type:     AUTO_START
Error Control:  IGNORE
Path:           "C:\Program Files\MKS Toolkit\bin\secshd.exe"
Dependency:     NuTCRACKERService
Dependency:     tcpip
Service Start Name:     LocalSystem
```

**Next Steps** After you have completed the setup of SSH on a host, test the setup on the host as explained in "Testing the SSH Setup on a Host" on page 36.

# Setting Up SSH on UNIX and Linux Systems

Setting up SSH on UNIX and Linux systems involves verifying that the SSH server daemon `sshd` is running and, if necessary, starting this daemon. Set up SSH on the DAS host and on all hosts where instances in your cluster will reside.

On UNIX and Linux systems, SSH software is typically installed as part of the base operating system. If SSH is not installed, download and install the appropriate OpenSSH (`http://www.openssh.com/`) SSH package for your operating system.

How to set up SSH on UNIX and Linux systems depends on the flavor of the operating system that you are running, as explained in the following sections:

## ▼ To Set Up SSH on Oracle Solaris Systems

1 **Ensure that the following options in the configuration file `/etc/ssh/sshd_config` are set to yes:**

- `StrictModes`
- `PubkeyAuthentication`

2 **Determine if the SSH server daemon `sshd` is running.**

```
$ /usr/bin/svcs status ssh
```

3 **If the SSH server daemon `sshd` is not running, start this daemon.**

If the daemon is running, no further action is required.

```
$ /usr/sbin/svcadm enable ssh
```

**Example 2–2** Determining if the `sshd` Daemon Is Running on an Oracle Solaris System

This example confirms that the SSH server daemon `sshd` is running on an Oracle Solaris system.

```
$ svcs status ssh
 STATE          STIME    FMRI
 online         Dec_14   svc:/network/ssh:default
 online         14:10:10 svc:/network/nfs/status:default
```

**See Also** `svcs`(1)

**Next Steps** After you have completed the setup of SSH on a host, test the setup on the host as explained in "Testing the SSH Setup on a Host" on page 36.

## ▼ To Set Up SSH on MacOS Systems

1 **Open System Preferences and click Sharing.**

The Sharing window opens.

2 **Ensure that Remote Login is selected in the Service list.**

**Next Steps** After you have completed the setup of SSH on a host, test the setup on the host as explained in "Testing the SSH Setup on a Host" on page 36.

### ▼ To Set Up SSH on Linux systems

1 **Ensure that the following options in the configuration file `/etc/ssh/sshd_config` are set to `yes`:**

- `StrictModes`
- `PubkeyAuthentication`

2 **Determine if the SSH server daemon `sshd` is running.**

```
$ /sbin/service sshd status
```

3 **If the SSH server daemon `sshd` is not running, start this daemon.**

If the daemon is running, no further action is required.

```
$ /sbin/service sshd start
```

**Example 2–3** Determining if the `sshd` Daemon Is Running on a Linux System

This example confirms that the SSH server daemon `sshd` is running on a Linux system.

```
$ /sbin/service sshd status
openssh-daemon (pid  2373) is running...
```

**Next Steps** After you have completed the setup of SSH on a host, test the setup on the host as explained in "Testing the SSH Setup on a Host" on page 36.

## Testing the SSH Setup on a Host

After setting up SSH on a host, test the setup to ensure that you can use SSH to contact the host from another host. Testing the SSH setup on a host verifies that the SSH server daemon `sshd` is running and that the SSH user has a valid user account on the host.

If you cannot use SSH to contact the host, troubleshoot the SSH setup before setting up SSH user authentication.

### ▼ To Test the SSH Setup on a Host

1 **From another host, use SSH to log in into the host that you are testing as the SSH user.**

```
$ ssh -l user-name host-name
```

*user-name*
The user name for the SSH user's account on the host.

*host-name*
The host name of the host that you are logging in to.

**2 In response to the prompt, type your password.**

If this step succeeds, your setup of SSH is complete.

**Troubleshooting** If your connection is refused, the SSH server daemon `sshd` is not running and you must start the daemon. For instructions, see the following sections:

- "To Configure and Start the Cygwin SSH Server Daemon `sshd`" on page 29
- "To Configure and Start the MKS Toolkit SSH Server Daemon `sshd`" on page 33
- "To Set Up SSH on Oracle Solaris Systems" on page 35

If your connection is accepted, but you cannot log in, ensure that the SSH user has a valid user account on the host.

**Next Steps** After testing the SSH setup, set up SSH user authentication to enable SSH to authenticate users without prompting for a password. For more information, see "Setting Up SSH User Authentication" on page 37.

# Setting Up SSH User Authentication

When a GlassFish Server subcommand uses SSH to log in to a remote host, SSH must be able to authenticate the SSH user without prompting for anything. If SSH prompts the SSH user, the subcommand fails. Setting up SSH user authentication ensures that this requirement is met.

Before setting Up SSH user authentication, determine the authentication scheme that you want to use. If SSH is already deployed at your site, the authentication scheme to use might already be chosen for you.

The following table lists the authentication schemes that GlassFish Server supports. The table also lists the advantages and disadvantages of each authentication scheme.

| Authentication Scheme | Advantages | Disadvantages |
|---|---|---|
| Public key without encryption | GlassFish Server provides tools to simplify set up. | SSH must be configured to locate users' key files in the correct location. File access permissions for key files and the directory that contains the key files must be set correctly. |

| Authentication Scheme | Advantages | Disadvantages |
|---|---|---|
| Public key with passphrase-protected encryption | This scheme is more secure than public key authentication without encryption. | SSH must be configured to locate users' key files in the correct location. File access permissions for key files and the directory that contains the key files must be set correctly. For each SSH user, password aliases are required. |
| Password | No SSH configuration is required to locate key files or to ensure that file access permissions are correct. | For each SSH user, password aliases are required. |

The following topics are addressed here:


- "To Set Up Public Key Authentication Without Encryption" on page 38
- "To Set Up Encrypted Public Key Authentication" on page 40
- "To Set Up Password Authentication" on page 43


## ▼ To Set Up Public Key Authentication Without Encryption

Use the `setup-ssh` subcommand in local mode to set up public key authentication without encryption. This subcommand enables you to set up public key authentication on multiple hosts in a single operation.

The `setup-ssh` subcommand generates a key pair and distributes the public key file to specified hosts. The public key file is protected only by the file system's file access permissions. If you require additional security, set up public key authentication with passphrase-protected encryption as explained in "To Set Up Encrypted Public Key Authentication" on page 40.

**Before You Begin** Ensure that the following prerequisites are met:

- SSH is set up on each host where you are setting up public key authentication. For more information, see the following sections:
  - "Setting Up Cygwin SSH on Windows" on page 27
  - "Setting Up the MKS Toolkit on Windows" on page 30
  - "Setting Up SSH on UNIX and Linux Systems" on page 34
- Only the SSH user has write access to the following files and directories on each host where you are setting up public key authentication:
  - The SSH user's home directory
  - The `~/.ssh` directory
  - The `authorized_key` file

If other users can write to these files and directories, the secure service might not trust the `authorized_key` file and might disallow public key authentication.

**1 Generate an SSH key pair and distribute the public key file to the hosts where you are setting up public key authentication.**

**Note –** Only the options that are required to complete this task are provided in this step. For information about all the options for setting up an SSH key, see the `setup-ssh`(1) help page.

```
asadmin> setup-ssh --generatekey=true host-list
```

*host-list*
A space-separated list of the names of the hosts where the SSH public key is to be distributed.

After generating the SSH key pair, the subcommand uses SSH to log in to each host in *host-list* as the SSH user to distribute the public key. Each time a password is required to log in to a host, you are prompted for the SSH user's password.

**2 In response to each prompt for a password, type the SSH user's password.**

**Example 2–4** Setting Up Public Key Authentication Without Encryption

This example generates and sets up an SSH key for the user `gfuser` on the hosts `sua01` and `sua02`.

```
asadmin> setup-ssh --generatekey=true sua01 sua02
Enter SSH password for gfuser@sua01>
Created directory /home/gfuser/.ssh
/usr/bin/ssh-keygen successfully generated the identification /home/gfuser/.ssh/id_rsa
Copied keyfile /home/gfuser/.ssh/id_rsa.pub to gfuser@sua01
Successfully connected to gfuser@sua01 using keyfile /home/gfuser/.ssh/id_rsa
Copied keyfile /home/gfuser/.ssh/id_rsa.pub to gfuser@sua02
Successfully connected to gfuser@sua02 using keyfile /home/gfuser/.ssh/id_rsa
Command setup-ssh executed successfully.
```

**Next Steps** After setting up public key authentication, test the setup by using `ssh` to log in as the SSH user to each host where the public key was distributed. For each host, log in first with the unqualified host name and the with the fully qualified name. If SSH does *not* prompt for password, public key authentication is set up correctly on the host.

If you are prompted for a password, verify that the public key file was copied correctly to the SSH user's `.ssh` directory.

**Troubleshooting** Setup might fail because file access permissions in the SSH user's home directory are too permissive. In this situation, ensure that the file access permissions in the SSH user's home directory meet the requirements for performing this procedure.

If you have set the file access permissions in the SSH user's home directory correctly, setup might still fail if you are using the MKS Toolkit. In this situation, correct the problem in one of the following ways:

- On each remote host, copy the public key file to the SSH user's `~/.ssh` directory and import the file. To import the file, select the Secure Service tab in the MKS configuration GUI and click Passwordless.
- Disable strict modes.

**See Also**
- "Setting Up Cygwin SSH on Windows" on page 27
- "Setting Up the MKS Toolkit on Windows" on page 30
- "Setting Up SSH on UNIX and Linux Systems" on page 34
- `setup-ssh`(1)

You can also view the full syntax and options of the subcommand by typing `asadmin help setup-ssh` at the command line.

## ▼ To Set Up Encrypted Public Key Authentication

Encrypted key file authentication uses an encrypted private key file that is protected with a passphrase. This passphrase must be provided to use the private key to unlock the public key. If you require encrypted public key authentication, you must use the SSH utility `ssh-keygen` to generate an SSH key pair with an encrypted private key. You can then use the `setup-ssh` subcommand to distribute the public key file to specified hosts.

When the SSH user runs a subcommand that requires SSH authentication, SSH must be able to authenticate the user without prompting for the passphrase. To meet this requirement, the passphrase must be stored in a password file that is passed to the `asadmin`(1M) utility each time such a subcommand is run. To prevent the passphrase from being stored in the password file in clear text, create an alias to represent the passphrase and store the alias in the password file.

---

**Note –** Only the options that are required to complete this task are provided in each step. For information about all the options for the commands and subcommands in this task, see their help pages or man pages.

---

**Before You Begin** Ensure that the following prerequisites are met:

- SSH is set up on each host where you are setting up public key authentication. For more information, see the following sections:
  - "Setting Up Cygwin SSH on Windows" on page 27
  - "Setting Up the MKS Toolkit on Windows" on page 30
  - "Setting Up SSH on UNIX and Linux Systems" on page 34

- Only the SSH user has write access to the following files and directories on each host where you are setting up public key authentication:
  - The SSH user's home directory
  - The `~/.ssh` directory
  - The `authorized_key` file

  If other users can write to these files and directories, the secure service might not trust the `authorized_key` file and might disallow public key authentication.

**1 Generate an SSH key pair with an encrypted private key file.**

Use the SSH utility `ssh-keygen`(1) for this purpose.

```
$ ssh-keygen -t type
```

*type*
The algorithm that is to be used for the key and which must be `rsa`, `dsa`, or `rsa1`.

The `ssh-keygen` utility prompts you for a file in which to save the key.

**2 In response to the prompt, accept the default file.**

The `ssh-keygen` utility prompts you for a passphrase.

**3 In response to the prompt, type your choice of passphrase for encrypting the private key file.**

The `ssh-keygen` utility prompts you to type the passphrase again.

**4 In response to the prompt, type the passphrase that you set in Step 3.**

**5 Distribute the public key file to the hosts where you are setting up public key authentication.**

Use the `setup-ssh`(1) `asadmin` subcommand for this purpose.

```
$ asadmin setup-ssh --generatekey=false host-list
```

*host-list*
A space-separated list of the names of the hosts where the SSH public key is to be distributed.

The subcommand uses SSH to log in to each host in *host-list* as the SSH user to distribute the public key. Each time a passphrase or a password is required to log in to a host, you are prompted for the passphrase or the SSH user's password.

**6 In response to each prompt, type the requested information.**

- **In response to each prompt for a passphrase, type the passphrase that you set in Step 3.**
- **In response to each prompt for a password, type the SSH user's password.**

7 **Create an alias for the passphrase that you set in Step 3.**

Use the `create-password-alias`(1) `asadmin` subcommand for this purpose.

`$` **`asadmin create-password-alias`** *alias-name*

*alias-name*
Your choice of name for the alias that you are creating.

The `create-password-alias` subcommand prompts you to type the passphrase for which you are creating an alias.

8 **In response to the prompt, type the passphrase that you set in Step 3.**

The `create-password-alias` subcommand prompts you to type the passphrase again.

9 **In response to the prompt, type the passphrase that you set in Step 3 again.**

10 **Create a plain text file that contains the following entry for the passphrase alias:**

`AS_ADMIN_SSHKEYPASSPHRASE=${ALIAS=`*alias-name*`}`

*alias-name*
The alias name that you specified in Step 7.

---

**Note –** When you run a subcommand that requires SSH authentication, you must pass this file as the `--passwordfile` option of the `asadmin` utility.

---

**Example 2–5** Setting Up Encrypted Public Key Authentication

This example generates an SSH key pair with an encrypted private key for the user `gfadmin` and distributes the public key to the hosts `sj01` and `ja02`. The example also creates an alias that is named `ssh-key-passphrase` for the private key's passphrase.

```
$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/gfadmin/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/gfadmin/.ssh/id_rsa.
Your public key has been saved in /home/gfadmin/.ssh/id_rsa.pub.
The key fingerprint is:
db:b5:f6:0d:fe:16:33:91:20:64:90:1a:84:66:f5:d0 gfadmin@dashost
$ asadmin setup-ssh --generatekey=false sj01 sj02
Key /home/gfadmin/.ssh/id_rsa is encrypted
Enter key passphrase>
Enter SSH password for gfadmin@sj01>
Copied keyfile /home/gfadmin/.ssh/id_rsa.pub to gfadmin@sj01
Successfully connected to gfadmin@sj01 using keyfile /home/gfadmin/.ssh/id_rsa
Successfully connected to gfadmin@sj02 using keyfile /home/gfadmin/.ssh/id_rsa
SSH public key authentication is already configured for gfadmin@sj02
Command setup-ssh executed successfully.
$ asadmin create-password-alias ssh-key-passphrase
```

```
Enter the alias password>
Enter the alias password again>
Command create-password-alias executed successfully.
```

The entry in the password file for the `ssh-key-passphrase` alias is as follows:

```
AS_ADMIN_SSHKEYPASSPHRASE=${ALIAS=ssh-key-passphrase}
```

**Troubleshooting**

Setup might fail because file access permissions in the SSH user's home directory are too permissive. In this situation, ensure that the file access permissions in the SSH user's home directory meet the requirements for performing this procedure.

If you have set the file access permissions in the SSH user's home directory correctly, setup might still fail if you are using the MKS Toolkit. In this situation, correct the problem in one of the following ways:

- On each remote host, copy the public key file to the SSH user's `~/.ssh` directory and import the file. To import the file, select the Secure Service tab in the MKS configuration GUI and click Passwordless.
- Disable strict modes.

**See Also**

- "Setting Up Cygwin SSH on Windows" on page 27
- "Setting Up the MKS Toolkit on Windows" on page 30
- "Setting Up SSH on UNIX and Linux Systems" on page 34
- `asadmin`(1M)
- `create-password-alias`(1)
- `setup-ssh`(1)
- `ssh-keygen`(1)

You can also view the full syntax and options of the subcommands by typing the following commands at the command line:

- `asadmin help create-password-alias`
- `asadmin help setup-ssh`

## ▼ To Set Up Password Authentication

When the SSH user runs a subcommand that requires SSH authentication, SSH must be able to authenticate the user without prompting for the password. To meet this requirement, the SSH user's password must be stored in a password file that is passed to the `asadmin`(1M) utility each time such a subcommand is run. To prevent the SSH user's password from being stored in the password file in clear text, create an alias to represent the password and store the alias in the password file.

**Before You Begin**

Ensure that SSH is set up on each host where you are setting up public key authentication. For more information, see the following sections:

- "Setting Up Cygwin SSH on Windows" on page 27
- "Setting Up the MKS Toolkit on Windows" on page 30
- "Setting Up SSH on UNIX and Linux Systems" on page 34

**1 Create an alias for the SSH user's password.**

---

**Note –** Only the options that are required to complete this task are provided in this step. For information about all the options for creating a password alias, see the `create-password-alias`(1) help page.

---

```
asadmin> create-password-alias alias-name
```

*alias-name*
Your choice of name for the alias that you are creating.

The `create-password-alias` subcommand prompts you to type the password for which you are creating an alias.

**2 In response to the prompt, type the SSH user's password.**

The `create-password-alias` subcommand prompts you to type the password again.

**3 In response to the prompt, type the SSH user's password again.**

**4 Create a plain text file that contains the following entry for the password alias:**

```
AS_ADMIN_SSHPASSWORD=${ALIAS=alias-name}
```

*alias-name*
The alias name that you specified in Step 1.

---

**Note –** When you run a subcommand that requires SSH authentication, you must pass this file as the `--passwordfile` option of the `asadmin` utility.

---

**Example 2–6** Creating an Alias for the SSH User's Password

This example creates an alias that is named `ssh-password` for the SSH user's password.

```
$ asadmin create-password-alias ssh-password
Enter the alias password>
Enter the alias password again>
Command create-password-alias executed successfully.
```

The entry in the password file for the `ssh-password` alias is as follows:

```
AS_ADMIN_SSHPASSWORD=${ALIAS=ssh-password}
```

**See Also**

- "Setting Up Cygwin SSH on Windows" on page 27
- "Setting Up the MKS Toolkit on Windows" on page 30
- "Setting Up SSH on UNIX and Linux Systems" on page 34
- `asadmin`(1M)
- `create-password-alias`(1)

You can also view the full syntax and options of the subcommand by typing the `asadmin help create-password-alias` at the command line.

# Installing and Removing GlassFish Server Software on Multiple Hosts

GlassFish Server software must be installed on all hosts where GlassFish Server will run. How to install GlassFish Server software on multiple hosts depends on the degree of control that you require over the installation on each host.

- If you require complete control over the installation on each host, install the software from a GlassFish Server distribution on each host individually. For more information, see *GlassFish Server Open Source Edition 3.1 Installation Guide*
- If the same set up on each host is acceptable, copy an existing GlassFish Server installation to the hosts. For more information, see "To Copy a GlassFish Server Installation to Multiple Hosts" on page 45.

GlassFish Server also enables you to remove GlassFish Server software from multiple hosts in a single operation. For more information, see "To Remove GlassFish Server Software From Multiple Hosts" on page 46.

The following topics are addressed here:



## ▼ To Copy a GlassFish Server Installation to Multiple Hosts

Use the `install-node` subcommand in local mode to copy an installation of GlassFish Server software to multiple hosts.

**Before You Begin** Ensure that SSH is set up on the host where you are running the subcommand and on each host where you are copying the GlassFish Server software.

- **Run the `install-node` subcommand.**

**Note –** Only the options that are required to complete this task are provided in this step. For information about all the options for copying an installation of GlassFish Server software, see the `install-node`(1) help page.

```
asadmin> install-node host-list
```

*host-list*
A space-separated list of the names of the hosts where you are copying the installation of GlassFish Server software.

**Example 2–7** Copying a GlassFish Server Installation to Multiple Hosts

This example copies the GlassFish Server software on the host where the subcommand is run to the default location on the hosts `sj03.example.com` and `sj04.example.com`.

```
asadmin> install-node sj03.example.com sj04.example.com
Created installation zip /home/gfuser/glassfish2339538623689073993.zip
Successfully connected to gfuser@sj03.example.com using keyfile /home/gfuser
/.ssh/id_rsa
Copying /home/gfuser/glassfish2339538623689073993.zip (81395008 bytes) to
sj03.example.com:/export/glassfish3
Installing glassfish2339538623689073993.zip into sj03.example.com:/export/glassfish3
Removing sj03.example.com:/export/glassfish3/glassfish2339538623689073993.zip
Fixing file permissions of all files under sj03.example.com:/export/glassfish3/bin
Successfully connected to gfuser@sj04.example.com using keyfile /home/gfuser
/.ssh/id_rsa
Copying /home/gfuser/glassfish2339538623689073993.zip (81395008 bytes) to
sj04.example.com:/export/glassfish3
Installing glassfish2339538623689073993.zip into sj04.example.com:/export/glassfish3
Removing sj04.example.com:/export/glassfish3/glassfish2339538623689073993.zip
Fixing file permissions of all files under sj04.example.com:/export/glassfish3/bin
Command install-node executed successfully
```

**See Also** `install-node`(1)

You can also view the full syntax and options of the subcommand by typing `asadmin help install-node` at the command line.

## ▼ To Remove GlassFish Server Software From Multiple Hosts

Use the `uninstall-node` subcommand in local mode to remove GlassFish Server software from multiple hosts.

**Before You Begin** Ensure that the following prerequisites are met:

- SSH is set up on the host where you are running the subcommand and on each host from which you are removing the GlassFish Server software.
- No process is accessing the parent of the base installation directory for the GlassFish Server software or any subdirectory of this directory.
- The configuration of the following items is the same on each host from which you are removing the GlassFish Server software:
  - Parent of the base installation directory for the GlassFish Server software
  - SSH port
  - SSH user
  - SSH key file

● **Run the `uninstall-node` subcommand.**

---

**Note –** Only the options that are required to complete this task are provided in this step. For information about all the options for removing GlassFish Server software, see the `uninstall-node`(1) help page.

---

```
asadmin> uninstall-node host-list
```

*host-list*
A space-separated list of the names of the hosts from which you are removing GlassFish Server software.

### Example 2–8 Removing GlassFish Server Software From Multiple Hosts

This example removes GlassFish Server software on the hosts `sj03.example.com` and `sj04.example.com` from the default location.

```
asadmin> uninstall-node sj03 sj04
Successfully connected to gfuser@sj03.example.com using keyfile /home/gfuser
/.ssh/id_rsa
Successfully connected to gfuser@sj04.example.com using keyfile /home/gfuser
/.ssh/id_rsa
Command uninstall-node executed successfully.
```

**See Also** `uninstall-node`(1)

You can also view the full syntax and options of the subcommand by typing `asadmin help uninstall-node` at the command line.