Installing MKS Toolkit on Clustered Nodes

For details on Windows clustered servers, see `http://www.microsoft.com/windows2000/technologies/clustering/default.asp`.

Normally, to install MKS Toolkit on the individual systems (or nodes) that make up a clustered server, you simply install it on each of those systems. If, however, you are planning on using MKS Secure Shell with such a cluster, there are additional steps that need to be taken.

Note This assumes that you have a licence for each copy of MKS Toolkit that you install.

For more information on MKS Secure Shell, see the *MKS Toolkit Connectivity Solutions Guide* as well as the **secsh** reference page (and related pages) in the online *MKS Toolkit Utilities Reference*.

Why are these additional steps necessary? The answer lies in the fact that MKS Secure Shell uses a per host identification mechanism. When you install MKS Toolkit on a system within a cluster, it creates an individual host key for that system. As a result, each node in a cluster has its own host key. Now, this is not a not problem when the MKS Secure Shell client explicitly connects to the IP of an individual system in the cluster. However, if the client connects to the cluster's IP address (that is, the IP address assigned to the entire cluster), it is, in truth, connecting to a specific system within the cluster. The potential problem lies in the fact that the clustered server may change the system the client is connected to from time to time. When this happens, the client finds itself connected to a system with a different host key and if `StrictHostKeyChecking` is enabled for that client, an error results.

The are three possible solutions to this problem:

1. Disable `StrictHostKeyChecking` for the clustered server. You can do this with the MKS Toolkit command:

   ```
   secsh -o StrictHostKeyChecking=no clustername
   ```

   where *clustername* is the name (or address) of the clustered server.

2. Use the MKS Toolkit control panel applet to disable `StrictHostKeyChecking` on a per host, per user, or per connection basis. This must be done for all systems that will access the cluster via a MKS Secure Shell client. To do this, create

a special host pattern using the control panel applet that matches all systems in the clustered server and set `StrictHostKeyChecking=No` for the matching systems.

3. Synchronize all nodes of the clustered server to use the same host key. This need only be done once when MKS Toolkit is installed on each system in the cluster and can be accomplished using an automated solution such as the one described in “The mksclust Solution” below.

Note The location of the host key file depends upon on which version of MKS Toolkit is being installed. MKS Toolkit 8.7 and later store the key in the registry, and this is the most “cluster friendly” place for it to be. Earlier versions of MKS Toolkit store the key in a file under `$ROOTDIR/etc` and the location differs by version.

Which solution you use depends upon your needs. For example, if you want to automate the as part of the MKS Toolkit installation, you would choose solution 1 or 3 because solution 2 requires interaction with the control panel applet. One case where this automation would be important would be if you are licensed to redistribute MKS Toolkit (with the connectivity option) with your application. And if high security is a must, solution 3 is the way to go because it does not involve disabling `StrictHostKeyChecking`, which would leave your client vulnerable to what are commonly known as a “man-in-the-middle” attacks.

The mksclust Solution

If you have decided that synchronizing the host keys on all nodes of the clustered server is the preferred solution for your needs, MKS provides a utility named **`mksclust`** that you can compile and use.

Note The **`mksclust`** utility only works with MKS Toolkit 8.7 and later. If you are using an earlier version, contact MKS for a solution tailored to your version.

You can find the source code for **`mksclust`** (`mksclust.cpp)` on the MKS FTP site using the following URL:

```
ftp://ftp.mks.com/pub/support/tk/reskit/cluster.zip
```

After unzipping `cluster.zip` to obtain `mksclust.cpp`, compile that file with the appropriate command. For version 6 of Microsoft Visual C++, use:

```
cl -GX -Ox -DDEBUG mksclust.cpp
```

For version 7.x of Microsoft Visual C++, use

```
cl -GX -Ox -DDEBUG mksclust.cpp -link \
/delayload:"clustapi.dll"
```

Note Version 7.1 of Microsoft C++, generates a linker warning due to a bug in the linker that causes `#pragma comment(linker, ...)` not to be accepted.

The `-DDEBUG` option in both of these commands produces an executable that generates a running commentary when it is run.

You now have a working **`mksclust`** utility (`mksclust.exe`). How you use this utility depends on whether or not MKS Toolkit is already installed on the cluster's nodes and, if not, how you are installing MKS Toolkit:

- When MKS Toolkit is already installed on all nodes of the cluster, run **`mksclust`** on any single node to copy that node's host key to the other nodes in the cluster.
- When you have added a new node to an existing cluster with MKS Toolkit installed, first run **`mksclust`** on one of the pre-existing nodes to ensure that the new node is assigned the synchronized host key and then install MKS Toolkit on the new node (if necessary).
- When you are installing MKS Toolkit for the first time on a cluster's nodes, first install MKS Toolkit on a single node and run **`mksclust`** on that node to assign that node's host key to the remaining nodes. Next install MKS Toolkit on the remaining nodes. Because the host key is already set when MKS Toolkit is installed, the host keys for these systems are not changed.
- When MKS Toolkit installation is a sub-installation of your application installation, modify the application installation to run **`mksclust`** both before and after installing MKS Toolkit. This ensures that all nodes in the cluster are updated correctly under all possible circumstances.

Finally, when you evict (that is, remove) a node from a cluster with synchronized host keys, first run the command:

```
secsh-keygen -t rsa
```

on the evicted node to generate that systems's new unsynchronized host key and remove the `HKLM/Software/Mortice Kern Systems/etc/ssh/ClusterMaster` value from the system's registry.