SHARE San Jose 2017

Finding the Needle in a Haystack – Diagnosing Common OpenSSH Problems

Kirk Wolf
Stephen Goetze

Dovetailed Technologies, LLC

Trademarks

- Co:Z® is a registered trademark of Dovetailed Technologies, LLC
- z/OS® is a registered trademark of IBM Corporation

Dovetailed Technologies

- Why are we presenting this topic?
  - Co:Z Co-Processing Toolkit for z/OS
    - Our product relies on IBM z/OS OpenSSH
    - 10 years of expertise supporting our customers

Agenda

- Basic understanding of SSH
  - SSH-2 RFC overview
  - Illustrated using a successful connection trace
- A strategy for solving SSH problems
- What is important (and not) in SSH logging
- Diagnosing common SSH problems
- Making sure z/OS OpenSSH is tuned properly

SSH Protocol Overview (1/4)

SSH-2 is a layered architecture defined by RFCs 4250-4254

From the bottom up:

TCP/IP

- A single duplex, transparent, byte-oriented connection
- "reliable" (but not secure)

SSH Protocol Overview (2/4)

SSH Transport Layer (SSH-TRANS)

- Responsible for privacy, integrity, compression, server authentication

1. Starts a single TCP connection, defines a packet layer on it
2. Negotiates SSH protocol version, exchange partner software versions
3. Performs Key Exchange (note: periodically redone)
   a. negotiation of algorithms for: kex, server key, cipher, mac, compression
   b. session key exchange (using selected kex algorithm)
      – server (host) authentication
   c. encryption starts....

SSH Protocol Overview (3/4)

SSH Authentication Layer (SSH-AUTH)

- Sits on Transport Layer, starts after key exchange
- Responsible for ***client*** authentication as a *userid on the server system.*
- Available methods are negotiated and can include:
  - password, publickey, hostbased, keyboard-interactive, gssapi (kerberos), …
- Server can require that client authenticate with more than one method

SSH Protocol Overview (4/4)

SSH Connection Layer (SSH-CONN)

- Also sits on Transport layer, starts after SSH-AUTH
- Provides for multiple, simultaneous "channels" to be multiplexed over the connection
- Channel types:
  - terminal ("shell")
  - remote program execution ("command")
  - "subsystem" (like command, used for sftp)
  - TCP and X port forwarding

SSH troubleshooting strategy

1. Look at client or server log messages to see which stage failed
   a. SSHD server messages are logged via syslogd daemon
   b. retry client and server in debug mode for more information
2. Consult the Google or FAQs to see if this has been seen and solved
3. Ask for help
   – http://dovetail.com/forum
   – comp.security.ssh newsgroup (e.g. via Google groups)
   – A Unix/Linux person

(and before anything, do a good job of configuring/tuning OpenSSH)

Running sshd in debug mode

1. Have an alternate listen port reserved for z/OS SSHD debugging
   - Instead of default port = 22, and open through firewalls

2. From a z/OS Unix shell:

```
> su    # you need to run as a uid=0 user
> /usr/sbin/sshd -ed -p 822
```

(-e -> messages will go to stderr rather than syslogd
-d -> DEBUG1 level messages; -dd, or -ddd for DEBUG2 / DEBUG3)

Running ssh client in debug mode

From a Unix shell (z/OS, *nix, etc):

```
> ssh -v -p 822 user@host
```

(-vv or -vvv for DEBUG2/DEBUG3)

Note: you cannot use a TSO OMVS shell to enter passwords; use an ssh shell connection.

A note about example logs

- The logs shown in this presentation might vary slightly from what you see.
  - more messages if you enable features like:
    ICSF, SMF, use of KeyRings, etc
  - different algorithms
  - different versions of OpenSSH
- Good news: the differences usually won't matter much

Quiz: Are these messages important?

```
…
debug3: Not a RSA1 key file ~/.ssh/id_rsa.
debug2: key_type_from_name: unknown key type '-----BEGIN'
debug3: key_read: missing keytype
debug3: key_read: missing whitespace
debug3: key_read: missing whitespace
debug3: key_read: missing whitespace
debug2: key_type_from_name: unknown key type '-----END'
debug3: key_read: missing keytype
```

Successful client log

From a client (z/OS Unix shell) :

```
> ssh -v -p 822 kirk@localhost

OpenSSH_6.4, OpenSSL 1.0.2h  3 May 2016
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: Reading configuration data /etc/ssh/zos_ssh_config
>> TCP connection starting
debug1: Connecting to localhost [127.0.0.1] port 822.
debug1: Connection established.
<< TCP connection started
```

Note: “>>” and “<<“ annotations added to actual log messages

Successful client log (cont.)

```
debug1: cipher_init: none from source OpenSSL, used in non-FIPS mode
debug1: identity file /u/kirk.ssh/id_rsa type 1
debug1: identity file /u/kirk.ssh/id_rsa-cert type -1
debug1: identity file /u/kirk.ssh/id_dsa type -1
debug1: identity file /u/kirk.ssh/id_dsa-cert type -1
debug1: identity file /u/kirk.ssh/id_ecdsa type -1
debug1: identity file /u/kirk.ssh/id_ecdsa-cert type -1
```

Successful client log (cont.)

>> Negotiate protocol version; exchange software versions

```
debug1: Enabling compatibility mode for protocol 2.0
debug1: Local version string SSH-2.0-OpenSSH_6.4
debug1: Remote protocol version 2.0, remote software version
OpenSSH_6.4
debug1: match: OpenSSH_6.4 pat OpenSSH*
```

>> Key Exchange

>> Algorithm negotiation

```
debug1: SSH2_MSG_KEXINIT sent
debug1: SSH2_MSG_KEXINIT received
debug1: mac_setup_by_alg: hmac-sha1 from source OpenSSL …
```

Successful client log (cont.)

```
debug1: kex: server->client aes128-cbc hmac-sha1 none
debug1: mac_setup_by_alg: hmac-sha1 from source OpenSSL, …
debug1: kex: client->server aes128-cbc hmac-sha1 none
debug1: choose_kex: ecdh-sha2-nistp256 from source OpenSSL, …
```

<< Algorithm negotiation complete

>> Session key exchange

```
debug1: sending SSH2_MSG_KEX_ECDH_INIT
debug1: expecting SSH2_MSG_KEX_ECDH_REPLY
```

>> server (host) key verification

```
debug1: Server host key: RSA MD5 fp
        2b:6e:a8:88:49:7e:af:60:a6:0a:10:c9:3c:b4:c4:ce
debug1: checking without port identifier
debug1: Host 'localhost' is known and matches the RSA host key.
debug1: Found key in /u/kirk/.ssh/known_hosts:12
debug1: found matching key w/out port
```

Successful client log (cont.)

```
debug1: ssh_rsa_verify: signature correct
```

<< Server (host) key verification done

<< Session key exchange done

>> Encryption starts

```
debug1: cipher_init: aes128-cbc from source OpenSSL, …
debug1: SSH2_MSG_NEWKEYS sent
debug1: expecting SSH2_MSG_NEWKEYS
debug1: cipher_init: aes128-cbc from source OpenSSL, …
debug1: SSH2_MSG_NEWKEYS received
```

<< Key Exchange done

Successful client log (cont.)

>> SSH-AUTH starts

```
debug1: SSH2_MSG_SERVICE_REQUEST sent
debug1: SSH2_MSG_SERVICE_ACCEPT received
debug1: Authentications that can continue: publickey,password
debug1: Next authentication method: publickey
debug1: Offering RSA public key: /u/kirk/.ssh/id_rsa
debug1: Server accepts key: pkalg ssh-rsa blen 279
debug1: read PEM private key done: type RSA
debug1: Authentication succeeded (publickey).
Authenticated to localhost ([127.0.0.1]:822).
```

<< SSH-AUTH complete

Successful client log (cont.)

>> SSH-CONN starts

```
debug1: channel 0: new [client-session]
debug1: Requesting no-more-sessions@openssh.com
debug1: Entering interactive session.
you have mail in /usr/mail/KIRK.
```

(user types "exit" or Cntrl-D to finish session)

```
debug1: client_input_channel_req: channel 0 rtype exit-status reply 0
debug1: client_input_channel_req: channel 0 rtype eow@openssh.com
reply 0
debug1: channel 0: free: client-session, nchannels 1
```

<< SSH-CONN done

```
Connection to localhost closed.
```

<< SSH-TRANS done

<< TCP connection closed

Successful server log

```
> /usr/sbin/sshd -ed -p 822

>> TCP connection started
Connection from 127.0.0.1 port 1050
>> Negotiate protocol version; exchange software versions
debug1: Client protocol version 2.0; client software version OpenSSH_6.4
debug1: match: OpenSSH_6.4 pat OpenSSH*
debug1: Enabling compatibility mode for protocol 2.0
debug1: Local version string SSH-2.0-OpenSSH_6.4
<<
Port of Entry information retained for uid:0  pid:50397224.
debug1: permanently_set_uid: 500/500 [preauth]
debug1: list_hostkey_types: ssh-rsa,ssh-dss [preauth]
```

Successful server log (cont.)

>> Key exchange

>> Alg negotiation

```
debug1: SSH2_MSG_KEXINIT sent [preauth]
debug1: SSH2_MSG_KEXINIT received [preauth]
debug1: mac_setup_by_alg: hmac-sha1 from source OpenSSL, …
debug1: kex: client->server aes128-cbc hmac-sha1 none [preauth]
debug1: mac_setup_by_alg: hmac-sha1 from source OpenSSL, …
debug1: kex: server->client aes128-cbc hmac-sha1 none [preauth]
debug1: choose_kex: ecdh-sha2-nistp256 from source OpenSSL, …
```

<< Alg negotiation done

Successful server log (cont.)

>> Session key exchange

```
debug1: expecting SSH2_MSG_KEX_ECDH_INIT [preauth]
debug1: cipher_init: aes128-cbc from source OpenSSL, …
debug1: SSH2_MSG_NEWKEYS sent [preauth]
debug1: expecting SSH2_MSG_NEWKEYS [preauth]
debug1: cipher_init: aes128-cbc from source OpenSSL, …
debug1: SSH2_MSG_NEWKEYS received [preauth]
debug1: KEX done [preauth]
```

<< Session key exchange done

<< Key exchange done

Successful server log (cont.)

>> SSH-AUTH starts

```
debug1: userauth-request for user kirk service ssh-connection method
none [preauth]
debug1: attempt 0 failures 0 [preauth]
debug1: userauth-request for user kirk service ssh-connection method
publickey [preauth]
debug1: attempt 1 failures 0 [preauth]
debug1: test whether pkalg/pkblob are acceptable [preauth]
debug1: temporarily_use_uid: 7001/4 (e=0/0)
debug1: trying public key file /u/kirk.ssh/authorized_keys
debug1: fd 4 clearing O_NONBLOCK
debug1: matching key found: file /u/kirk.ssh/authorized_keys, line 3
      RSA MD5 fp d1:6b:c8:85:2d:77:2e:8c:2c:34:d3:be:80:30:d1:41
```

Successful server log (cont.)

```
debug1: restore_uid: 0/0
Postponed publickey for kirk from 127.0.0.1 port 1050 ssh2 [preauth]
debug1: userauth-request for user kirk service ssh-connection method
publickey [preauth]
debug1: attempt 2 failures 0 [preauth]
debug1: temporarily_use_uid: 7001/4 (e=0/0)
debug1: trying public key file /u/kirk.ssh/authorized_keys
debug1: fd 4 clearing O_NONBLOCK
debug1: matching key found: file /u/kirk.ssh/authorized_keys, line 3
RSA MD5 fp d1:6b:c8:85:2d:77:2e:8c:2c:34:d3:be:80:30:d1:41
debug1: restore_uid: 0/0
debug1: ssh_rsa_verify: signature correct
```

Successful server log (cont.)

```
Accepted publickey for kirk from 127.0.0.1 port 1050 ssh2: RSA MD5 fp
d1:6b:c8:85:2d:77:2e:8c:2c:34:d3:be:80:30:d1:41
debug1: monitor_child_preauth: kirk has been authenticated by
privileged process
```

<< SSH-AUTH done

```
debug1: mac_setup_by_alg: hmac-sha1 from source OpenSSL, …
debug1: mac_setup_by_alg: hmac-sha1 from source OpenSSL, …
debug1: monitor_read_log: child log fd closed
User child is on pid 50397228
debug1: permanently_set_uid: 7001/4
debug1: cipher_init: aes128-cbc from source OpenSSL, …
debug1: cipher_init: aes128-cbc from source OpenSSL, …
```

Successful server log (cont.)

>> SSH-CONN starts

```
debug1: Entering interactive session for SSH2.
debug1: server_init_dispatch_20
debug1: server_input_channel_open: ctype session rchan 0 win 1048576..
debug1: input_session_request
debug1: channel 0: new [server-session]
debug1: session_new: session 0
debug1: session_open: channel 0
debug1: session_open: session 0: link with channel 0
debug1: server_input_channel_open: confirm session
debug1: server_input_global_request: rtype no-more-
sessions@openssh.com want_reply 0
debug1: server_input_channel_req: channel 0 request pty-req reply 1
```

Successful server log (cont.)

```
debug1: session_by_channel: session 0 channel 0
debug1: session_input_channel_req: session 0 req pty-req
debug1: Allocating pty.
debug1: session_new: session 0
debug1: session_pty_req: session 0 alloc /dev/ttyp0001
debug1: server_input_channel_req: channel 0 request shell reply 1
debug1: session_by_channel: session 0 channel 0
debug1: session_input_channel_req: session 0 req shell
```

<< SSH-CONN start complete (user is logged into shell)

Successful server log (cont.)

(user exits from shell)

```
debug1: Received SIGCHLD.
debug1: session_by_pid: pid 50397293
debug1: session_exit_message: session 0 channel 0 pid 50397293
debug1: session_exit_message: release channel 0
debug1: session_by_tty: session 0 tty /dev/ttyp0002
debug1: session_pty_cleanup: session 0 release /dev/ttyp0002
```

<< SSH-CONN ends

```
Received disconnect from 127.0.0.1: 11: disconnected by user
```

<< SSH-TRANS end

<< TCP Connection closed

```
debug1: do_cleanup
```

Common SSH problems

1. Firewall blocking connection
2. Mismatching ciphers/macs
3. Server (host) is not known by client
4. Server (host) key has changed
5. User public key not authorized by server
6. Bad file permissions for private key / authorized key
7. User not authorized
8. Bad key ring on client
9. SFTP fails to run
10. SFTP packet corruption
11. Corrupted packets detected by SSH

1) Firewall blocking connection

```
> ssh -v -o connectTimeout=10  192.168.32.59
...
debug1: Connecting to 192.168.32.59 [192.168.32.59] port 22.
debug1: connect to address 192.168.32.59 port 22:
        EDC8127I Connection timed out. (errno2=0x74940000)
FOTS2204 ssh: connect to host 192.168.32.59 port 22:
        EDC8127I Connection timed out. (errno2=0x74940000)
```

1) Firewall blocking connection (2)

Or sometimes the firewall will allow you to connect and then drop you a little later:

```
> ssh -v -o connectTimeout=10  192.168.32.59
...
debug1: Connecting to 192.168.32.59 [192.168.32.59] port 22.
debug1: Connection established.
debug1: cipher_init: none from source OpenSSL
debug1: cipher_init: none from source OpenSSL
debug1: identity file /u/kirk/.ssh/id_rsa type -1
debug1: identity file /u/kirk/.ssh/id_dsa type -1
FOTS1337 ssh_exchange_identification: read:
         EDC8121I Connection reset. errno2=0x76650446)
```

2) Mismatched MACs/Ciphers (client)

```
> sftp -P 4242 -v -o "MACs=hmac-md5-96" lisa@zosdtl13
OpenSSH_7.2p2 Ubuntu-4ubuntu2.1, OpenSSL 1.0.2g-fips  1 Mar 2016
debug1: Reading configuration data /etc/ssh/ssh_config
...
debug1: Connecting to zosdtl13 [192.168.0.49] port 4242.
debug1: Connection established.
...
debug1: Enabling compatibility mode for protocol 2.0
debug1: Local version string SSH-2.0-OpenSSH_7.2p2 Ubuntu-4ubuntu2.1
debug1: Remote protocol version 2.0, remote software version
OpenSSH_6.4
debug1: match: OpenSSH_6.4 pat OpenSSH* compat 0x04000000
debug1: Authenticating to zosdtl13:4242 as 'lisa'
```

2) Mismatched MACs/Ciphers (client)

```
debug1: SSH2_MSG_KEXINIT sent
debug1: SSH2_MSG_KEXINIT received
debug1: kex: algorithm: ecdh-sha2-nistp256
debug1: kex: host key algorithm: ssh-rsa
Unable to negotiate with 192.168.0.49 port 4242: no matching MAC
found. Their offer: hmac-sha1,hmac-sha1-96
Couldn't read packet: Connection reset by peer
```

2) Mismatched MACs/Ciphers (server)

```
...
debug1: SSH2_MSG_KEXINIT sent [preauth]
debug1: SSH2_MSG_KEXINIT received [preauth]
no matching mac found: client hmac-md5-96 server hmac-sha1,hmac-sha1-96 [preauth]
debug1: do_cleanup [preauth]
```

Solution: Specify MACs that include match with server

3) Server (host) not known by client

```
> ssh -v -oBatchmode=yes git@github.com
...
debug1: Connecting to github.com [192.30.253.112] port 22.
debug1: Connection established.
...
debug1: SSH2_MSG_KEXINIT sent
debug1: SSH2_MSG_KEXINIT received
debug1: mac_setup_by_alg: hmac-sha1 from source ICSF, …
debug1: kex: server->client aes128-ctr hmac-sha1 none
debug1: mac_setup_by_alg: hmac-sha1 from source ICSF, …
debug1: kex: client->server aes128-ctr hmac-sha1 none
debug1: choose_kex: ecdh-sha2-nistp256 from source OpenSSL, …
debug1: sending SSH2_MSG_KEX_ECDH_INIT
```

3) Server (host) not known by client

```
debug1: expecting SSH2_MSG_KEX_ECDH_REPLY
debug1: Server host key: RSA MD5 fp 16:27:ac:a5:76:28:2d:36:63:1b:56…
FOTS1370 Host key verification failed.
```

If run from an interactive terminal, the last message will be:

```
  Are you sure you want to continue connecting (yes/no)?
```

Solutions:

a) interactively accept host public key into $HOME/.ssh/known_hosts,
   or use -oStrictHostKeyChecking=no

b) use ssh-keyscan command to get the public key

c) administrator can add to master list /etc/ssh/ssh_known_hosts   (can push to clients)

d) use DNSSEC with SSHFP records (not supported by z/OS OpenSSH)

e) use GSSAPI (Kerberos) key exchange if supported on both ends

4) Server (host) key has changed

```
> ssh -v  kirk@someserver.com
...
debug1: SSH2_MSG_KEXINIT sent
debug1: SSH2_MSG_KEXINIT received
...
debug1: sending SSH2_MSG_KEX_ECDH_INIT
debug1: expecting SSH2_MSG_KEX_ECDH_REPLY
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    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 a host key has just been changed.
```

4) Server (host) key has changed

```
The fingerprint for the RSA key sent by the remote host is
SHA256:Lj21ZqdmT9Dg7Zv3viMY/OmXNGB5qF3FH5jPawkpols.
Please contact your system administrator.
Add correct host key in /home/kirk/.ssh/known_hosts to get rid of this
message.
Offending RSA key in /home/kirk/.ssh/known_hosts:1
remove with:
ssh-keygen -f "/home/kirk/.ssh/known_hosts" -R someserver.com
RSA host key for someserver.com has changed and you have requested strict
checking.
Host key verification failed.
Couldn't read packet: Connection reset by peer
```

Solution: Independently verify correct host public key, remove offending key, and see (3)

5) User public key not authorized by server

(ssh client run from a batch job)

```
> ssh -v -p 822 -i test5_rsa  kirk@localhost acommand
...
debug1: SSH2_MSG_NEWKEYS received
debug1: SSH2_MSG_SERVICE_REQUEST sent
debug1: SSH2_MSG_SERVICE_ACCEPT received
debug1: Authentications that can continue: publickey,password
debug1: Next authentication method: publickey
debug1: Offering RSA public key: test5_rsa
debug1: Authentications that can continue: publickey,password
debug1: No more authentication methods to try.
FOTS1373 Permission denied (publickey,password).
```

5) User public key not authorized by server

(sshd server)

```
Connection from 127.0.0.1 port 1089
...
debug1: SSH2_MSG_NEWKEYS received [preauth]
debug1: KEX done [preauth]
debug1: userauth-request for user kirk service ssh-connection method none
[preauth]
debug1: attempt 0 failures 0 [preauth]
debug1: userauth-request for user kirk service ssh-connection method
publickey [preauth]
debug1: attempt 1 failures 0 [preauth]
debug1: test whether pkalg/pkblob are acceptable [preauth]
debug1: temporarily_use_uid: 7001/4 (e=0/0)
```

5) User public key not authorized by server

```
debug1: trying public key file /u/kirk/.ssh/authorized_keys
debug1: fd 4 clearing O_NONBLOCK
debug1: restore_uid: 0/0
debug1: temporarily_use_uid: 7001/4 (e=0/0)
debug1: trying public key file /u/kirk/.ssh/authorized_keys2
debug1: Could not open authorized keys '/u/kirk/.ssh/authorized_keys2':
EDC5129I No such file or directory. (errno2=0x05620062)
debug1: restore_uid: 0/0
Failed publickey for kirk from 127.0.0.1 port 1089 ssh2: RSA MD5 fp
2a:bb:6e:9b:9b:04:63:99:2f:46:d3:c9:6c:76:e4:4f
```

Solution: put user public key in server userid’s ~/.ssh/authorized_keys

6) Bad file permissions for priv key / authorized_keys

```
> ssh -v -p 822  kirk@localhost
...
debug1: SSH2_MSG_NEWKEYS received
debug1: SSH2_MSG_SERVICE_REQUEST sent
debug1: SSH2_MSG_SERVICE_ACCEPT received
debug1: Authentications that can continue: publickey,password
debug1: Next authentication method: publickey
debug1: Offering RSA public key: /u/kirk/.ssh/id_rsa
debug1: Authentications that can continue: publickey,password
debug1: Trying private key: /u/kirk/.ssh/id_dsa
debug1: Trying private key: /u/kirk/.ssh/id_ecdsa
debug1: Next authentication method: password
kirk@localhost's password:
```

6) Bad file permissions for priv key / authorized_keys

(sshd server)

```
...
debug1: userauth-request for user kirk service ssh-connection method none
[preauth]
debug1: attempt 0 failures 0 [preauth]
debug1: userauth-request for user kirk service ssh-connection method
publickey [preauth]
debug1: attempt 1 failures 0 [preauth]
debug1: test whether pkalg/pkblob are acceptable [preauth]
debug1: temporarily_use_uid: 7001/4 (e=0/0)
debug1: trying public key file /u/kirk/.ssh/authorized_keys
debug1: fd 4 clearing O_NONBLOCK
FOTS2174 Authentication refused: bad ownership or modes for directory
/u/kirk/.ssh
```

Required file / dir permissions

(parent directory(s) of $HOME must be owned and writable only by root)

```
$HOME                     - 7xx   *1
  $HOME/.ssh              - 7xx   *1
    id_dsa, id_rsa,...    - 600   *2
    authorized_keys       - 6xx   *1
    known_hosts           - 6xx   *1
```

*1 must be owned by user and writable only by user or root

*2 must be owned by user and readable and writable only by user or root

7) User not authorized

```
> ssh -v -p 822 larry@localhost
…
debug1: SSH2_MSG_NEWKEYS received
debug1: SSH2_MSG_SERVICE_REQUEST sent
debug1: SSH2_MSG_SERVICE_ACCEPT received
debug1: Authentications that can continue: publickey,password
debug1: Next authentication method: publickey
debug1: Offering RSA public key: /u/larry/.ssh/id_rsa
debug1: Authentications that can continue: publickey,password
debug1: Trying private key: /u/larry/.ssh/id_dsa
debug1: Trying private key: /u/larry/.ssh/id_ecdsa
debug1: Next authentication method: password
```

7) User not authorized

```
larry@localhost's password:
debug1: Authentications that can continue: publickey,password
FOTS1346 Permission denied, please try again.
larry@localhost's password:
```

7) User not authorized

(sshd server)

```
> sshd -ed -p 822
...
Connection from 127.0.0.1 port 1075
...
debug1: KEX done [preauth]
debug1: userauth-request for user larry service ssh-connection method
none [preauth]
debug1: attempt 0 failures 0 [preauth]
FOTS2167 User larry from 127.0.0.1 not valid
input_userauth_request: invalid user larry [preauth]
debug1: userauth-request for user larry service ssh-connection method
publickey [preauth]
```

7) User not authorized

```
debug1: attempt 1 failures 0 [preauth]
debug1: userauth-request for user larry service ssh-connection method
password [preauth]
debug1: attempt 2 failures 1 [preauth]
Failed password for invalid user larry from 127.0.0.1 port 1075…
```

In this case, larry is not a valid user on the server.

The following cases would also look exactly the same ***to the client***:

- a bad password
- user is not listed in sshd_config AllowUsers or AllowGroups
- user is uid=0 and PermitRootLogin=no

8) Problem using SAF/RACF key ring for client key

```
> ssh -v -p 822 -oIdentityKeyRingLabel="SSH-RING SSH-CERT“
              kirk@localhost
...
debug1: Connecting to localhost [127.0.0.1] port 822.
debug1: Connection established.
debug1: cipher_init: none from source OpenSSL, used in non-FIPS mode
FOTS2914 zsshGetKeyFromRecord: Certificate validation for key ring
'SSH-RING' label 'SSH-CERT' failed (53817378). Certificate is expired.
...
debug1: Next authentication method: password
kirk@localhost's password:
```

Solution: either replace the certificate or renew it (resign with same private key)

9) SFTP fails to run

```
> sftp -v -oPort=822 kirk@localhost
...
debug1: Authentication succeeded (publickey).
...
debug1: Sending subsystem: sftp
debug1: client_input_channel_req: channel 0 rtype exit-status reply 0
debug1: client_input_channel_req: channel 0 rtype eow@openssh.com...
debug1: channel 0: free: client-session, nchannels 1
debug1: fd 0 clearing O_NONBLOCK
debug1: fd 1 clearing O_NONBLOCK
Transferred: sent 2440, received 1672 bytes, in 0.6 seconds
Bytes per second: sent 3786.3, received 2594.6
debug1: Exit status 127
FOTS0841 Connection closed
```

9) SFTP fails to run

```
> /usr/sbin/sshd -ed -p 822
...
Accepted publickey for kirk from 127.0.0.1 port 1062 ssh2: RSA MD5 fp
d1:6b:c8:85:2d:77:2e:8c:2c:34:d3:be:80:30:d1:41
...
debug1: Entering interactive session for SSH2.
...
debug1: input_session_request
debug1: channel 0: new [server-session]
debug1: session_new: session 0
debug1: session_open: channel 0
debug1: session_open: session 0: link with channel 0
debug1: server_input_channel_open: confirm session
...
```

9) SFTP fails to run

```
debug1: session_input_channel_req: session 0 req subsystem
subsystem request for sftp by user KIRK
debug1: subsystem: cannot stat /usr/local/coz/bin/sftp-server.sh:
EDC5129I No such file or directory. (errno2=0x053B006C)
debug1: subsystem: exec() /usr/local/coz/bin/sftp-server.sh
debug1: Received SIGCHLD.
debug1: session_by_pid: pid 50397203
debug1: session_exit_message: session 0 channel 0 pid 50397203
debug1: session_exit_message: release channel 0
Received disconnect from 127.0.0.1: 11: disconnected by user
```

Solution: correct sshd_config: Subsystem sftp line

10) SFTP message corruption

```
> sftp -v -oPort=822  kirk@localhost
...
debug1: SSH2_MSG_NEWKEYS received
...
Authenticated to localhost ([127.0.0.1]:822).
debug1: channel 0: new [client-session]
debug1: Requesting no-more-sessions@openssh.com
debug1: Entering interactive session.
debug1: Sending subsystem: sftp
FOTS0843 Received message too long 2743634338
```

10) SFTP message corruption

(sshd server)

```
...
subsystem request for sftp by user KIRK
debug1: subsystem: exec() /usr/local/coz/bin/sftp-server.sh
debug1: Received SIGCHLD.
debug1: session_by_pid: pid 16842829
debug1: session_exit_message: session 0 channel 0 pid 16842829
debug1: session_exit_message: release channel 0
Received disconnect from 127.0.0.1: 11: disconnected by user
Debug1: do_cleanup
```

None of these messages are very helpful!

Don't forget Strategy #2: Google "sftp received message too long"

10) SFTP message corruption

Some causes for corrupted SFTP messages (packets):

- one or more of the following scripts has an "echo" message to stdout:
  sftp-server.rc (Co:Z SFTP), ~/.ssh/rc, /etc/ssh/sshrc, (and others on non-z/OS)

```
# Fix by:
 if test -t 1 ; then   #only for a TTY
   echo "Welcome to my world"
   ...
 fi
```

- an IBM z/OS Comm Server Resolver Trace
  Beyond belief, this can write messages to stdout for all jobs! (why not stderr??)
  If this shuts down all SFTPs at your site, you aren't the first victim.

11) Corrupted packets detected by SSH

```
> sftp somehost
  ...
  ...  (a long sftp transfer or ssh session)
  ...
  FOTS1189 Corrupted MAC on input.
  Disconnecting: Packet corrupt
```

Some causes:

- a bad network card or router corrupted a packet
- or IBM OpenSSH APAR: OA51665: OPENSSL AES128-CTR CIPHER MAY FAIL LARGE TRANSFERS
    - the workaround is....

Making Sure z/OS OpenSSH is Tuned Properly

- The following information is taken from:

  IBM Ported Tools OpenSSH / z/OS V2R2 OpenSSH - Quick Install Guide
  http://dovetail.com/docs/pt-quick-inst/index.html

- For more information, see also:

  IBM Ported Tools for z/OS: OpenSSH
  http://www.ibm.com/servers/eserver/zseries/zos/unix/ported/openssh/index.html

Using ICSF to enable /dev/random

- Required for HOS1130
- 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 use ssh including both **sshd** userids.
- **Note:** With HCR77A1, this can be skipped by defining resource

```
RDEFINE XFACILIT CSF.CSFSERV.AUTH.CSFRNG.DISABLE UACC(READ)
```

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 40-50%.
- 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
  - CSF1TRC - PKCS #11 Token record create
  - CSF1TRD - PKCS #11 Token record delete
  - CSF1SKE - PKCS #11 Secret key encrypt
  - CSF1SKD - PKCS #11 Secret key decrypt
  - CSFOWH - One-Way Hash Generate

ICSF Cipher and MAC Acceleration

```
RDEFINE CSFIQA  CLASS(CSFSERV) UACC(NONE)
RDEFINE CSF1TRC CLASS(CSFSERV) UACC(NONE)
RDEFINE CSF1TRD CLASS(CSFSERV) UACC(NONE)
RDEFINE CSF1SKE CLASS(CSFSERV) UACC(NONE)
RDEFINE CSF1SKD CLASS(CSFSERV) UACC(NONE)
RDEFINE CSFOWH  CLASS(CSFSERV) UACC(NONE)
/* 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, HOS2220 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 XFACILIT CSF.CSFSERV.AUTH.CSFOWH.DISABLE
    UACC(READ)
RDEFINE XFACILIT CSF.CSFSERV.AUTH.CSFRNG.DISABLE
    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.

Verifying ICSF setup

- Run the ssh client under TSO OMVS (new feature!)

```
/SYSTEM/home/user> ssh -vvv myuser@127.0.0.1
…
debug1: zsshVerifyIcsfSetup: ICSF FMID is ‘HCR77A0’
debug2: -----------------------------------
debug2: CRYPTO    SIZE     KEY      SOURCE
debug2: -----------------------------------
debug2: AES       256      CLEAR    CPU
debug2: DES       56       CLEAR    CPU
```

Verifying ICSF setup

```
…
debug2: MDC-2    128       NA       CPU
debug2: MDC-4    128       NA       CPU
debug2: MD5      128       NA       SW
debug2: SHA-1    160       NA       CPU
debug2: SHA-2    512       NA       CPU
debug2: TDES     168       CLEAR    CPU
```

Note: `SOURCE=CPU` means CPACF, which is what ICSF uses for SSH Cipher and MAC acceleration.

Note: The strength/size is the largest bit length supported by the facility. In the display above, AES-128, AES-192, and AES-256 are supported via ICSF with CPACF.

Verifying ICSF setup

…

```
debug1: mac_setup_by_alg: hmac-sha1 from source ICSF
debug1: zsshIcsfMacInit (429): CSFPTRC successful:
return code = 0, reason code = 0, handle = 'SYSTOK-
SESSION-ONLY 00000000S    '
```

Note: These messages indicate that ICSF was used for MAC hmac-sha1

LE Tuning Recommendations

- Ported Tools OpenSSH uses LE XPLINK runtime libraries (like Java, WebSphere, etc)

See: “Placing Language Environment Modules in LPA ..”

- Add SCEELPA to LPALST
- Add SCEERUN and SCEERUN2 to LNKLST
- SCEERUN and SCEERUN2 should be program controlled
- Implement samples CEE.SCEESAMP(CEEWLPA) and (EDCWLPA) as shipped

References

- IBM z/OS V2R2 OpenSSH: User’s Guide
  (Order number: SC27-6806-01)
- IBM Ported Tools for z/OS 1.3.0: OpenSSH User’s Guide
  (Order Number: SA23-2246-03)
- Dovetailed Technologies Resources
  - IBM Ported Tools OpenSSH / z/OS V2R2 OpenSSH Quick Install Guides
  - The Three Headed Dog Ate My SSH Keys! Using OpenSSH in a Single Sign-on Corporate Environment with z/OS, Windows and Linux
    SHARE in San Antonio 2016
  - OpenSSH for z/OS: New Features and Functions
    SHARE in San Antonio 2016
  - IBM Ported Tools for z/OS: OpenSSH - Key Authentication
    Webinar recording also available
  - Community forum http://dovetail.com/forum

References (cont.)

- Website References
  - OpenSSH   http://www.openssh.org/
    - **The C source code is the definitive reference**!
  - comp.security.ssh newsgroup (e.g. via Google groups)
- Books
  - SSH, The Secure Shell: The Definitive Guide 2nd Edition (Barrett et. al.)
  - SSH Mastery: OpenSSH, PuTTY, Tunnels and Keys (Lucas)

Thank You for Attending!

Please remember to complete your evaluation of this session in the SHARE mobile app.

Session 20125:
Finding a Needle in a Haystack - Diagnosing Common OpenSSH Problems