Configuring SCP push of mail logs on ESA

Contents

Introduction

This document describes how to setup and configure secure copy push (SCP) of mail logs (or other log types) from a Cisco Email Security Appliance (ESA) to an external syslog server.

Background Information

An Administrator may receive error notifications stating that logs cannot be pushed using SCP, or there may be error logs stating key(s) mismatch.

Prerequisites

On the syslog server that the ESA will SCP log files to:

1. Assure that the directory to be used is available.
2. Review '/etc/ssh/sshd_config' for the AuthorizedKeysFile settings. This tells SSH to accept authorized_keys and look in the user's home directory for key_name sting written in .ssh/authorized_keys file: `AuthorizedKeysFile %h/.ssh/authorized_keys`
3. Verify the permissions of the directory to be used. You may need to make permissions changes: Permissions on '$HOME' is set to 755.Permissions on '$HOME/.ssh' is set to 755.Permissions on '$HOME/.ssh/authorized_keys' is set to 600.

File Level Restrictions and Permissions on UNIX/Linux

There are three types of access restrictions:

```
Permission Action chmod option ===================================== read (view) r or 4 write (edit) w or 2 execute (execute) x or 1
```

There are also three types of user restrictions:

```
User ls output ================== owner -rwx------ group ----rwx--- other -------rwx
```

Folder/Directory Permissions:

```
Permission Action chmod option ============================================================= read (view contents: i.e., ls command) r or 4 write (create or remove files from dir) w or 2 execute (cd into directory) x or 1
```

Numeric Notation:

Another method for representing Linux permissions is an octal notation as shown by `stat -c %a`. This notation consists of at least three digits. Each of the three rightmost digits represents a different component of the permissions: owner, group, and others.

Each of these digits is the sum of its component bits in the binary numeral system:

```
Symbolic Notation Octal Notation English ========================================================== ---------- 0000 no permissions --- x--x--x 0111 execute --w--w--w- 0222 write --wx-wx-wx 0333 write & execute -r--r--r-- 0444 read -r-xr-xr-x 0555 read & execute -rw-rw-rw- 0666 read & write -rwxrwxrwx 0777 read. write & execute
```

For step #3, recommendation to set the $HOME directory to 755 would be: 7=`rwx` 5=`r-x` 5=`r-x`

This means that the directory has the default permissions `-rwxr-xr-x` (represented in octal notation as 0755).

Configuring SCP push of mail logs on ESA

1. Run the CLI command **logconfig**.
2. Select the option **new**.
3. Choose the log file type for this subscription, this will be "1" for IronPort Text Mail Logs, or any other log file type of your choice.
4. Enter the name for the log file.
5. Select the appropriate log level. Typically you would need to select "3" for Informational, or any other log level of your choice.
6. When prompted 'Choose the method to retrieve the logs', select "3" for **SCP Push**.
7. Enter in the IP address or DNS hostname to deliver the logs to.
8. Enter the port to connect to on the remote host.
9. Enter the directory on remote host to place logs.
10. Enter in a filename to use for log files.
11. Configure, if needed, system based unique identifiers like *$hostname*, *$serialnumber* to append to the log filename.
12. Set Maximum filesize before transferring.
13. Configure time-based rollover of the log files, if applicable.
14. When asked "Do you want to enable host key checking?", enter "Y".
15. You are then presented the "Please place the following SSH key(s) into your authorized_keys file so that the log files may be uploaded."
16. Copy that key, as you will need to put the SSH key in your 'authorized_keys' file on the Syslog server. Paste the key given from logconfig to $HOME/.ssh/authorized_keys file on the Syslog server.
17. From the ESA, run the CLI command **commit** to save and commit configuration changes.

Configuration of the log can also be accomplished from the GUI: **System Administration > Log Subscriptions**

> **Note**: Please review the Logging chapter of the ESA User Guide for complete details and further information.

Confirmation

Hostkeyconfig

Run the command **logconfig** > **hostkeyconfig**. You should see an entry for the syslog server configured listed as "ssh-dss" with an abbreviated key similar to the key provided during configuration.

```
myesa.local > logconfig
...
[]> hostkeyconfig

Currently installed host keys:
1. 172.16.1.100 ssh-dss AAAAB3NzaC1kc3MAAACBAMUqUBGztO0T...OutUns+DY=
```

System Logs

System logs record the following: boot information, virtual appliance license expiration alerts, DNS status information, and comments users typed using commit command. System logs are useful for troubleshooting the basic state of the appliance.

Running the command **tail system_logs** from the CLI will provide you a live look to the system status.

You may also choose the CLI command **rollovernow** and select the number associated to the log file. You will see this the log file SCP to your syslog server in system_logs:

```
myesa.local > tail system_logs

Press Ctrl-C to stop.
Thu Jan 5 11:26:02 2017 Info: Push success for subscription mail_logs: Log
mail_logs.myesa.local.@20170105T112502.s pushed via SCP to remote host 172.16.1.100:22
```

Advanced Troubleshooting

If there are continued issues with connectivity to the syslog server, from local host and using ssh, run "ssh testuser@hostname -v" to test the user access in verbose mode. This may aide troubleshooting to show where the ssh connection is not succeeding.

```
$ ssh testuser@172.16.1.100 -v
OpenSSH_7.3p1, LibreSSL 2.4.1
debug1: Reading configuration data /Users/testuser/.ssh/config
debug1: /Users/testuser/.ssh/config line 16: Applying options for *
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: /etc/ssh/ssh_config line 20: Applying options for *
debug1: Connecting to 172.16.1.100 [172.16.1.100] port 22.
```

```
debug1: Connection established.
debug1: identity file /Users/testuser/.ssh/id_rsa type 1
debug1: key_load_public: No such file or directory
debug1: identity file /Users/testuser/.ssh/id_rsa-cert type -1
debug1: identity file /Users/testuser/.ssh/id_dsa type 2
debug1: key_load_public: No such file or directory
debug1: identity file /Users/testuser/.ssh/id_dsa-cert type -1
debug1: key_load_public: No such file or directory
debug1: identity file /Users/testuser/.ssh/id_ecdsa type -1
debug1: key_load_public: No such file or directory
debug1: identity file /Users/testuser/.ssh/id_ecdsa-cert type -1
debug1: key_load_public: No such file or directory
debug1: identity file /Users/testuser/.ssh/id_ed25519 type -1
debug1: key_load_public: No such file or directory
debug1: identity file /Users/testuser/.ssh/id_ed25519-cert type -1
debug1: Enabling compatibility mode for protocol 2.0
debug1: Local version string SSH-2.0-OpenSSH_7.3
debug1: Remote protocol version 2.0, remote software version OpenSSH_6.6.1p1 Ubuntu-2ubuntu2.8
debug1: match: OpenSSH_6.6.1p1 Ubuntu-2ubuntu2.8 pat OpenSSH_6.6.1* compat 0x04000000
debug1: Authenticating to 172.16.1.100:22 as 'testuser'
debug1: SSH2_MSG_KEXINIT sent
debug1: SSH2_MSG_KEXINIT received
debug1: kex: algorithm: curve25519-sha256@libssh.org
debug1: kex: host key algorithm: ssh-dss
debug1: kex: server->client cipher: chacha20-poly1305@openssh.com MAC: <implicit> compression:
zlib@openssh.com
debug1: kex: client->server cipher: chacha20-poly1305@openssh.com MAC: <implicit> compression:
zlib@openssh.com
debug1: expecting SSH2_MSG_KEX_ECDH_REPLY
debug1: Server host key: ssh-dss SHA256:c+YpkZsQyUwi3tkIVJFXHAstwlkdewO1G0s7P2khV7U
debug1: Host '172.16.1.100' is known and matches the DSA host key.
debug1: Found key in /Users/testuser/.ssh/known_hosts:5
debug1: rekey after 134217728 blocks
debug1: SSH2_MSG_NEWKEYS sent
debug1: expecting SSH2_MSG_NEWKEYS
debug1: rekey after 134217728 blocks
debug1: SSH2_MSG_NEWKEYS received
debug1: Skipping ssh-dss key /Users/testuser/.ssh/id_dsa - not in PubkeyAcceptedKeyTypes
debug1: SSH2_MSG_SERVICE_ACCEPT received
debug1: Authentications that can continue: publickey,password
debug1: Next authentication method: publickey
debug1: Offering RSA public key: /Users/testuser/.ssh/id_rsa
debug1: Authentications that can continue: publickey,password
debug1: Trying private key: /Users/testuser/.ssh/id_ecdsa
debug1: Trying private key: /Users/testuser/.ssh/id_ed25519
debug1: Next authentication method: password
testuser@172.16.1.100's password: <<< ENTER USER PASSWORD TO LOG-IN >>>
debug1: Enabling compression at level 6.
debug1: Authentication succeeded (password).
Authenticated to 172.16.1.100 ([172.16.1.100]:22).
debug1: channel 0: new [client-session]
debug1: Requesting no-more-sessions@openssh.com
debug1: Entering interactive session.
debug1: pledge: exec
debug1: No xauth program.
Warning: untrusted X11 forwarding setup failed: xauth key data not generated
debug1: Requesting authentication agent forwarding.
debug1: Sending environment.
debug1: Sending env LANG = en_US.UTF-8
debug1: Sending env LC_CTYPE = en_US.UTF-8
```