z/OS 2.4 IBM Education Assistant (IEA)

Solution (Epic) Name: **OpenSSH upgrade to 7.6p1**

OpenSSH improve ECDSA support

Element(s)/Component(s): z/OS OpenSSH

Agenda

- Trademarks
- Session Objectives
- Overview
- Usage & Invocation
- Interactions & Dependencies
- Migration & Coexistence Considerations
- Installation
- Session Summary
- Appendix

Trademarks

- See url http://www.ibm.com/legal/copytrade.shtml for a list of trademarks.
- Additional Trademarks:
  - None

Session Objectives

- To introduce the enhancements to z/OS V2R4 OpenSSH

Overview – Non-Executable Memory Support

- Who (Audience)
  - z/OS OpenSSH Admin
- What (Solution)
  - z/OS 2.4 OpenSSH upgraded to OpenSSH 7.6p1.
  - ECDSA key is supported in key rings and FIPS mode.
- Wow (Benefit / Value, Need Addressed)
  - Support for many new functions and crypto algorithms are included, so as to be compatible with other OpenSSH or SSH implementations that wish to use these new functions and algorithms.
  - Additional support for ECDSA user and host keys with ICSF (in Key Rings) and with FIPS mode.

Usage & Invocation

- Key Exchange algorithms (listed in default preference order):
  - **curve25519-sha256**
  - **curve25519-sha256@libssh.org**
  - ecdh-sha2-nistp256*
  - ecdh-sha2-nistp384*
  - ecdh-sha2-nistp521*
  - diffie-hellman-group-exchange-sha256*
  - **diffie-hellman-group16-sha512**
  - **diffie-hellman-group18-sha512**
  - diffie-hellman-group-exchange-sha1*
  - **diffie-hellman-group14-sha256**
  - diffie-hellman-group14-sha1*

Bold = new with z/OS V2.4 (in OpenSSH 7.6.1)

* = supported by ICSF

- The list of available kex algorithms may be obtained using "ssh -Q kex"

Usage & Invocation

- Key algorithms - used for ssh host(server) or user keys (listed in default preference order):
  - ecdsa-sha2-nistp256-cert-v01@openssh.com
  - ecdsa-sha2-nistp384-cert-v01@openssh.com
  - ecdsa-sha2-nistp521-cert-v01@openssh.com
  - **ssh-ed25519-cert-v01@openssh.com**
  - ssh-rsa-cert-v01@openssh.com
  - ecdsa-sha2-nistp256
  - ecdsa-sha2-nistp384
  - ecdsa-sha2-nistp521
  - **ssh-ed25519**
  - ssh-rsa

Bold = new with z/OS V2.4 (in OpenSSH 7.6.1)

ed25519 is a elliptic curve signature scheme that offers better security than ECDSA and DSA and good performance.

- The list of available key algorithms may be obtained using "ssh -Q key"

Usage & Invocation

- Cipher algorithms (listed in default preference order):
  - **chacha20-poly1305@openssh.com**
  - aes128-ctr*#
  - aes192-ctr*#
  - aes256-ctr*#
  - aes128-gcm@openssh.com
  - aes256-gcm@openssh.com

Bold = new with z/OS V2.4 (in OpenSSH 7.6.1)

* = supported by ICSF

\# = supported using CPACF instructions

chacha20-poly1305@openssh.com combines Daniel Bernstein's ChaCha20 stream cipher and Poly1305 MAC to build an authenticated encryption mode

- The list of available Cipher algorithms may be obtained using "ssh -Q cipher"

Usage & Invocation

- Mac algorithms (listed in default preference order):
  - hmac-sha2-256-etm@openssh.com*#
  - hmac-sha2-512-etm@openssh.com*#
  - hmac-sha1-etm@openssh.com*#
  - hmac-sha2-256*#
  - hmac-sha2-512*#
  - hmac-sha1*#
  - umac-64-etm@openssh.com
  - umac-128-etm@openssh.com
  - umac-64@openssh.com
  - umac-128@openssh.com

* = supported by ICSF
= supported using CPACF instructions

- The list of available MAC algorithms may be obtained using “ssh -Q mac”

Usage & Invocation

- New command: **ssh-proxyc** - HTTP SOCKS-5 Proxy command for ssh client
  - ssh-proxyc enables an ssh client to connect through a SOCKS-5 proxy to remote host
  - Some installations do not allow for direct ssh outbound communication, but require connection through a SOCK5 proxy server. The ssh option "ProxyCommand" can specify an external program that will perform the SOCKS negotiation.
  - The ssh-proxyc command requires the ssh "ProxyUseFdPass" option, which supports passing the fd for the connected socket back to the ssh client so that once the SOCKS negotiation is complete, the proxy command can exit and not be required for the I/O.

Usage & Invocation

Format: ssh-proxyc [-46Ehv] -p proxy_address[:port] destination [port]

- -4 Use IPv4
- -6 Use IPv6
- -E Disable EBCIDIC-ASCII conversions for SOCKS negotiation
- -h help text
- -v Verbose
- -p addr[:port] Specify proxy address and port

Example:

```
ssh -oProxyUseFdpass=yes -oProxyCommand='ssh-proxyc -p
socks_server:1080 %h %p' user@remote_host
```

Or, in ssh_config:

```
Host *.mydomain.com

ProxyCommand ssh-proxyc -p socks_server:1080 %h %p
```

Usage & Invocation

Other new functions in OpenSSH 7.6.1:

- Support “=+”and “=-” syntax to easily append or remove methods from algorithm lists.
  - For example: “HostKeyAlgorithms=+ssh-dss”, “Ciphers=-aes128-cbc”

- Support UNIX domain socket forwarding.
  - A remote TCP port may be forwarded to a local UNIX domain socket and vice versa or both ends may be a UNIX domain socket.

- Support client-side hostname canonicalization using a set of DNS suffixes and rules in ssh_config.

Usage & Invocation

Other new functions in OpenSSH 7.6.1:

- More flexibility in configuration files. Match blocks have more criteria and can include more options within the block.

- The hash algorithm used when displaying key fingerprints can be specified with FingerprintHash option for ssh and sshd and –E option for ssh-keygen. The valid options are “md5” and “sha256”, the default is “sha256”.

- Support SHA256 and SHA512 RSA signatures in certificates.

Usage & Invocation

- Elliptic-curve DSA (ECDSA) user and host keys are supported with ICSF (in Key Rings) and with FIPS or no-FIPS mode.
  - ECDSA keys must use the NIST curves of size 256, 384, or 521 bits.

    Example - generate a certificate and a ECDSA host public/private key pair using the NIST p256 curve stored in key rings:

```
RACDCERT GENCERT ID(SSHDAEM) SUBJECTSDN(CN('host-ssh-
ecdsa-cn')) SIZE(256) NISTECC WITHLABEL('host-ssh-
ecdsa’)
```

Usage & Invocation

- Key Ring keys will now use Systems SSL for signing and verification, regardless of whether in FIPS mode.
  - This will allow key ring private keys to be stored in ICSF.
  - This change may require that customers authorize key ring users to the CSFDSG and CSFDSV resources in the CSFSERV class.

Interactions & Dependencies

- To exploit this item, all systems in the Plex must be at the new z/OS level: No

- Software Dependencies
  - None
- Hardware Dependencies
  - None
- Exploiters
  - N/A

Migration & Coexistence Considerations

- z/OS V2.4 OpenSSH does not support:
  - SSH Version 1 protocol (also referred to as SSH-1).
  - Running without privilege separation for sshd (SSH Daemon).
  - Support for the legacy v00 OpenSSH cert format.
  - Support for pre-authentication compression by sshd (SSH Daemon). SSH clients will either need to support delayed compression mode or otherwise compression will not be negotiated.
  - Support for Blowfish and RC4 ciphers and the RIPE-MD160 HMAC (Hash Message Authentication Code).
  - Accepting RSA keys smaller than 1024 bits.

Migration & Coexistence Considerations

- In addition, z/OS V2.4 OpenSSH will not have the following functions enabled by default:
  - Support for the 1024-bit Diffie Hellman key exchange, specifically diffie-hellman-group1-sha1.
  - Support for ssh-dss, ssh-dss-cert-* host and user keys.
  - Support for MD5-based and truncated HMAC algorithms, specifically hmac-sha1-96.
  - Support for the Triple DES cipher, specifically 3des-cbc, in the SSH client's default algorithm proposal.

Installation

- No special considerations

- Verifying version:

  $ ssh -V

  OpenSSH_7.6p1, OpenSSL 1.0.2h 3 May 2016

Session Summary

- The following z/OS OpenSSH Epics have been explained:
  - 187995 OpenSSH upgrade to 7.6p1
  - 187994 OpenSSH improve ECDSA support

- Upgrade to OpenSSH 7.6p1 provides various functional, performance and security requirements.

- ECDSA keys are now supported in key rings and FIPS mode.

Appendix

- z/OS OpenSSH User's Guide
- Open source reference guide:
  - OpenSSH http://www.openssh.org/
  - OpenSSL http://www.openssl.org/