IBM z Systems

OpenSSH for z/OS

IBM Ported Tools for z/OS V1.3

Kershaw Mehta
kershaw@us.ibm.com

IBM, Chief Architect for Cloud Computing on z Systems

What is SSH?

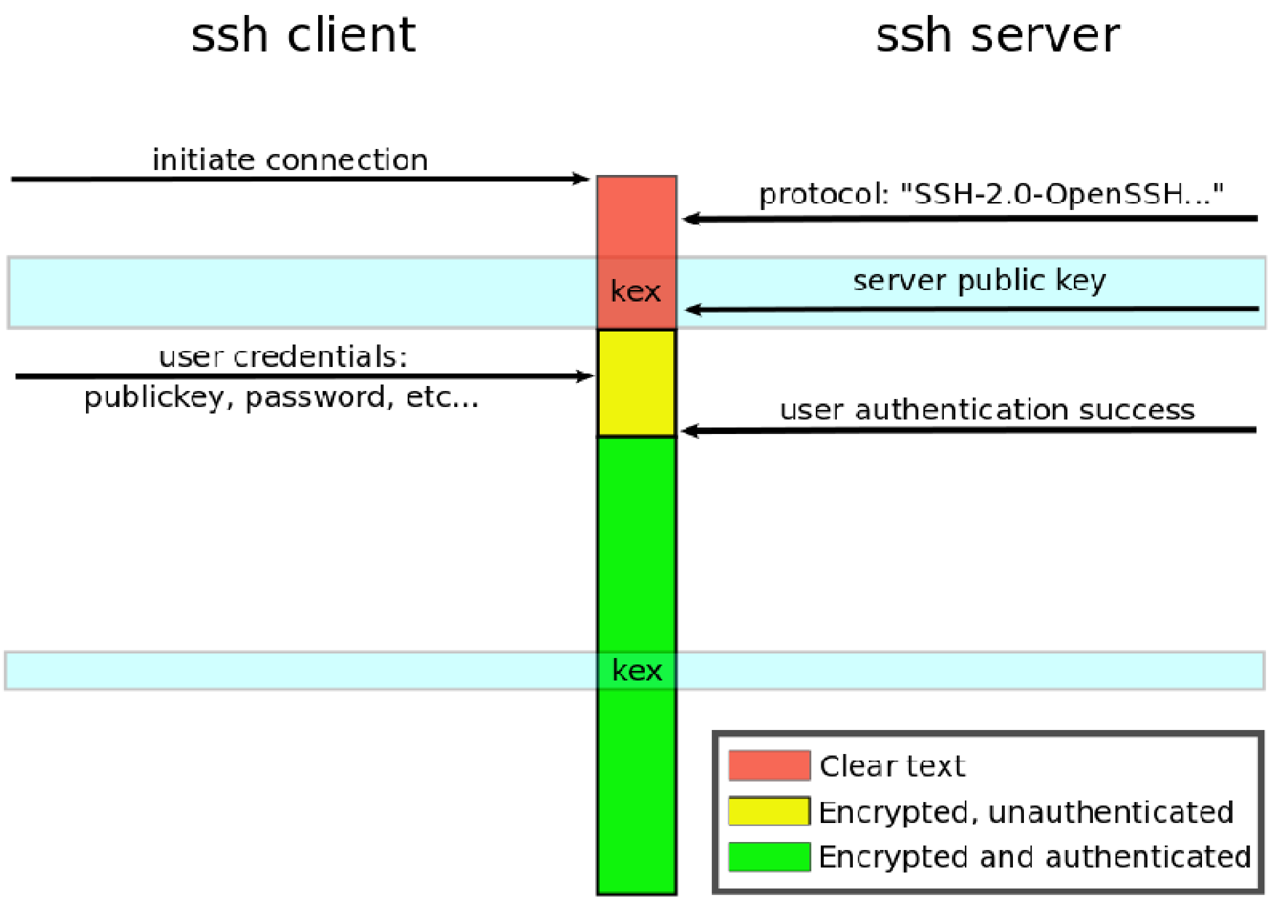
- The IETF SSH-2 standard protocol (RFC 4251 etc)
- Features:
  - A secure (encrypted) connection over **one** TCP/IP socket between a client and a server
  - Authentication of both user and host
  - (optional) LZ compression
  - Support for one or more simultaneous ***application channels*** over the same connection: terminal, sftp, command, port forwarding, ...
- 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

SSH2 Crypto at-a-glance

- **Key Exchange – "kex"**
  - Some variant of Diffie-Hellman
  - Client generated random number and the server key are used to:
    - Allow the client to authenticate the identity of the server
    - Independently generate and exchange a secret session key
    - Enable session rekeying. Typically once/hour or GB.
- **User Authentication**

  At start of session, a password or **user** key pair can be used to authenticate the user to the server.
- **Session Encryption**
  - A symmetric Cipher uses the shared session key to encrypt the packet payload.
  - A MAC algorithm (typically SHA-1) is used to generate a hash of each packet.

SSH Encryption and Authentication


ssh client
ssh server
initiate connection
protocol: "SSH-2.0-OpenSSH..."
kex
server public key
user credentials:
publickey, password, etc...
user authentication success
kex
🟥 Clear text
🟨 Encrypted, unauthenticated
🟩 Encrypted and authenticated

IBM Ported Tools for z/OS V1.3 – OpenSSH Overview

- Base upgraded from OpenSSH 5.0p1 to 6.4p1
- ICSF acceleration of CTR mode AES ciphers
  - CTR mode is now preferred over CBC*
- New SMF logging detail
- Enabled ssh client to be invoked under TSO/OMVS shell
  - entry of password credentials not permitted
- Relaxed syntax of IdentityKeyRingLabel
  - double quotes optional when entered from ssh, sftp, or scp command line

*http://www.kb.cert.org/vuls/id/958563

OpenSSH Product Notes

- New release FMID HOS1130 installs over the previous release FMID HOS1120
- HOS1130 is supported on z/OS 1.13 and later
- HOS1120 supported through z/OS V2R1, but withdrawn from marketing (ie. no longer available to be ordered) when we made HOS1130 available.
- /dev/random is now **required** – HOS1130 will not run without ICSF active!
- Verifying version

```
$ ssh -V
OpenSSH_6.4p1, OpenSSL 1.0.1c 10 May 2012

$ /usr/sbin/sshd -d -t
…
debug1: sshd version OpenSSH_6.4p1, OpenSSL 1.0.1c 10 May 2012
```

IBM's Direction

- OpenSSH is planned to be provided as a base element in z/OS V2.2
- Same level that is in IBM Ported Tools V1.3…
- But we do plan on providing future functions that are exclusive to the OpenSSH element in z/OS V2.2 (HOS2220):
- *"IBM plans to add OpenSSH to z/OS and enhance it by providing Kerberos support, which is designed to enable single sign-on from Microsoft™ Windows™ domains, and also to leverage the capabilities of IBM zEnterprise Data Compression (zEDC). These capabilities are also planned to be made available in the version of OpenSSH that is part of IBM Ported Tools for z/OS."*
  - IBM Software Announcement: ZP15-0006, January 14, 2015

OpenSSH 6.4p1 Feature Highlights (since 5.0p1)

- Key Exchange algorithms can now be specified (-oKexAlgorithms)
  - diffie-hellman-group1-sha1
  - diffie-hellman-group14-sha1
  - diffie-hellman-group-exchange-sha1
  - diffie-hellman-group-exchange-sha256
  - **ecdh-sha2-nistp256**
  - **ecdh-sha2-nistp384**
  - **ecdh-sha2-nistp521**

- NIST Elliptic-curve algorithms added

Note: new algorithms **highlighted**

OpenSSH 6.4p1 Feature Highlights (since 5.0p1)

- Key Algorithms – used for ssh host (server) or user keys
  - ssh-rsa,ssh-dss
  - **ecdsa-sha2-nistp256**
  - **ecdsa-sha2-nistp384**
  - **ecdsa-sha2-nistp521**
  - **ssh-rsa-cert-v01@openssh.com**
  - **ssh-dss-cert-v01@openssh.com**
  - **ecdsa-sha2-nistp256-cert-v01@openssh.com**
  - **ecdsa-sha2-nistp384-cert-v01@openssh.com**
  - **ecdsa-sha2-nistp521-cert-v01@openssh.com**
  - **ssh-rsa-cert-v00@openssh.com**
  - **ssh-dss-cert-v00@openssh.com**

Note: 1. New algorithms **highlighted**
2. non-standard non-RFC names have "@openssh.com"

- NIST Elliptic-curve DSA w/ SHA-2 algorithms added
- OpenSSH "certificates" added (more later)

OpenSSH 6.4p1 Feature Highlights (since 5.0p1)

- Cipher algorithms – default preference order shown
  - **aes128-ctr***
  - **aes192-ctr***
  - **aes256-ctr***
  - Arcfour256
  - arcfour128,
  - **aes128-gcm@openssh.com**
  - **aes256-gcm@openssh.com**
  - aes128-cbc*
  - 3des-cbc*
  - blowfish-cbc
  - cast128-cbc
  - aes192-cbc*
  - aes256-cbc*
  - arcfour,rijndael-cbc@lysator.liu.se*

Note: **1.** New algorithms **highlighted**
2. non-standard non-RFC names have "@openssh.com"
3. ICSF support noted with "*"

OpenSSH 6.4p1 Feature Highlights (since 5.0p1)

- Cipher algorithms
  - AES GCM (Galois/Counter Mode) ciphers added to OpenSSH
    - Function as both Cipher and HMAC in one
  - AES CTR mode ICSF support has been added to HOS1130
    - Accelerates the most widely used OpenSSH Ciphers

OpenSSH 6.4p1 Feature Highlights (since 5.0p1)

- MAC algorithms – default preference order shown
  - **hmac-md5-etm@openssh.com***
  - **hmac-sha1-etm@openssh.com***
  - **umac-64-etm@openssh.com**
  - **umac-128-etm@openssh.com**
  - **hmac-sha2-256-etm@openssh.com***
  - **hmac-sha2-512-etm@openssh.com***
  - hmac-ripemd160-etm@openssh.com*
  - **hmac-sha1-96-etm@openssh.com***
  - **hmac-md5-96-etm@openssh.com***
  - hmac-md5*
  - hmac-sha1*
  - **umac-64@openssh.com**
  - **umac-128@openssh.com**
  - **hmac-sha2-256***
  - **hmac-sha2-512***
  - hmac-ripemd160*
  - hmac-ripemd160@openssh.com*
  - hmac-sha1-96*
  - hmac-md5-96*

Note 1. New algorithms **highlighted**
2. non-standard non-RFC names have "@openssh.com"
3. ICSF support noted with "*"

OpenSSH 6.4p1 Feature Highlights (since 5.0p1)

- MAC algorithms
  - SHA-2 algorithm added (with ICSF support)
  - UMAC algorithm support added
  - “-etm@openssh.com” algorithms are **not** new algorithms!
    - They are variants that indicate that the MAC is calculated **after** encryption (“Encrypt-then-MAC”) rather than the other way around. The community now considers this more secure (in theory).

OpenSSH 6.4p1 Feature Highlights (since 5.0p1)

- Dynamic port assignment for remote port forwarding
  - ssh -R 0:host:port
  - A remote port of “0” can be specified in which case a dynamic port will be assigned on the server.
  - The client will report a message with the specific ephemeral port assigned.
- More flexibility in configuration files
  - Match blocks have more criteria and can include more options within the block.

OpenSSH 6.4p1 Feature Highlights (since 5.0p1)

- Support for public key (user and host) certificates
  - These are not X.509 certificates, but a simpler implementation that is unique to OpenSSH.
    - A single key (“CA key”) may sign (vouch for) the public keys of many users or servers. If a host or user trusts the CA public key, then it implicitly accepts the keys that have been signed by it.
  - For more information, see the User's Guide / man page for the ssh-keygen command.
  - These have been available for a few years, but are not widely used.
  - No z/OS Key Ring support for these or their associated keys.

OpenSSH 6.4p1 Feature Highlights (since 5.0p1)

- Multi factor authentication methods
  - The server may specify that more than one authentication method is required for a/all user(s). For example:
  - AuthenticationMethods publickey,password publickey,publickey

- SFTP enhancements
  - Support for recursive file transfer in a directory tree (get/put -r)
  - sftp server read-only mode
  - sftp “df” command - displays file system attributes
  - improved performance of directory listings
  - “ls -h” option - human readable file attribute units
  - ***No*** sftp support for MVS datasets, spool files, etc.

New SMF type 119 records for OpenSSH

- Two new SMF 119 records were added:
  - subtype 94(x'5E') Client connection started record
  - subtype 95(x'5D') Server connection started record
- If SMF recording is configured (in zos_ssh_config / zos_sshd_config), these records will be written just after the user has been authenticated by the server.
- The content of these records is identical, and a subset of other 119 SSH records:
  - standard SMF 119 header
  - common 119 TCP/IP identification section
  - SSH common security section (identifies which algorithm(s) were used)

Other SMF updates

- BPX.SMF permission is now required for ssh client users if SMF recording is enabled, since the **ssh** command is not APF authorized.
  - Hence you may need to provide a UACC(READ) for this facility class, if you want to cut SMF records for all users using ssh command.

- C-level mapping macros in /samples/ssh_smf.h and the assembler mapping macros in SYS1.MACLIB(FOTSMF77) have been updated.

- New algorithm names are added into the related SMF record

- **sftp** client records target path name in subtype 97
  - An additional triplet (section) was added to this record which contains the target (remote) path name for a SFTP client file transfer
  - For SCP, this triplet will be present, but the count (SMF_119SSH_S6Num) and length (SMF_119SSH_S6Len) will be zero.

Other New Functions

Run under TSO OMVS

- ssh client command is enabled to run under TSO OMVS (3270), but prompting for passwords or pass phrases is not allowed.

IdentityKeyRingLabel

- double-quotes are optional when entered from an ssh (or sftp, scp) command line.
- Example:

A key ring named SSHring that is owned by KeyRingOwnerID and a certificate labeled 'my label with blanks' is as follows:

```
IdentityKeyRingLabel="KeyRingOwnerID/SSHring my label with
blanks"
```

If the option is specified as a command-line option, you would issue:

```
-o IdentityKeyRingLabel=KeyRingOwnerID/SSHring my label with
blanks
```

Migration Considerations - Config Files

- Customers with prior releases should review their configuration files to determine applicability of new features. Many new configuration options have been added through OpenSSH 6.4, and defaults for others have been changed.
  - ssh_config
  - sshd_config
  - zos_ssh_config
  - zos_sshd_config
- As in previous releases, protocol 1 is disabled by default.
- RhostsAuthentication (protocol 1 only) was removed in OpenSSH 3.7 and is no longer supported. RhostsRSAAuthentication may be used as a more secure alternative.

Migration Considerations - SFTP

- OpenSSH 6.4 changes sftp so that non-error messages are not printed to stdout if running a batch file (-b).
  - In effect, the -q (quiet mode) option is turned on with -b and cannot be turned off.
  - Since this will impact many customers, it has been changed in OpenSSH for z/OS (HOS1130) so that -b does not force -q.
  - The -q option can be specified in addition to -b.   Therefore this is not actually a migration action, but the behavior will not be consistent with other implementations.

Migration Considerations - /dev/random

- In OpenSSH 6.4 no longer supports the use of `ssh-rand-helper`
- In HOS1130, neither the ssh client or sshd server will run unless the UNIX /dev/random device is working.
  - ICSF support of /dev/random is now **REQUIRED**.
    - Version HCR7780 or later must be installed and running
    - With HCR77A0, a **crypto card is NOT required**!
    - With HCR77A1, CSFRNG check can be skipped by defining resource `CSF.CSFSERV.AUTH.CSFRNG.DISABLE` in class `XFACILIT`
  - If `/dev/random` is not available, then ssh/sshd will fail with:

```
FOTS1949 PRNG is not seeded.  Please activate the
Integrated Cryptographic Service Facility (ICSF)
```

Using ICSF to enable /dev/random

- Need to allow required users access to ICSF CSFRNG service. For most environments, this can be granted to all:

```
RDEFINE CSFSERV CSFRNG UACC(NONE)

PERMIT CSFRNG CLASS(CSFSERV) ID(*) ACCESS(READ)

SETROPTS RACLIST(CSFSERV) REFRESH
```

- You must authorize all userids that uses ssh including both **sshd** userids.

- **Note:** With HCR77A1, this can be skipped by defining resource `CSF.CSFSERV.AUTH.CSFRNG.DISABLE` in class `XFACILIT`

To test (from a normal z/OS user UNIX shell):

```
$ head /dev/random | od -x
```

ICSF Cipher and MAC Acceleration

- ICSF must be active
- CPACF - processor feature 3863 (free and enabled by default in most countries)
- Properly configured, ICSF and CPACF instructions can reduce overall CPU usage by > 50%.
- PTF for APAR OA45548 must be installed to take advantage of AES-CTR mode.

ICSF Cipher and MAC Acceleration

The following CSFSERV profiles control access:

- CSFIQA - ICSF Query Algorithm

```
RDEFINE CSFIQA CLASS(CSFSERV) UACC(NONE)
```

- CSF1TRC - PKCS #11 Token record create

```
RDEFINE CSF1TRC CLASS(CSFSERV) UACC(NONE)
```

- CSF1TRD - PKCS #11 Token record delete

```
RDEFINE CSF1TRD CLASS(CSFSERV) UACC(NONE)
```

- CSF1SKE - PKCS #11 Secret key encrypt

```
RDEFINE CSF1SKE CLASS(CSFSERV) UACC(NONE)
```

- CSF1SKD - PKCS #11 Secret key decrypt

```
RDEFINE CSF1SKD CLASS(CSFSERV) UACC(NONE)
```

- CSFOWH - One-Way Hash Generate

```
RDEFINE CSFOWH CLASS(CSFSERV) UACC(NONE)
```

ICSF Cipher and MAC Acceleration

To permit all, some users, or a group

```
PERMIT  CSFIQA  CLASS(CSFSERV) ID(*) ACCESS(READ)
...
SETROPTS CLASSACT(CSFSERV)
SETROPTS RACLIST(CSFSERV) REFRESH
```

Note: You must authorize all userids that use ssh including both **sshd** userids.

ICSF Cipher and MAC Acceleration

- Configuration of `sshd_config` and `ssh_config` Ciphers and MACs options
  - The HOS1130 shipped versions of these files are optimized to choose the best fit with conventional OpenSSH installations along with ICSF acceleration
  - See the guide for information/implications reordering these lists
- Update both z/OS specific configuration files:
  - `/etc/ssh/zos_ssh_config` and `/etc/ssh/zos_sshd_config`

```
# Use either software or ICSF for Ciphers and MACs

CiphersSource any

MACsSource any
```

HCR77A1 performance enhancement option

```
RDEFINE CSF.CSFSERV.AUTH.CSFOWH.DISABLE CLASS(XFACILIT) UACC(READ)
RDEFINE CSF.CSFSERV.AUTH.CSFRNG.DISABLE CLASS(XFACILIT) UACC(READ)
SETROPTS CLASSACT(XFACILIT)
SETROPTS RACLIST(XFACILIT) REFRESH
```

- Defining these profiles in the XFACILIT class will disable SAF/RACF checks for CSFOWH (hash) and CSFRNG (random number) APIs.
- Since ICSF uses CPACF instructions for these anyway (which can’t be protected by SAF/RACF), this is usually an acceptable option.

References

- **IBM Ported Tools for z/OS:** OpenSSH User's Guide (Pub #: SA23-2246-***03***)
- **Website References:**
  - IBM Ported Tools for z/OS - OpenSSH http://www.ibm.com/servers/eserver/zseries/zos/unix/ported/openssh/
  - OpenSSH http://www.openssh.org/
- **ICSF Reference Guides:**
  - z/OS Cryptographic Services ICSF Overview (Pub #: SA22-7519)
  - z/OS Cryptographic Services ICSF Administrator's Guide (Pub #: SA22-7521)
  - z/OS Cryptographic Services ICSF System Programmer's Guide (Pub #: SA22-7520)
  - z/OS Cryptographic Services ICSF Application Programmer's Guide (Pub #: SA22-7522)
  - z/OS Cryptographic Services Writing PKCS #11 Applications (Pub #: SA23-2231)