Secure Shell (SSH) in HP Systems Insight Manager

Introduction

The purpose of this white paper is to provide an overview of what Secure Shell (SSH) is, demonstrate how it is used in HP Systems Insight Manager, and discuss some of the problems that can be encountered during its usage.

SSH was chosen for remote task execution in HP Systems Insight Manager because of its wide availability, security, and ease of use. In other multi-system management tools such as HP Servicecontrol Manager, proprietary agents are employed on the managed systems. SSH can replace these agents and, in the process, reduce the amount of setup and maintenance required on the managed node.

Since we are dealing with a standard protocol on the managed system, we can only do what the protocol allows. For example, the management agent of HP Servicecontrol Manager provides client-side logging, as well as an API for the central management server (CMS) to install authentication keys on managed systems. With SSH, these two features are notably absent. Logging is confined to the CMS, and the authentication keys have to be pushed to the managed system using a manual method such as sftp.

HP Systems Insight Manager custom commands and command line tools require that SSH be installed and configured on the CMS, as well as on each of the managed systems, in order to work properly. In the next section, we will discuss these features in detail.

Features of HP Systems Insight Manager requiring SSH

All command line tools in HP Systems Insight Manager are executed by the distributed task facility (DTF) using SSH— even those executing on the CMS itself. SSH was used for execution on the CMS for platform independence—multiple native methods are not needed to support Linux®, HP-UX, and Windows®. Tasks can be run the same way across all platforms—they are always executed through SSH.

Custom commands, or application launch tools, come from Insight Manager 7. They are executed on the CMS. When you select a custom command to be executed against a set of managed systems, a process is executed on the CMS. The list of systems is passed to the DTF through an environmental variable. The custom command then does what it was written to do against each target system. The target systems do not necessarily have to be running SSH to function properly. The custom command could operate though another protocol that, for example, network switches understand. Unlike most command line tools, only the CMS has to be running an SSH server to enable custom commands.

Command line tools come from HP Servicecontrol Manager. There are two styles: single-system aware (SSA) and multi-system aware (MSA.) MSA tools function much like custom commands: the tool is run on an execution node (which is usually the CMS), and the target systems are passed by an environmental variable. The tool is then responsible for communicating with the managed systems using whatever protocol it uses. An example of an MSA tool is Software Distributor for HP-UX. The execution node is the system running the Software Distributor service. SSH is required to be running on that node so that the CMS can contact it with information about the software to install and the managed systems on which to install it.

Unlike custom commands and command line tools, SSA tools are run directly on the managed system. The DTF opens an SSH connection with each of the target systems, executes the command over the SSH protocol, and stores any output (valid command output as well as error messages) in the database. This process occurs on each target system that you selected. Because of this, each target system must be running an SSH server. Examples of both MSA and SSA command line tools that ship with HP Systems Insight Manager can be found in Appendix C: Tool examples.

To summarize, the CMS must have an SSH server installed and configured in order to run any custom commands and most MSA command line tools. In addition, each managed system that you want to select as a target for a SSA command line tool must be running a properly configured SSH server.

Now that you know what features require SSH, we will discuss the protocol itself.

SSH overview

Why SSH?

SSH was chosen for a few simple reasons: it provides a way to execute commands and copy files remotely; it has secure authentication mechanisms; it encrypts all data sent over the wire, unlike the traditional UNIX® 'r' services; and most importantly, it is a popular, non-proprietary protocol. The main consideration for choosing SSH was to eliminate the necessity for a proprietary, Java™-based management agent (for example, HP Servicecontrol Manager's **mxagent**) to be installed on each managed node.

SSH is an Internet Engineering Task Force (IETF) draft standard. It was created to replace the UNIX "r" services remsh, rlogin, and rcp. These services provide remote shell, execution, and file copy. Unfortunately, all data passed between the communicating systems using "r" services is unencrypted clear-text. Additionally, the authentication mechanisms are weak and vulnerable to attack—rhost authentication is vulnerable to man-in-the-middle attack, and passwords are passed over the network in clear text. SSH provides a mechanism to verify the identity of the remote system using key-based host authentication, prevents password snooping by using over-the-wire encryption of all communications between the client and server, and provides stronger user authentication methods via public key authentication.

SSH authentication mechanisms

To create a connection, the SSH client first contacts the remote system. Session keys are exchanged, and are then used to encrypt all further communication between the client and server. The remote SSH server then sends its identity, known as the *host key*, to the SSH client for verification.

The first time a connection is made between systems is the only time the connection is vulnerable to a man-in-the-middle attack. Since the identity of the remote system is unknown, there is nothing to compare it to. Generally, SSH clients let you know that the remote host is unknown, show you the fingerprint of the host key, and ask if you would like to accept it. If accepted, the host key of the remote system is stored for comparison in subsequent connections.

Once the identity of the remote system has been verified, the SSH client sends the username of the user who is requesting a login, along with credentials for the user to the remote SSH server. The user is authenticated in one of three ways—using host-based authentication, password authentication, or public key authentication. In the case of host-based authentication, the SSH client sends its host key to the remote SSH server. The remote server then checks its list of trusted hosts and verifies if the SSH client is one of them. If it is, the remote server trusts that the SSH client has already properly authenticated the user, and allows the login to proceed.

If a password is sent, the remote SSH server simply uses the username and password information to try to authenticate the user. The only difference between the way SSH does this and the way "r" services do this is that, with SSH, the password is encrypted when it is transmitted over the network—just like everything else sent over an SSH connection.

Unlike password authentication, the public key authentication mechanism is unique to SSH, and it is the most secure way to log in. A public key is harder to guess than a password, and the mechanism does not require the SSH server to trust that the SSH client has properly authenticated the user. In public key authentication, the SSH client sends the user's public key along with the username. The SSH server then checks the list of authorized keys for the user, and if there is a match, it sends a

message (encrypted with the public key) back to the client. The client then decrypts the message, using its private key, and sends a return message to the server to prove it has the corresponding private key. Once the server receives this confirmation, the authentication is complete.

Now that an encrypted session is open and the user is authenticated, the session can be used to copy files and execute commands.

HP Systems Insight Manager uses version 2 of the SSH protocol, which closes some weaknesses in the original protocol. HP uses RSA algorithms to generate public & private key pairs, names after the inventors Rivest, Shamir and Adleman. HP supplies the OpenSSH version of an SSH server for Windows systems, and use the SSH server built in to other operating systems; other SSH servers compliant with SSH-2 should work with HP Systems Insight Manager, but there can be configuration differences.

SSH files

There are several important files involved in the mechanisms described above. On the client side, the list of known hosts and the public/private key pair used for public key authentication. On the server side, a public key for each user and the host key of the server.

Since HP Systems Insight Manager uses OpenSSH, the locations and filenames described here are specific to OpenSSH.

SSH client configuration directory

Each user who runs the standard OpenSSH client has a configuration directory that the client uses to store these files. On HP-UX and Linux, it is the hidden directory ".ssh" under the user's home directory (for example, `/home/sshuser/.ssh`.) On Windows the directory can usually be found in the user's Documents and Settings directory (for example, `C:\Documents and Settings\sshuser\.ssh`.)

This directory is automatically created by SSH the first time it is needed. The first time a connection is made from a system, the directory is created so that the file `known_hosts` can be created. When **mxagentconfig** in HP Systems Insight Manager is executed against a managed system to set up user authentication, the directory is created so that the key from the CMS can be placed in the user's authorized keys file.

Known hosts

The list of known host keys can be found in the file `known_hosts`. It contains the host keys that the user has accepted. Whenever you connect to an SSH server for the first time from the command line client, the client informs you that it does not know the host, and asks if it can add it:

```
$ ssh peanut
The authenticity of host 'peanut (192.168.0.2)' can't be
established.
RSA key fingerprint is
31:d7:ce:aa:24:c3:42:fe:77:cd:48:80:f6:0e:34:b6.
Are you sure you want to continue connecting (yes/no)?
```

When you accept it, an entry is added to `known_hosts`. If the host key of the SSH server ever changes, when the server is reinstalled for example, or if another system tries to impersonate that server, the given key will not match the known key and the client will not allow the connection to continue:

$ ssh peanut

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
```

```
Someone could be eavesdropping on you right now (man-in-the-middle
attack)!
It is also possible that the RSA host key has just been changed.
The fingerprint for the RSA key sent by the remote host is
31:d7:ce:aa:24:c3:42:fe:77:cd:48:80:f6:0e:34:b6.
Please contact your system administrator.
Add correct host key in /home/sshuser/.ssh/known_hosts to get rid
of this message.
Offending key in /home/sshuser/.ssh/known_hosts:1
RSA host key for peanut has changed and you have requested strict
checking.
Host key verification failed.
```

Public/Private key pair

For public key authentication, a key pair is created and stored in the user's `.ssh` directory. The private key never leaves the client. It is used during authentication to decode messages that the remote SSH server encodes with the matching public key.

The public key is not used by the client—it is stored in the user's `.ssh` configuration directory so it can be copied to remote systems. In fact, if this file is ever lost, it can be regenerated from the private key. Therefore, it mainly exists for convenience.

Key pairs are generally stored with names matching the type of key they are. The private key has no suffix, and the public key is the same name with `.pub` appended. For example, an OpenSSH DSA key pair is stored in the files `id_dsa` and `id_dsa.pub`. An RSA key pair is stored in `id_rsa` and `id_rsa.pub`, and so on.

Authorized keys

The last file in the user's SSH configuration directory that we will discuss is the authorized keys file, `authorized_keys2`. This is the list of keys that is checked when a remote login is being requested using public key authentication. If the key being presented by the remote client is listed in the file, the SSH server uses it to encrypt a challenge for the remote client and then allows it to log in provided the response to the challenge is correct. If the public key is not present, the authentication fails.

This file is generally maintained manually. You generate a key pair, copy the public key to all of the systems you want to log into using password authentication, and then concatenate it to the end of your `authorized_keys2` file on each of those systems. Alternatively, you could have your home directory NFS mounted on each of the systems—then you would only have to update one file.

This can become tedious for a large number of systems, and it requires you to remotely log into each of the systems, copy the key over, and then issue some command to update the key file. Fortunately, HP Systems Insight Manager provides a tool, **mxagentconfig**, that helps simplify this process. This tool is also used by the Install OpenSSH tool that deploys OpenSSH onto a Windows system. **Mxagentconfig** is discussed in the following section.

SSH features in HP Systems Insight Manager

Now that we have seen the HP Systems Insight Manager features that require SSH, as well as an overview of the SSH protocol, we can talk about how these pieces fit together. In the last section we talked a lot about clients and servers. All client actions are performed on behalf of the DTF by a built-in SSH client.

The DTF contains an SSH client that uses the SSH version 2 protocol to perform all of its actions on managed systems. These actions include opening password-authenticated sessions for installing the public key of the DTF in each execution user's authorized keys file, executing management commands on the managed systems, and collecting output from them.

mxagentconfig

Now we will examine what happens when you set up a managed node using *mxagentconfig*. Its purpose is to store the host key of the target system in the `known_hosts` file on the CMS, and to place the public key of the DTF in the user's authorized keys file so that future connections can be made using public key authentication.

First, **mxagentconfig** opens an SSH connection to the specified managed system. This causes the managed system to send its host key, which is verified against the list of known hosts on the CMS. If the host key is unknown, it is added to the list. If a host key is already stored for that node, the key that was sent during this connection is compared to it. If the keys match, the connection is allowed to continue. If it does not match, the connection fails. This failure prevents man-in-the-middle attacks, except for the first time when the host key of the managed system is unknown.

Once the identity of the managed system has been verified, **mxagentconfig** authenticates the specified username using password authentication. A secure ftp (sftp) channel is then opened. This is used to look for the user's SSH directory (`.ssh`) in the user's home directory. If it does not exist, it is created. Then mxagentconfig checks for the existence of the authorized keys file (`authorized_keys2.`)

If it exists, **mxagentconfig** appends the public key of the DTF to the user's authorized keys file. If it does not exist, the authorized keys file is created for the user with the public key of the DTF as its first entry. At this point, the user is configured for public key authentication on the managed system.

Tool execution user

Using **mxagentconfig**, you can set up public key authentication so the DTF can execute tasks for a particular user. But how do you decide which users to set up?

Tools in HP Systems Insight Manager all have the concept of the execution user— the user who runs a tool when it is executed. If this user is not specified in the tool definition file (TDEF), it defaults to whoever is logged in to HP Systems Insight Manager. Therefore, if you log into a Windows CMS as Administrator, for example, any tools you run that do not specify an execution user will run as Administrator.

This is most often a concern when running cross-platform tools. If you log into a Windows CMS and run an RPM query against a Red Hat Linux server, the tool should run as root, not as Administrator. For this reason, the tools delivered with HP Systems Insight Manager generally specify root for Linux and HP-UX, and Administrator for Windows. The general guideline is that **mxagentconfig** should be run for root on Linux and HP-UX managed systems, and Administrator on Windows managed systems.

The concept of execution user is most important with tools that do not specify who to run as. Since these tools run as whoever is logged in, **mxagentconfig** must be run to set up keys for each user who wants to run the tool. In other words, if a certain tool runs as the logged-in user, and an adminuser wants to be able to execute the tool, **mxagentconfig** must be run for adminuser on each managed system the tool is to be run on. This is an important concept in troubleshooting. If you are getting an authentication exception trying to run a tool, be sure that the keys have been set up for Administrator or root, as well as for the user having trouble executing a command.

For more information on execution user, please refer to the online help or manpage for the **mxtool** file.

Known hosts

As the CMS encounters new SSH servers, it automatically adds them to its list of known hosts. Subsequent connections are verified using the stored host key so that it can be checked during future

connections. This does leave the CMS open to a man-in-the-middle attack the first time an SSH connection is made, since the CMS automatically adds it.

This can be mitigated by performing a standard system discovery as soon as the system is installed. Discovery causes SSH connections to be made against each machine to determine what version, if any, of SSH the managed system is running. When that connection is made, an entry is added to the known hosts.

If an SSH server is ever reinstalled on a managed system, it causes the host key to change and the CMS no longer allows connections to it. To resolve this, use **`mxagentconfig -r -n <nodename>`** to delete the keys from the `known_hosts` file.

NOTE: The CMS retains a copy of the `known_hosts` file in memory; simply editing and removing this file while the CMS is running has no impact and the changes are ignored.

Disallowing new keys

In some situations, the system administrator might decide that allowing the CMS to automatically add keys is unacceptable. In this case, add the following line to the file `mx.properties`:

```
MX_SSH_ADD_UNKNOWN_HOSTS=false
```

For more information on changing CMS properties, refer to Appendix B: Changing server properties.

With this option set, the CMS no longer adds keys to the `known_hosts` file; it refuses to connect to an unknown system. There are two ways to use this capability: You can run an initial discovery to create the `known_hosts` file and then set the option— or, you can set the option before initial discovery and create the `known_hosts` file manually.

The easiest way to create a `known_hosts` file manually is to log in, through SSH, to each system from the command line. This process entails the following steps:

1. Stop the CMS.
2. Delete the existing CMS `known_hosts` file.
3. Delete the Administrator's `known_hosts` file.
4. Start the CMS.
5. Log into each system— including the CMS itself— through SSH, instructing it to add it to the Administrator's `known_hosts` file. Be sure that you make a connection using each system's long name (for example, name.domain.com), short name (for example, name), and IP address (for example, 15.1.48.11.)
6. Copy the administrator's `known_hosts` file back to the CMS `known_hosts` file location

This process can, unfortunately, have the same vulnerability as allowing the keys to be added automatically. The only absolutely secure way to create the `known_hosts` file is to physically go to each system and copy the key from there. To do this, repeat the process above, but only log into the local system through SSH. Collect the individual `known_hosts` entry from each machine this way, and then concatenate them together.

Refer to SSH client configuration directory for more information.

File location

The CMS list of known hosts can be found in the `sshtools` subdirectory of the configuration directory. Depending on your platform, this is

```
C:\Program Files\HP\Systems Insight
Manager\config\sshtools\known_hosts
```

on Windows, and

```
/etc/opt/mx/config/sshtools/known_hosts
```

on HP-UX and Linux.

SSH port

Normally, SSH servers listen on TCP port 22. If, for some reason, this needs to be changed, the SSH port that HP Systems Insight Manager uses is configurable. Note that this only changes the port that the CMS uses to initiate SSH sessions. Every SSH server on each managed system must be configured to listen on that port as well. The port is set by setting *MX_SSH_PORT* in the `mx.properties` file.

Example: To change the port to 6450, add the following line to `mx.properties`:

```
MX_SSH_PORT=6450
```

For more information on changing CMS properties, refer to Appendix B: Changing server properties.

Windows SSH server

While HP-UX and most Linux distributions usually ship with OpenSSH already installed, the same is not true of Windows-based operating systems. HP Systems Insight Manager provides a version of OpenSSH to be used with the DTF on Windows systems. This is installed along with the rest of the CMS software when installing the CMS. For managed systems, it can be installed from the management CD or downloaded from HP's website.

The version provided by HP Systems Insight Manager was repackaged to work seamlessly with the install process. It was also modified to provide greater security than other widely-available distributions. Since OpenSSH is part of OpenBSD, it was originally implemented for UNIX-like operating systems. In order to easily port it to Windows, an emulation layer called Cygwin is used.

Cygwin provides a UNIX emulation layer so that UNIX software can be easily ported to Windows. It also has some well-known security problems—it creates world-readable data structures to emulate UNIX processes. In order to make OpenSSH more secure, the version distributed with HP Systems Insight Manager contains a modified Cygwin compatibility layer that restricts access to these data structures to members of the Administrator's group. Because of this, when HP Systems Insight Manager's version of OpenSSH is used, only Windows Administrators can log into the Windows system via SSH.

Cygwin mounts

To find where OpenSSH looks for certain files, you first need to determine where they are stored. The Unix files of concern are `/etc/passwd`, `/etc/group`, and `/home/hpsimuser`.

Cygwin emulates a UNIX environment. In order to locate files such as /etc/passwd and /etc/group, and the user's home directory (for example, `/home/hpsimuser`), Cygwin sets up mount points.

In the registry, navigate to

```
HKEY_LOCAL_MACHINE\SOFTWARE\Cygnus Solutions\Cygwin\mounts v2
```

Under this registry key the following three mount points are defined:

```
/

/home

/usr/bin
```

The native key under each of these is set to the corresponding Windows directory. Therefore, to determine where `/home` maps to, look up

```
HKEY_LOCAL_MACHINE\SOFTWARE\Cygnus Solutions\Cygwin\mounts
v2\/home/native
```

This mount point defaults to `C:\Documents and Settings`.

Similarly, the root directory (/) defaults to `C:\Program Files\OpenSSH`. So, `/etc/passwd` is found in `C:\Program Files\OpenSSH\etc\passwd`.

Passwd and group

For each user who is allowed to use SSH, there must be an entry in the password file. If a user who is not listed in the password file tries to log in, the connection fails with an illegal user error.

When the HP Systems Insight Manager OpenSSH package is installed, it sets up password entries for whomever is running the install, as well as Administrator. Administrator is set up because all of the pre-installed Windows command line tools run as Administrator. The `/etc/group` file is also created at install time, but this file should not need updating to add subsequent users.

In order to authorize other users, a password file entry needs to be made for them (the entry actually contains a SID—the password remains internal to Windows.) The entry is created with the **mkpasswd** command. Say you want to verify that hpsimuser is an allowed SSH user. First, check for an entry for hpsimuser in `/etc/passwd` (`C:\Program Files\OpenSSH\etc\passwd`.) Once the new user's absence is determined, add the user:

1. Open a DOS window and navigating to `C:\Program Files\OpenSSH\etc.`
2. Execute the command, **`mkpasswd`** `–l –u hpsimuser >> passwd`**.**
3. Execute the command, **`mkpasswd`** `–d –u hpsimuser domain >> passwd`**.**

One of these commands might return an error. That is fine, as error output prints out on the screen and not be redirected to the file. If the **mkpasswd** command cannot be found, navigate to the bin directory of the OpenSSH installation (usually `C:\Program Files\OpenSSH\bin\mkpasswd`.)

Once the password entry has been created, the user should be able to log in. When troubleshooting a user, another thing to check is the capitalization of the home directory. OpenSSH is case-sensitive in this regard, so `/home/HPsimUser` is not the same as `/home/hpsimuser`. Check that the capitalization in the password file is the same as the directory to which it refers.

Here is a sample passwd entry (**Note**: this would occur on a single line with no new lines.) The second-to-last field specifies the home directory:

```
Administrator:unused_by_nt/2000/xp:500:513:U-PCDLONG2\Administrator,S-1-
5-21-3769691966-4004114397-3833753107-00:/home/Administrator:/bin/switch
```

Coexistence problems with other Cygwin installations

Only one Cygwin-based program can be installed on a system at any given time.

In order for Cygwin to function, there are certain registry settings that have to exist—namely, the mount points defined above. The installer checks for the Cygwin registry keys and refuses to install if they exist. The installation also fails if the full Cygwin distribution, or any other software that uses Cygwin (for example, the Python distribution in WinCVS), is installed. This is an unfortunate consequence of multiple Cygwin installations not being able to coexist.

There are other products that use Cygwin out there, and HP Systems Insight Manager's OpenSSH distribution is not compatible with them. This includes other freely available OpenSSH distributions. If you are already using another version of OpenSSH and do not want to install the HP Systems Insight Manager version, that is fine. Keep in mind, however, that the HP Systems Insight Manager version is the only version that restricts access to the Cygwin data structures.

If the user has already installed the generic distribution of OpenSSH for Windows and sets up the keys to work with the CMS, the security hole that existed before HP Systems Insight Manager was used will still exist. It will not affect any other managed systems or the CMS. The potential exists for

a non-administrator user on the managed node to interfere with any DTF tasks run on that node. However, this same problem existed on this system before HP Systems Insight Manager was in use.

If you are having trouble getting the HP Systems Insight Manager OpenSSH package to install, search your system for the Cygwin registry keys, as well as the file `cygwin1.dll`. The location of the file might give you some idea of what software is installed that is conflicting with OpenSSH.

Documents and settings directory

The user's profile directory, usually `C:\Documents and Settings\user`, must exist so that SSH has a place to put its files. SSH creates a directory in the user's profile directory in which to place its known hosts and authorized keys files (for example, `C:\Documents and Settings\user\.ssh\known_hosts`.)

This directory is created the first time the user logs in, if it does not already exist. Therefore, be sure to log in as the domain user on each managed system on which that domain user is going to execute DTF tasks. Otherwise, when mxagentconfig is run for that user against that managed system, it fails because the authorized keys file cannot be created.

Configuring Systems without an 'Administrator' account

The standard installation of HP SIM assumes a local account called 'Administrator' is available on Windows, and this account is used when running standard tools such as Tools ➔ Command Line Tools ➔ Windows ➔ del. There are some additional steps you must take if you do not have a local account with this name: you must choose and configure an account to use, and update the tools to use the correct account name.

1. Select a user account that is to be used to run tools on Windows systems, including managed systems and the CMS. This should be a user with administrative rights on these managed systems[1]; it can be the same account used to install HP Systems Insight Manager. The user can be a domain account or a local account with the same name on each system. If this user account is to be used to manage Linux or HP-UX systems, the account name must be no longer than eight characters.

   Take the following steps if the account you want to use is not the one you used to install HP Systems Insight Manager:

   a. Create the account in Windows if it does not already exist, then log in to Windows on the CMS using this account to ensure this user's home directory is created.

   b. Enable SSH access for this user by adding the user to the OpenSSH passwd file:

      - Navigate to `C:\Program Files\OpenSSH\etc`
      - If a local account is to be used, run
        `mkpasswd` `–l –u <username> >> passwd`
      - Or if you have chosen a domain account, run
        `mkpasswd` `–d –u <username> <domain> >>passwd`

   c. Add this user account to HP Systems Insight Manager with full configuration rights and authorizations on all systems, either using the GUI or the command below:
      `mxuser` `–a <domain>\<username> -p full –C Administrator`

2. Modify the Windows HP Systems Insight Manager tools to use the new user account:

   a. Navigate to the tools directory, for example, `C:\Program Files\HP\Systems Insight Manager\tools`

   b. Edit `mx-tools.xml` (for example, using notepad)

[1] The user must be an administrator if the OpenSSH server supplied by HP SIM is used. If another SSH server is used then this need not be an administrator, provided the chosen user has sufficient right to run the desired tools on the managed system.

c. Find each execute-as-user line in the XML file and change Administrator to the account specified in step 1

<execute-as-user>**Administrator**</execute-as-user>

Changes to:

<execute-as-user>***username***</execute-as-user>

d. Run:

```
mxtool -m -f toolname.xml -x force
```

e. Repeat these steps for the other XML tools that use the Administrator account:

`openssh-install.xml, proliant-msa-tools.xml, wbemsubscriptions.xml`

3. Configure each of your managed systems that is to run tools with this user account:
   a. Add the user to the passwd file on each managed system (the user is already be configured if SSH was installed using that user account.) The commands used are the same as those used on the CMS in step 1b above.
   b. Run **mxagentconfig** on the CMS to copy the authentication keys for this user to each managed system:

```
mxagentconfig -a -u <username> -p <password> -n <system>
```

4. On every Windows Server 2003 system that is to be managed, including the CMS, a local administrative user account must exist, a domain account is not sufficient.
   a. If such an account does not exist, create a new user account and add it to the Administrators group.
   b. Follow the instructions for configuring the SSH service on Windows 2003, using this new user in place of the Administrator account.

Conclusion

HP Systems Insight Manager uses the SSH-2 protocol to execute tasks on managed systems. This requires an SSH server to be running and accepting requests on each managed system on which tasks are to be executed.

Features of HP Systems Insight Manager that require SSH to be installed and configured include custom commands and command line tools (both MSA and SSA.) HP Systems Insight Manager provides an OpenSSH package to be installed on Windows-based managed systems, as well as a key management tool (**mxagentconfig**) for setting up a user with the public key of the DTF.

The information contained here gives you an idea of the topology of remote task execution in HP Systems Insight Manager—and also gives you an idea of where to start troubleshooting when there is a problem.

Appendix A: Troubleshooting

When you have a problem executing a task, one of the following might be the cause:

- The SSH server on the managed system on which you are trying the command is not available
- The user running the command is not authorized to log in through SSH to the managed system
- The user trying to run the command does not have the HP Systems Insight Manager authorizations to run this tool on that managed system

In general, make sure that SSH is available by trying to log in outside of HP Systems Insight Manager. Then, make sure the user is able to log in through SSH using password authentication, again using some method outside of HP Systems Insight Manager. And finally, check the user's authorizations in HP Systems Insight Manager, and make sure **mxagentconfig** has been run for that user against that managed system.

Most importantly, make sure the user trying to run the command is the correct user. Sometimes the tool is designed to be run by a particular user such as root or Administrator. Other tools are designed to be run by the user who is logged into the CMS.

Problem: An MxAuthenticationException is generated when a tool is run, either from the GUI or the command line interface.

Solution: Several things can cause authentication exceptions:

- The user might not have the privileges needed to run the tool
- The user might not be set up with the public key of the DTF

1. Make sure that the user you are trying to run as has privileges to run that tool on that system. Refer to the HP Systems Insight Manager online help to check and grant authorizations.

2. Make sure that the SSH server is accessible on the target system.

   From the CMS, attempt to connect to the target system using an SSH command line tool. There is no need to log in, but make sure that you can connect. Try to log in as the administrative user to a Windows system, and as 'root' to an HP-UX/Linux system.

   From an HP-UX/Linux CMS:

   ```
   ssh root@<HP-UX/Linux node>
   ```

   or

   ```
   ssh Administrator@<Windows node>
   ```

   From a Windows CMS:

   ```
   <OpenSSH directory>\bin\ssh root@<HP-UX/Linux node>
   <OpenSSH directory>\bin\ssh Administrator@<Windows node>
   ```

   If you are prompted to accept a host key or enter a password, then the SSH server is accessible.

3. Re-run **mxagentconfig** to make sure that the keys are transferred:

   ```
   mxagentconfig -a -n <node name or IP> -u <user> -p <password>
   ```

4. On the system you are attempting to run tools on, check the permissions of some directories.

   Check the permission on the home directory of the user you are trying to run the tool as.

   - The home directory should have permissions: drwxr-xr-x (755)
   - The .ssh directory within the home directory should have permissions: drwxr-xr-x (755)

- The `authorized_keys2` file in the `.ssh` directory should have permissions: -rw-r--r-- or -rwxr-xr-x (644 or 755)

a. To check these permissions:

On Windows:

```
<OpenSSH Install Directory>\bin\ls -ld <File or directory name>
```

On HP-UX/Linux:

```
ls -ld <File or directory name>
```

b. To change permissions:

On Windows:

```
<OpenSSH Install Directory>\bin\chmod <Permission number><File or directory name>
```

On HP-UX/Linux:

```
chmod <Permission number> <File or directory name>
```

(Permission number is the number above, for example, 644/755)

When the command is run, the Execute-as user is listed in the status. This is the user that you have to run **mxagentconfig** for.

5. If execution has worked in the past and now is failing, verify that SSH has been reinstalled on the target system. Reinstalling SSH causes the system to have a different host key. Therefore, SSH will not be able to verify that the target system is the one that it is trying to contact.

   If SSH has been reinstalled, then use **mxagentconfig** to modify the `known_hosts` file:

   ```
   mxagentconfig –r –n <nodename>
   ```

   NOTE: mxagentconfig is the only tool to remove the entry from the `known_hosts` file while the CMS is running. Manually editing the file while the CMS is running has no impact.

   Alternately, you can also remove the entire `known_hosts` file when the CMS is not running, which means that SSH will re-register the keys of every system next time it contacts them. This could be a security problem until each system has been contacted.

6. Remove the `.ssh` directory from the home directory of the user on the managed system. This removes any old keys or old permissions that could cause **mxagentconfig** to fail.

7. Run **mxagentconfig** again.

Problem: mxagentconfig fails when trying to authorize a user on a Windows system that did not install OpenSSH.

Solution: The user is probably not authorized to use SSH on that system.

1. If trying to run as a Domain User, that user MUST log into the system prior to running **mxagentconfig**. The user's Documents and Settings directory does not exist until the user logs in, and if the user's Documents and Settings directory does not exist, then **mxagentconfig** fails.

2. As an administrative user on the system, run both:

   c:\Program Files\OpenSSH\bin\mkpasswd -l -u <username> >>
   "c:\Program Files\OpenSSH\etc\passwd"

   and

   c:\Program Files\OpenSSH\bin\mkpasswd -d -u <username> <Domain name> >>
   "c:\Program Files\OpenSSH\etc\passwd"

   Note: One of these might exit with an error, depending on the user.
   This is acceptable and expected.

3. Re-run **mxagentconfig**. If **mxagentconfig** still fails, make sure SSH is running by following the steps outlined above.

4. Make sure that the username being sent to **mxagentconfig** does not include the domain.
   Use `myusername` instead of `mydomain\myusername`.

5. Remove the `.ssh` directory from the home directory of the user on the managed system. This removes any old keys or old permissions that could cause **mxagentconfig** to fail.

6. If none of these work, then manually copy over the key.
   Transfer the file `.dtfSshKey.pub` to the managed system. The file can be found in the sshtools configuration directory.
   Linux and HP-UX:
   `/etc/opt/mx/config/sshtools/.dtfSshKey.pub`
   Windows:
   `<HP SIM Install Directory>\config\sshtools\.dtfSshKey.pub`
   On Windows:
   "type <location of .pub file> >> <user's home directory>\.ssh\authorized_keys2"
   On Linux and HP-UX:
   "cat <location of .pub file> >> ~/.ssh/authorized_keys2

Problem: When executing a task, the message 'Unknown OS' is displayed.

Solution: The installation might not have been completed properly.

1. If you are trying to execute a task on a Windows system, make sure that it was rebooted after installation of SSH. A reboot is required to complete the installation.

2. Enable DMI, WBEM, or SNMP on the system so the type of operating system can be determined, then run data collection to update the HP Systems Insight Manager database.

3. Make sure that commands to determine the operating system are working.
   For Windows, type: **ver**
   For HP-UX and Linux, type: **uname**

Problem: mxexec is not working with Windows 'runas' command.

Solution: A user who does not have full configuration rights cannot run the command line interface tools. This is expected behavior.

Problem: Windows 2003 does not allow the Local System account to have the privileges it needs to run the SSH service.

Solution: Configure the service to run as a real administrative user.

1. Stop the OpenSSH Server service:
2. Go to **Start** Menu ➔ **Control Panel** ➔ **Administrative Tools** ➔ **Services** to bring up the services window. Find the service labeled **OpenSSH Server** and stop it.
3. Change the **Log On As** user:
   a. In the same window, right-click **OpenSSH Server** service and select **Properties**.
   b. Select the **Log On** tab.

c. Select the **This account** radio button, and enter `.\Administrator`. Enter Administrator's password and click **OK**.

4. Set file permissions:
   a. Bring up a file explorer window by right-clicking the **Start** menu button, and selecting **Explore**. Navigate to `C:\Program Files\OpenSSH\var\log`. Delete any files you find in that directory.
   b. Navigate to `C:\Program Files\OpenSSH\etc` and select the files `ssh_host_dsa_key`, `ssh_host_key`, and `ssh_host_rsa_key` by holding down **Ctrl** and left-clicking on them. Then right-click on one of the files, select **Properties**, **Security** tab. Click **Advanced**. Select the **Owner** tab. Click **Other Users or Groups** and change the owner to **Administrators**.
5. Set user privileges:

   Select **Start** Menu ➔ **Control Panel** ➔ **Administrative Tools** ➔ **Local Security Policy** to bring up the security policy window. Find the **Policies for Create a Token Object** and **Replace a Process Level Token**. Add **Administrator** to this group by double-clicking the appropriate privilege, Click **Add User or Group**, enter `Administrator` in the **Enter the Object Names to Select** box, and click **Check Names** to verify the entry. Then click **OK**.
6. Start the OpenSSH Server service:

   Go to **Start** Menu ➔ **Control Panel** ➔ **Administrative Tools** ➔ **Services** to bring up the services window. Find the service labeled **OpenSSH Server** and start it.

   At this point, the service **Log On As user** is set to Administrator, and Administrator has been granted **Create a Token Object and Replace a Process Level Token** privileges. Return to the **Services** window and start the service.
7. Reinstall Systems Insight Manager SSH keys:

   Now, OpenSSH is properly configured to work under Windows 2003. In order to get command line and custom tasks to work in HP Systems Insight Manager, you will have must re-run **mxagentconfig** for Administrator if HP Systems Insight Manager was installed by someone other than Administrator.

   To do this, open a command window and run:

   ```
   mxagentconfig –a –u Administrator –p <Administrator's password> -n
   <cms machine name>
   ```

   Alternately, run **mxagentconfig** from the command line with no parameters and enter the data into the GUI.

Problem: Standard Windows tools run on the CMS fail with authentication error.

Solution: The Administrator account might not be correctly configured on the CMS to run SSH tools. Run **mxagentconfig** to configure the Administrator:

```
mxagentconfig –a –u Administrator –p <Administrator's password> –n <cms
machine name>
```

Problem: mxagentconfig or command execution fails after reinstalling the openSSH server.

Solution: The `known_hosts` file contains the signature of the old SSH server, and does not allow connections to a server at the same address but with a different key. Edit the `known_host` file under `<install dir>/config/sshtools/known_hosts` to remove all the lines containing the target hostname and IP address. The new key is added automatically unless adding unkown hosts has been disabled – refer to the earlier section on Known hosts for full details.

Appendix B: Changing server properties

The vast majority of users do not need to change any of the default server properties. Please change these values only if absolutely necessary.

The HP Systems Insight Manager system daemons read server properties at startup time. In order to change one of these properties, it is necessary to stop the system daemons, set the property in `mx.properties`, and restart the daemons.

1. Stop the system daemons.

   On HP-UX and Linux, type `/opt/mx/bin/mxstop`

   On Windows:

   a. Select **Start → Control Panel → Administrative Tools → Services** to display the services window.

   b. Find the service that begins with **HP Systems Insight Manager** and stop it (double-click the service and click **Stop.**)

2. Edit the property.

   On HP-UX and Linux, edit the file

   ```
   /etc/opt/mx/config/mx.properties
   ```

   On Windows, edit the file

   ```
   C:\Program Files\HP Systems Insight Manager\config\mx.properties
   ```

   If the property you want to change does not exist in the property file, add it. Otherwise, edit the property with the desired value.

3. Restart the system daemons.

   On HP-UX and Linux, type `/opt/mx/bin/mxstart.`

   On Windows:

   a. Select **Start → Control Panel → Administrative Tools → Services** to display the services window.

   b. Find the service that begins with **HP Systems Insight Manager** and start it (double-click the service and click **Start**.)

   It might take some time for the daemons to initialize and the system to begin responding again.

Appendix C: Tool examples

This section provides examples of MSA and SSA tools available in HP Systems Insight Manager.

MSA tools

Category	Tool Name	Description
Command Line Tools	PostgreSQL DB Backup	Back up the Systems Insight Manager PostgreSQL database.
Configuration Tool	Subscribe to WBEM Events, Unsubscribe to WBEM Events	Configure a managed system to send WBEM indications to HP Systems Insight Manager.

SSA tools

Category	Tool Name	Description
Configure	Configure DMI Access	Set DMI access on selected nodes.
Configure	Configure SNMP Access	Set SNMP access on selected nodes.
General Tools	Install RPM	Install RPM Package Manager package(s.)
General Tools	Query RPM	Query installed RPM Package Manager package(s) version.
General Tools	Uninstall RPM	Uninstall RPM Package Manager package(s.)
General Tools	Verify RPM	Verify installed RPM Package Manager package(s.)
General Tools	bdf	Report free disk space on files or filesystems.
General Tools	cat	Display the contents of a file.
General Tools	copy	Copy one or more files to another location.
General Tools	cp	Copy file or files to a destination file or directory.
General Tools	del	Delete one or more files (or all files in specified directories.)
General Tools	df	Report free disk space on files or filesystems.
General Tools	dir	Display list of files and subdirectories in a directory.
General Tools	find	Recursively descend a directory hierarchy.
General Tools	ls	List files or directories.
General Tools	mv	Move file or files to a destination.
General Tools	net	Display Windows System and Network information.
General Tools	netstat	Display active network connections.
General Tools	ps	List system processes.
General Tools	rm	Remove files or directory trees.
General Tools	rmdir	Remove a directory and all its contents.
General Tools	type	Display the contents of one or more text files.
Partition Management	Create Partition	Start the Create Partition dialog on the selected node in the complex.
Partition Management	Partition Manager	Start the Partition Manager graphical user interface on the selected node in the complex.
Partition Management	Show Complex Details	Start the Show Complex Details dialog on the selected node in the complex.
Partition Management	View Partition Manager Log	Start the Log Viewer dialog on the selected node in the complex.
Resource Management	Display Resource Usage	Display the current Process Resource Manager resource usage.
Resource Management	Event Monitoring Service	Configure and view resource monitoring requests on the managed node.

Category	Tool Name	Description
Resource Management	List Resource Availability	List Process Resource Manager resources available.
Resource Management	Process Resource Manager Console	Run the Process Resource Manager for managing system resources.
Software Management	CLI List Software	Example tool that runs Software Distributor (SD) swlist command on each node.
Software Management	CLI Preview Install	Example tool that runs Software Distributor (SD) swinstall -x match_target=true command on each node.
Software Management	CLI Verify Software	Example tool that runs swverify command on each node.
Software Management	Set SD Access	Set Software Distributor (SD) access to the target node via the appropriate SD access control lists (ACLs.)
Software Management	Software Distributor Daemon Log	Display the tail end of the Software Distributor (SD) daemon log.
Software Management	View Depot Software	Start the Software Distributor (SD) graphical user interface to view depot software and depot logfile.
Software Management	View Installed Software	Start the Software Distributor (SD) graphical user interface to view installed software and agent logfile.
Software Management	View Software Distributor Agent Log	Display the tail end of the Software Distributor (SD) agent log.
System Administration	Accounts for Users and Groups	Start the HP-UX SAM Accounts for Users and Groups functional area.
System Administration	Auditing	Start the HP-UX SAM Auditing functional area.
System Administration	Disks and File Systems	Start the HP-UX SAM Disks and File Systems functional area.
System Administration	Kernel Configuration	Start the HP-UX SAM Kernel Configuration functional area.
System Administration	Peripheral Devices	Start the HP-UX SAM Peripheral Devices functional area.
System Administration	Printers and Plotters	Start the HP-UX SAM Printers and Plotters functional area.
System Administration	System Properties	Start the HP-UX SAM System Properties functional area.
System Administration	System Security Policies	Start the HP-UX SAM System Security Policies functional area.
System Administration	Verified Commands	Start the HP-UX SAM Verified Commands functional area.
System Administration	View SAM Log	Start the HP-UX SAM Log Viewer X application.

Appendix D: Glossary

API— application programming interface. An interface provided for programs to execute services provided by a piece of software, vs. a human executing those services via the command line or a GUI.

CMS— central management server. The system on which HP Systems Insight Manager is installed.

Cygwin— a UNIX compatibility layer that is used to port some UNIX utilities to Windows.

DSA— digital signature algorithm. A public key algorithm used by SSH.

GUI— graphical user interface. For example, the Web-based portal interface to HP Systems Insight Manager.

Host key— the public key that proves the identity of a particular host.

IETF— Internet Engineering Task Force. From the IETF Web page: "The Internet Engineering Task Force (IETF) is a large open international community of network designers, operators, vendors, and researchers concerned with the evolution of the Internet architecture and the smooth operation of the Internet."

Managed system— any system on the network being managed by HP Systems Insight Manager, including the CMS itself.

Mount point— maps a physical file system name to a logical name, which can then be used for convenience.

MSA tool— multi-system aware tool. This is a tool that executed on a certain system called the execution node, and then performs tasks against the target systems. Target systems are provided to the tool by an environment variable.

OpenBSD— a free, Berkeley Software Division (BSD) 4.4–based UNIX-like operating system. Their implementation of the SSH protocol is OpenSSH.

OpenSSH— a free version of the SSH protocol suite, implemented and supported by the OpenBSD project.

Private key— the private half of a public/private key pair. The private key is stored in and owner read-only file (for example, only the owner can view it) on a particular system. The private key is never transmitted to another system.

Public key— the public half of a public/private key pair. The public key can be freely distributed without fear that it can be used to impersonate the user. It can only be used for authentication in conjunction with a private key.

Remote task— a task initiated on the CMS, and executed on a managed system.

RSA— Rivest-Shamir-Adleman. A public key algorithm used by SSH.

SFTP— Secure File Transfer Protocol. It is the part of the SSH protocol used to transfer files between systems. This protocol is performed with the same server as command execution.

SSA Tool—single-system aware tool. This type of tool is executed via SSH on the target system.

SSH— Secure Shell. An IETF recommendation. There are two protocols: the original SSH version 1 protocol (SSH-1) and the current SSH version 2 (SSH-2.) Whenever SSH is mentioned in this document, it refers to the SSH-2 protocol.

SSH client— connects to SSH servers to perform remote task execution and file copy.

SSH server— listens for and services requests coming in on the proper TCP/IP port, usually port 22.

Target system— the system selected for a tool to run on.

TDEF— tool definition file. It defines parameters of a tool, its execution user, tool box, etc. in XML format.

For more information

HP Systems Insight Manager

www.hp.com/go/hpsim

IETF secsh working group home page

www.ietf.org/html.charters/secsh-charter.html

OpenSSH

www.openssh.org

5982-4832EN, 11/2004