**Name** update-node-ssh – updates the configuration data of a node

**Synopsis**
```
update-node-ssh [--help]
[--nodehost node-host]
[--installdir install-dir] [--nodedir node-dir]
[--sshport ssh-port] [--sshuser ssh-user]
[--sshkeyfile ssh-keyfile]
[--force={false|true}]
node-name
```

**Description** The `update-node-ssh` subcommand updates the configuration data of a node. This subcommand requires secure shell (SSH) to be configured on the machine where the domain administration server (DAS) is running and on the machine where the node resides. You may run this subcommand from any machine that can contact the DAS.

This subcommand can update any node, regardless of whether the node is enabled for communication over SSH. If the node is not enabled for communication over SSH, the subcommand enables SSH communication for the node and updates any other specified configuration data.

Options of this subcommand specify the new values of the node's configuration data. The default for most options is to leave the existing value unchanged. However, if this subcommand is run to enable SSH communication for a node, default values are applied if any of the following options is omitted:

- `--sshport`
- `--sshuser`
- `--sshkeyfile`

By default, the subcommand fails and the node is not updated if the DAS cannot contact the node's host through SSH. To force the node to be updated even if the host cannot be contacted through SSH, set the `--force` option to `true`.

This subcommand is supported in remote mode only.

**Options**

`--help`
`-?`
Displays the help text for the subcommand.

`--nodehost`
The name of the host that the node is to represent after the node is updated.

`--installdir`
The full path to the parent of the base installation directory of the GlassFish Server software on the host, for example, `/export/glassfish3/`.

`--nodedir`
The full path to the directory that is to contain GlassFish Server instances that are created on the node.

`--sshport`
The port to use for SSH connections to this node. The default depends on whether this subcommand is run to enable SSH communication for the node:

- If the node is already enabled for communication over SSH, the default is to leave the port unchanged.
- If this subcommand is run to enable SSH communication for the node, the default port is 22.

If the `--nodehost` is set to `localhost`, the `--sshport` option is ignored.

`--sshuser`
The SSH user that is to run the process for connecting to this node. The default depends on whether this subcommand is run to enable SSH communication for the node:

- If the node is already enabled for communication over SSH, the default is to leave the user unchanged.
- If this subcommand is run to enable SSH communication for the node, the default is the user that is running the DAS process.

To ensure that the DAS can read this user's SSH private key file, specify the user that is running the DAS process. If the `--nodehost` is set to `localhost`, the `--sshuser` option is ignored.

`--sshkeyfile`
The absolute path to the SSH private key file for user that the `--sshuser` option specifies. This file is used for authentication to the `sshd` daemon on the node.

The path to the key file must be reachable by the DAS and the key file must be readable by the DAS. The path may contain Java properties of the form `${`*prop.name*`}`.

The default depends on whether this subcommand is run to enable SSH communication for the node:

- If the node is already enabled for communication over SSH, the default is to leave the key file unchanged.
- If this subcommand is run to enable SSH communication for the node, the default is the a platform-dependent key file in the user's `.ssh` directory, for example:
    - `id_rsa`
    - `id_dsa`
    - `identitylocation`

`--force`
Specifies whether the node is updated even if validation of the node's parameters fails. To validate a node's parameters, the DAS must be able to contact the node's host through SSH. Possible values are as follows:

`false`
: The node is not updated if validation of the node's parameters fails (default).

`true`
: The node is updated even if validation of the node's parameters fails.

## Operands

*node-name*
: The name of the node to update. The node must exist. Otherwise, an error occurs.

## Examples

**EXAMPLE 1** Updating the Host That a Node Represents

This example updates the host that the node `lssh` represents to `sj04`.

```
asadmin> update-node-ssh --nodehost sj04 lssh

Command update-node-ssh executed successfully.
```

**EXAMPLE 2** Forcing the Update of a Node

This example forces the update of the node `sj01` to enable the node to communicate over SSH.

```
asadmin> update-node-ssh --force sj01
Warning: some parameters appear to be invalid.
Could not connect to host sj01 using SSH.
Could not authenticate. Tried authenticating with specified key at
/home/gfuser/.ssh/id_rsa
Continuing with node update due to use of --force.

Command update-node-ssh executed successfully.
```

## Exit Status

| | |
|---|---|
| 0 | command executed successfully |
| 1 | error in executing the command |

## See Also

`create-node-config`(1), `create-node-ssh`(1), `delete-node-config`(1), `delete-node-ssh`(1), `install-node`(1), `list-nodes`(1), `uninstall-node`(1)

`asadmin`(1M)