IBM Ported Tools for z/OS: OpenSSH - Key Authentication

June 12,2012

Kirk Wolf
Steve Goetze

DovetailedTechnologies

http://dovetail.com
info@dovetail.com

Dovetailed Technologies

We provide z/OS customers world wide with innovative solutions that enhance and transform traditional mainframe workloads:

- Co:Z Co-Processing Toolkit for z/OS
- T:Z Quickstart for Tomcat and z/OS
- JZOS - acquired by IBM in 2005 and now part of the z/OS Java SDK

Co:Z Components

- Co:Z SFTP **
  - OpenSSH SFTP with z/OS exploitation
- Co:Z Batch
  - full featured BPXBATCH replacement
- Co:Z Dataset Pipes
  - convert datasets to streams / streams to datasets
  - other z/OS Unix commands / utilities
- Co:Z Launcher **
  - z/OS hybrid batch processing (distributed apps+data)
- Co:Z Target System Toolkit **
  - used with Co:Z Launcher and Dataset Pipes

** Requires IBM Ported Tools OpenSSH

Co:Z® UP TO A FRIENDLIER SOFTWARE PRICING MODEL:

FREE

Use the Co:Z Co-Processing Toolkit and our free support forum at no cost. Annual site support agreements start at just $4995.

DovetailedTechnologies
www.dovetail.com
+1.636.300.0901

Co:Z SFTP
OpenSSH secure file transfer with support for MVS datasets and SMF logging

Co:Z Launcher
Execute Unix or Windows processes from a batch job with access to MVS datasets

Co:Z Dataset Pipes
Flexible conversion of MVS datasets to/from pipes or Unix files

Co:Z FTP-SSH Proxy
FTP tunnelling over SSH

Co:Z Batch
A better BPXBATCH

Co:Z Co-Processing Toolkit

for z/OS | Dataset Pipes : SFTP : FTP-SSH Proxy : JZOS : DtlSpawn

Agenda

- IBM Ported Tools for z/OS - OpenSSH
  - Features
  - Co:Z Toolkit dependencies
- SSH authentication: two kinds
- Using host (server) keys in OpenSSH
  - How to distribute host public keys
- SSH User keys
  - an alternative to user passwords
- Next webinar:
  "IBM Ported Tools for z/OS: OpenSSH – Using Key Rings"

What is “SSH”?

- The IETF SSH-2 standard protocol (RFC 4251 etc)
- Features:
  - A secure (encrypted) connection over **one** TCP/IP socket between a client and a server
  - *Authentication of the user and host.*
  - (optional) LZ compression
  - Support for one or more simultaneous ***application channels*** over the same connection: terminal, sftp, command, port fwd, ...
- There are many compatible implementations,
  - **OpenSSH** is by far the most popular; it is a default package on all Unix/Linux distributions
  - PuTTY is a popular free Windows client

IBM Ported Tools for z/OS - OpenSSH

- A port of OpenSSH for z/OS
  - z/OS Unix commands: **ssh, sshd, sftp, sftp-server**, etc.
- No support for MVS datasets, spool files, etc.
- Release 1.2 added support for:
  - SSH keys in SAF/RACF keyrings
  - SMF logging (new SMF 119 record subtypes)
- PTF UA63842 added:
  - ICSF hardware acceleration for ciphers and MACs
- Co:Z Toolkit requires IBM Ported Tools OpenSSH:
  - Co:Z SFTP client invokes **ssh**
  - Co:Z SFTP server is invoked by **sshd**
  - Co:Z Launcher invokes **ssh**
  - Co:Z Dataset Pipes *can* be used remotely via **sshd**

SSH Authentication

```
ssh myuser@myserver.com
```

Each connection is authenticated using **both**:

- Host (server) authentication
  - The server (myserver.com) proves its identity using a public / private “host” key pair
- User authentication
  - The client authenticates its use of “myuser” on the server using one of the following:
    - password
    - public / private “user” key pair

  or on many non-z/OS platforms:
    - GSS-API (Kerberos), Smart cards / tokens, PAM

Public / private keys

- A pair of large numbers with a unique relationship:
  - Data encrypted by a private key can only be decrypted using the corresponding public key
- Alice encrypts (“signs”) some agreed upon data using the private key and sends the encrypted data to Bob
- Bob decrypts the data using the public key and checks that it matches the known data (“verifies the signature”)

→ If successful, then Bob knows that Alice has the private key

→ SSH supports two asymmetric key algorithms: RSA and DSA

SSH Host (Server) Authentication

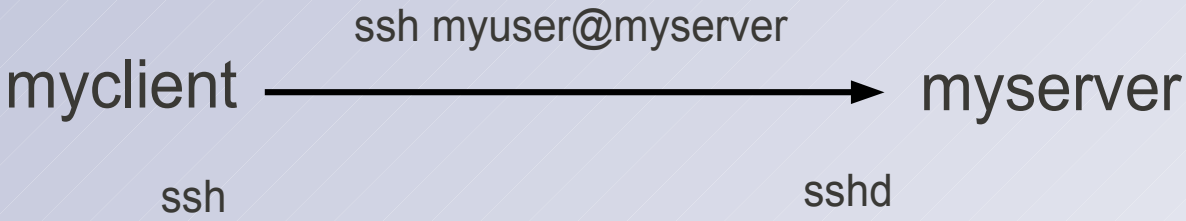
- Each SSHD server has a public/private “host” key pair.
  (Usually two pairs: one for each algorithm: rsa & dsa)
- Clients have file(s) that associate server hostnames or ip adresses with a verified public key
  - ~/.ssh/known_hosts -or-
  - /etc/ssh/ssh_known_hosts
- If a client connects to a host without a verified copy of the public key, the user must “accept” the key
  - `ssh option “-o StrictHostKeyChecking=no”`
    will automatically accept the key for a new host (*use with care*)
- If a client connects to a host and the host key doesn't match the “known” public key, the connection fails.

Creating host (server) key files

```
ssh-keygen -t rsa  { -b bits }
           -f /etc/ssh/ssh_host_rsa_key
           -N ''
```

- `-t` (type) is rsa or dsa.
  Generally, you would define both types to support clients who only have one or the other. With rsa, you can define the key length in bits; the default is 2048
- `-f` gives the name of the private key file.
  Associated public key file is: `ssh_host_rsa_key`**`.pub`**
  Only "root" should be able to read the private key file.
- `-N ''` indicates a null passphrase.
  (passphrases may not be used for host private keys)

Server authentication using known_hosts




```
# file: /etc/ssh/ssh_known_hosts
#   or ~/.ssh/known_hosts
myserver ssh-rsa  AA123....=
```

/etc/ssh/ssh_host_rsa_key **
/etc/ssh/ssh_host_rsa_key.pub

Who is Alice? Who is Bob?

Note: the single line containing the public key is copied from ssh_host_rsa_key.pub and added to the client's known_host file

** Consider instead putting private key in SAF keyring
(next webinar: “IBM Ported Tools OpenSSH – Using Key Rings”)

Demo: Host (server) key authentication

SSH Host (public) Key Distribution

- Can you depend on users to verify server host public keys before accepting?
- Administrators can pre-build `/etc/ssh/ssh-known-hosts`
  - You can use a common “master” file with the organization's host (server) public keys
  - The **ssh-keyscan** command can be used to gather
  - Consider automatically publishing to your client machines
  - For non-OpenSSH client machines (e.g. Windows/PuTTY), you may need to convert the format of this file.
- If your domain's DNS supports SSHFP records, place each host's public keys in the DNS
  - Requires DNS-SEC in order to be secure
  - Not all DNS clients or SSH client products *currently* support this
  - This is hopefully the future

User (client) authentication with keys

- Client has the private key
  - in a file, by default: `~/.ssh/id_rsa` or `id_dsa`
- Server has the matching public key associated with the userid
  - in: `~/.ssh/authorized_keys`
    - Private keys stored in files must be secured so that only the owner can read. Even then, there is a risk that this user will distribute a copy.
    - authorized_keys file must be secured so that only the owner can create or modify.

Creating User keys

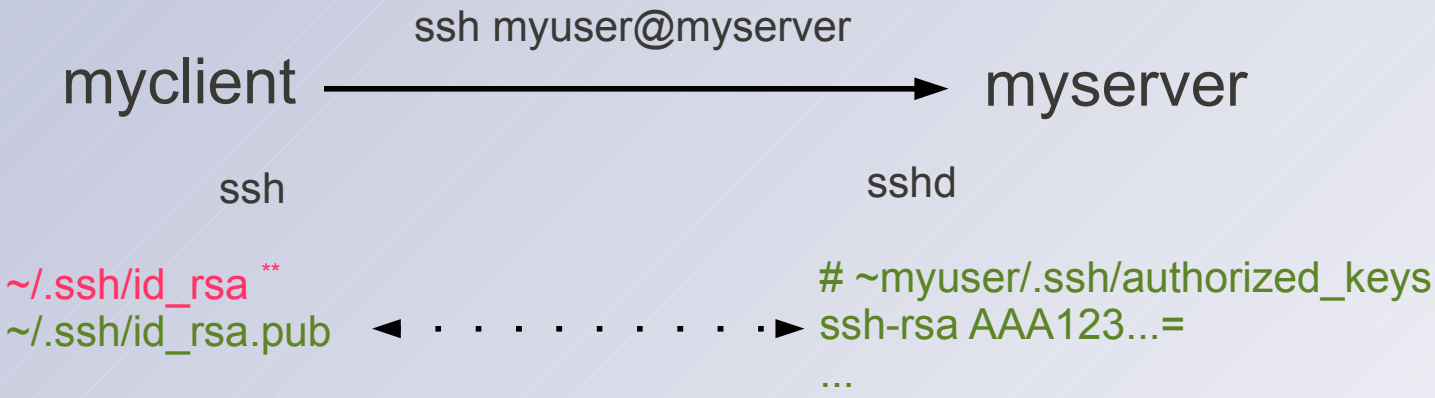
```
ssh-keygen -t rsa  { -b bits }
           { -f ~/.ssh/id_rsa }
           { -N 'pass phrase' }
```

- `-t` (type) is rsa or dsa
  RSA is more common, and supported by crypto co-processor.
  With RSA, you can specify the number of bits; default is 2048.

- `-f` names the private key file; the public key file will add a **.pub** suffix. If not supplied, then user will be prompted with a default of `~/.ssh/id_rsa`

- `-N` provides a pass phrase used to encrypt the private key
  If given, the user will be prompted when using the key

User authentication using OpenSSH key files



Who is Alice? Who is Bob?

** Consider instead putting private key in SAF keyring
(next webinar: *"Using SAF(RACF) keyrings with z/OS OpenSSH"*)

Demo: User key authentication

SSH Authentication Review

- Host (server) authentication
  - Server (SSHD) has host private key(s)
  - Clients have matching host public key
    - known_hosts is a list of: `<host -> public key>`
- User key authentication
  - User has private key
  - Server has matching public key
    - $HOME/.ssh/authorized_keys is a list of the user's public keys

Common Pitfalls

- z/OS client or server userid must have an OMVS segment.
- Avoid sharing UNIX uids between z/OS userids
- key files are text, and EBCDIC on z/OS.
- Must use proper file permissions (or OpenSSH may ignore):

```
~  - 7xx (user home - not group or world writable)
  ~/.ssh    -  700
   id_dsa, id_rsa (private keys) - 600
   authorized_keys - 600
   known_hosts  - 600

/etc    -  755
/etc/ssh  - 755
    ssh_host_rsa_key  - 600
    ssh_host_rsa_key.pub   - 644
```

More information

- IBM Ported Tools for z/OS: OpenSSH User's Guide
- Co:Z SFTP User's Guide
- http://dovetail.com/forum (public bulletin board)
- Our webinar archives: http://dovetail.com/webinars
- Next webinar (part 2):

  "IBM Ported Tools OpenSSH: Using Key Rings"
  June 19, 2012 2PM EDT
  To enroll: https://www3.gotomeeting.com/register/275261614