close window

IBM Systems magazine

e-Newsletter Exclusive

Print

On-Demand Systems Management

SSH simplifies AIX administrative tasks

April 2011 | by Jaqui Lynch

In today's on-demand world, customers have to spin up LPARs and deploy them more frequently and faster than before. The capability to rapidly make changes or to provision important files and run scripts on multiple LPARs has become a critical component in managing systems. However, for many, having to enter a password to do this on every system touched is a major annoyance. The ideal solution is to be able to script repeatable processes so no manual intervention is required. There are ways to do this using SSH that make this kind of deployment far more simple.

What is SSH?

Secure shell (SSH) was written to provide more secure alternatives to insecure protocols such as rlogin, rcp, FTP and telnet. It provides authentication, encryption and data integrity across the Internet. The version that currently ships with AIX v7 and v6 is openssh v5.4p1, and it also uses openssl 0.9.8. You can check if these are installed as follows:

```
lslpp -l | grep openss
openssh.base.client     5.4.0.6100   COMMITTED   Open Secure Shell Commands
openssh.base.server     5.4.0.6100   COMMITTED   Open Secure Shell Server
openssh.man.en_US       5.4.0.6100   COMMITTED   Open Secure Shell
openssh.msg.EN_US       5.4.0.6100   COMMITTED   Open Secure Shell Messages -
openssh.msg.en_US       5.4.0.6100   COMMITTED   Open Secure Shell Messages -
openssh.base.client     5.4.0.6100   COMMITTED   Open Secure Shell Commands
openssh.base.server     5.4.0.6100   COMMITTED   Open Secure Shell Server
openssl.base      0.9.8.1300   COMMITTED   Open Secure Socket Layer
openssl.license      0.9.8.1300   COMMITTED   Open Secure Socket License
openssl.man.en_US      0.9.8.1300   COMMITTED   Open Secure Socket Layer
openssl.base      0.9.8.1300   COMMITTED   Open Secure Socket Layer
```

You can also check the version of ssh with:

```
ssh -V
OpenSSH_5.4p1, OpenSSL 0.9.8m 25 Feb 2010
```

The most recent version of SSH available is at AIX v5.8, however it's source code that you'll need to compile yourself. So for the purposes of this article, I'll use the SSH v5.4 that comes with AIX.

SSH encrypts all of the authentication traffic, helping ensure user names and passwords don't travel in the clear. SSH provides several authentication options, including the use of UNIX passwords or the use of a public/private key pair for authentication. Additionally, SSH interfaces with TCP wrappers for logging and access control and has its own built-in access control that's fairly granular. The suite includes several features, including the capability to do secure backups, tunneling and X11 forwarding, all within a secure environment. SSH also comes with a secure FTP server: SFTP.

Sending Commands Without Passwords

The first thing to do is ensure the systems are secure in all other ways, as you are about to tell these systems to trust each other. Once the systems are generally secure then you can move on to setting up your ssh keys. In the examples below, the actions were performed as root. However it's also possible to use ssh to run non-root scripts.

On the server you plan to use as your control server (in this case I'm using my network installation management, or NIM server) you first need to set up the keys as follows:

```
#ssh-keygen -t dsa
```

You'll then see the following lines. Just hit enter whenever prompted:

```
Generating public/private dsa key pair.
Enter file in which to save the key (//.ssh/id_dsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in //.ssh/id_dsa.
Your public key has been saved in //.ssh/id_dsa.pub.
The key fingerprint is:
LINE OF STUFF HERE
The key's randomart image is:
SEVERAL LINES OF STUFF HERE
+-----------------+
```

You should now see the following files:

```
#ls /.ssh
id_dsa     id_dsa.pub
```

Use cat to grab the contents of /.ssh/id_dsa.pub and save them to notepad on your desktop. ssh to the client and when prompted to save the host key, say yes. This ensures the host key is saved into /.ssh/known_hosts. Now on the client:

```
cd
mkdir .ssh
chmod 700 .ssh
cd .ssh
vi authorized_keys2
Store the contents of id_dsa.pub here
```

Another option is to scp the file over as follows, but this assumes /.ssh is already set up:

```
scp /.ssh/id_dsa.pub clientsrvr:.ssh/authorized_keys2
```

From the NIM server you should now be able to ssh commands to the client as follows:

```
ssh clientsrvr df
```

It's possible you may have to run the ssh-add command to get the keys loaded. In my case, I didn't have to, but you can find more information on this online.

Going forward, you can use ssh to send commands to the client server without having to type in a password; you can run scripts to perform management functions. Here's a quick script that will allow you to test whether it's working to a list of LPARs (scripts provided by Andrew Goade):

```
# testssh.sh
echo "Running test Script"
for host in $(cat /usr/local/etc/cmdhosts.txt)
do
    echo "$host df"
    ssh root@$host df
done
```

The file in /usr/local/etc/cmdhosts.txt simply contains a list of the network names or IPs of the LPARs you want to send the commands to.

As another example, you may want to have a script to shut down all of the LPARs in a list. It would look something like this (again cmdhosts.txt would be the list of hosts/LPARs):

```
# shutdown_lpars.sh
echo "Running test Script"
for host in $(cat /usr/local/etc/cmdhosts.txt)
do
    echo "$host shutting down now"
    ssh root@$host shutdown -h now
done
```

As you can see, this can be tailored to perform a number of administrative tasks on multiple systems, which can save you a significant amount of time. I recently set up more htan 70 LPARs and did a significant amount of the work using ssh with scripts, including performing functions like adding multipath I/O (MPIO) to Linux after it was already running.

Another function administrators perform regularly is backing up or restoring a filesystem. Usually I set these up to run automatically as cron jobs, but occasionally I have to start one up automatically on networks where NFS isn't allowed. Using SSH tunnels allows me to avoid using NFS. (Below is the command to backup /freddy to a remote system called nimit.)

```
From the client, backing up to a server called nimit:
    tar -cvzf - /freddy | ssh root@nimit "cat > /backups/freddy.tar.gz"
Back the directory up to a remote tape drive on nimit:
    tar -cvzf - /freddy | ssh root@nimit "cat > /dev/rmt0"
Restore the tar backup using SSH:
    cd /
ssh root@nimit "cat /backups/freddy.tar.gz" | tar -zxvf -
```

In all the above examples, absolute paths were used, so the backups will include those.

Other useful things you can do with SSH include:

- X11 forwarding that allows the encryption of network X windows traffic so that the data and command streams can't be modified in-flight.
- Port forwarding allows the forwarding of TCP/IP connections to a remote system over an encrypted channel.
- Backup using tar via an SSH tunnel.
- Add SSH to the rdist/rsync configs and tunnel them.

Comments

Having a good understanding of SSH and how to use it to run scripts and commands remotely is very beneficial in today's complex environments. The use of scripts and lists of hosts allows you to group your LPARs/hosts together and issue commands to them all at the same time. It also allows you to write scripts to ensure a whole environment is shutdown in the correct order. It's also possible to write scripts that will connect using SSH to the Hardware Management Console and bring the LPARs back up in the correct order as well. The use of scripting for this, combined with technologies like FLRT and NIM makes managing servers much easier and can save a considerable amount of time.

And of course, depending on your AIX and SSH versions the syntax may vary so please test these commands first on non-critical servers.

References

Article on setting up SSH keys
http://pkeck.myweb.uga.edu/ssh/

OpenSSH Home Page
http://www.openssh.org/

OpenSSH man pages
http://www.openssh.org/manual.html

Article on SSH
http://www.ibmsystemsmag.com/aix/administrator/security/Stronger-Security-With-OpenSSH/