Instructions For Use—CM130 network communications

Purpose

This document explains the networking operations available to the network system operator of the CM130 analyzer. The networking operations include the following list of processes:

- Downloading and installing necessary utilities
- Creation of Public Keys
- Uploading of Public Keys to the CM130 analyzer
- Downloading of Measurement Data from the CM130 analyzer, which includes the:
    - Measurement Data Transfer Process
    - Time Stamp Tagging Process
    - Operations Acknowledgement Process

Scope

This document provides the process and instructions for the networking operations that are necessary for the control and maintenance of the CM130 analyzer networking capabilities.

Definitions, acronyms and abbreviations

Name	Definition
Client	The computer connecting to the analyzer.
Server	The CM130 analyzer hosting the measurement data.
Host	The CM130 analyzer hosting the measurement data.
SSH	Secure SHell
XML	eXtensible Markup Language

Font usage

During the description of these processes, multiple fonts are utilized to help the reader understand the change in context. The fonts and their usages are described below:

- Arial—General process description
- `Courier New`—Exact command to be entered on a shell command line
- ***Arial Bold Italic***—The description of an execution command, file name, or user account name within the body of the explanation of the process.

Overall networking operations description

The CM130 analyzer allows access to measurement information to be used for offline processing. To access this information, the system must be configured for secure network operations. Each external system that is used to access any particular CM130 analyzer must have a matching private/public network key stored within the system for authentication purposes. The information that follows defines the processes that are used to perform the operations that allow access to the measurement data stored on the CM130 analyzer.

The private/public key is used for the passwordless network access of the ***nobody*** user account on the CM130 analyzer. To accomplish this system network update, a special user account is required, ***keyxfer***. When the ***keyxfer*** account is first accessed, the default password must be used. The ***keyxfer*** user account can only access the system using a password. The ***keyxfer*** user account will only be used when the private/public key information for the ***nobody*** user account needs to be updated, so please do not forget the password. The ***nobody*** user account is only used to pull measurement data from the CM130 analyzer.

The following processes are done to setup and execute the secure network access capabilities of the CM130 analyzer:

1. Obtain and install processing utilities.
2. Create the files necessary for uploading to the CM130 analyzer.
3. Log in to the ***keyxfer*** user account.
4. Perform the Key Transfer process to upload the public key(s) to the individual CM130 analyzer from which the measurement data will be obtained.
5. Perform the Measurement Data Transfer process to download the measurement data, acknowledge the reception and authenticate the transfer.

Communication process

The CM130 analyzer can be connected to a local network. Once connected to the user's local network and credentials are established, the CM130 analyzer allows the transfer of measurement data to a remote client. The ability to transfer measurement data to a remote client lets the user remotely monitor the analyzer data and build data trends. Such an ability requires cyber security rules and processes to be in place to meet the needs of addressing networking attacks. As such, the CM130 analyzer has provisions built into the system that a user needs to comply with in order to allow the remote data transfers.

A specific set of steps are needed to let the client update the public keys on the CM130 for the remote data access account. The client must follow a specific set of steps to interface the client with the server and maintain a safe networking connection between the two. The client can also send the updated public keys file to the CM130, providing access to pull measurement data from the server.

The following are the steps the CM130 analyzer will follow to receive, validate and copy the new file to its final destination:

1. Accept a valid connection request.
2. Receive the key file via SCP.
3. Receive the md5sum file.
4. Receive the sha1sum file.
5. Within the next 15 minute boundry, the CM130 analyzer will:
    a. Validate the md5sum file.
    b. Validate the sha1sum file.
    c. If all previous steps pass, then move on to the final step, else fail process.
    d. The CM130 analyzer will move the validated file to its final place.

All new connections that attempt to pull measurement data with the ***nobody*** user account will authenticate with the new public keys.

Downloading and installing necessary utilities

Downloading and installing the file conversion utility

Download the dos2unix file conversion utility using the following link:

https://sourceforge.net/projects/dos2unix/

Downloading and installing the PuTTY application

The PuTTY application lets the user log in to the remote server via the SSH protocol. Additional, useful applications that are provided with the download package are *puttygen.exe, plink.exe* and *pscp.exe*.

The *puttygen.exe* application lets the user create a private/public key pair combination that lets users, or automated scripts, log in to a remote server that is running the *sshd* application, which will listen on port 22 for an SSH connection.

The *plink.exe* application lets the user connect to a remote host via the SSH protocol using either command line or scripting capabilities to allow remote command execution.

The *pscp.exe* application allows the copying of files from the client to a remote server using either command line or scripting capabilities.

To download the installer, go to the following URL:

http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html

Once at the URL, navigate and download the application from the available link. Download and install the application.

Note: *If the default locations are used for the installation, make sure to add the default location to the system PATH environment variable. This will prevent the need to type the entire path to the utility each time you execute the application.*

Download file creation

Criteria for all files

- All files must be ASCII based.
- All files must be a plain, flat text file.
- All file formatting must be a *NIX based file with each line terminating via the LF character.

  CR+LF files will not work.

Create/Update the public keys file

The CM130 analyzer lets one or more clients remotely connect to the analyzer and download measurement data. The ability of the client to connect to the analyzer and download the measurement data is governed by public keys that are artifacts of the remote clients' private keys. Each client has a private key and shares the public key artifact with the analyzer. The following are the steps for the analyzer to accept, validate and install the public keys:

- The filename must be: **keys-file.pub**
- The collection of keys must be contained within a single file.
- Each key must have the following format:
  - ssh-rsa <encryption stream> <**optional description>** <LF>
  - This key may be obtained from the PuTTYgen utility by copying and pasting the keyvalue from the display window within the displayed dialog.

    Example:

    ```
    ssh-rsa
    AAAAB3NzaC1yc2EAAAABJQAAAQEAn5/BsDCIsjJtkTqhkC8c0PVPRQ8yZiJ9tgBkaxF6m2sj
    +AP1rAYPqL/3L0z0SrLA+JniDTeQADrDIZ70RL0hB+bGHBZr8M22e8m3CbInP1w7TZehiW76sZI
    +xPmzK+E63wAeFY6eMlZYMs73AyRpkd89vtUlf3/NcksvhQGza/aFcar
    +9xIxf61IJOjobJugJLptju0UY/DEM8HLGBLXseY83kY60zLjfNLehtwbHmxpHqHmMJ6fs6xqFTGaR
    yyKoTxd/FJTY7YhHsbWfIgFnANq36tsckkbKtsSxkHNeikQpXOmVMM5Z2V2WSGxbWYBJuOMQuReAH1
    Fg3yPtX5Emw== rsa-key-20160915
    ```
- Each key is SSH-2 RSA encrypted.

This file may be created using the following process:

1. Open the PuttyGen utility.
2. Perform the necessary operations to generate the key.
3. Save the private key to a known location for use in the measurement data downloading process.
4. Do not save the public key using the save button.
5. Copy the keyvalue from the display window within the displayed dialog.
6. Open the keys-file.pub file in Notepad.
7. Paste the keyvalue to the end of the file using Notepad.
8. Save the file and close Notepad.
9. Execute the following command line:
   ```
   dos2unix keys-file.pub
   ```

You now have a properly formatted keys-file.pub file that is ready to be downloaded to the CM130 analyzer.

Create MD5 sum file

The public key file must be accompanied by an MD5 sum hash file and must be named MD5SUM.

- The file name must be: **`MD5SUM`**
- This file will have only a single line that contains the MD5 sum for the key file.
- Example:

  ```
  3062b9ad5a705aa065955c8df022c891 keys-file.pub
  ```

This file may be created using the following process:

1. Open a Windows command (cmd.exe) window.
2. Change the directory to the directory where the key file is located.
3. Execute the following command line (one line):
   ```
   for /f %a in ('certutil -hashfile keys-file.pub MD5 ^| findstr ^1') do echo %a
   keys-file.pub > MD5SUM
   ```
4. Execute the following command line: `dos2unix MD5SUM`

You now have a properly formatted MD5SUM file that is ready to be downloaded to the CM130 analyzer.

Create SHA1 sum file

The public key file must be accompanied by a SHA1 hash file and must be named SHA1SUM.

- The file name must be: **`SHA1SUM`**
- This file will have only a single line that contains the SHA1 sum for the key file.
- Example:
  ```
  f862a804df49fe7c6b7dc3628eef160a05304481 keys-file.pub
  ```

This file may be created using the following process:

1. Open a Windows command (cmd.exe) window.
2. Change the directory to the directory where the key file is located.
3. Execute the following command line (one line):
   ```
   for /f %a in ('certutil -hashfile keys-file.pub SHA1 ^| findstr ^1') do echo %a
   keys-file.pub > SHA1SUM
   ```
4. Execute the following command line: `dos2unix SHA1SUM`

You now have a properly formatted SHA1SUM file that is ready to be downloaded to the CM130 analyzer.

Log in to the keyxfer user account

The CM130 analyzer lets the client connect using a SSH connection and the keyxref user account.

- **Login name—**keyxfer
- **Password—**1@34%^YtrEwq

Note: *To change the password, refer to Change the password for the keyxfer user account (optional) on page 6.*

Upload key files

Obtain the initial key file datetime

- The following process assumes that the user is working within a Windows command window.
- To validate that the public keys file was updated, the current keyfile date needs to be gathered. Run the following command, substituting the proper <ip address>:
  ```
  plink keyxfer@<ip address>: public-key-datetime
  ```
- The CM130 analyzer will return a string in the format:
  ```
  CCYY-MM-DD_HH:mm:SS (e.g. 2016-01-28_17:22:41)
  ```
  The string will be read from the currently used public keys file. The following is a breakdown of the entries:
  - CCYY - century and two digit year (e.g. 2016)
  - MM - month [01 .. 12]
  - DD - day of month [01 .. 31]
  - HH - hour of day [00 .. 23]
  - mm - minutes of the hour [00 .. 59]
  - SS - seconds of the minute [00 .. 59]
- Save the returned datetime value for future comparison.

Transfer key files to the CM130 analyzer

1. Open a Windows command (cmd.exe) window.
2. Change the directory to the directory where the previously created key files are located (keys-file.pub, MD5SUM and SHA1SUM).
3. Transfer the public key(s) file by running the following command, substituting the proper <ip address>:
   ```
   pscp -scp keys-file.pub keyxfer@<ip address>:.
   ```
4. Transfer the MD5 sum file by running the following command, substituting the proper <ip address>:
   ```
   pscp -scp MD5SUM keyxfer@<ip address>:.
   ```
5. Transfer the SHA1 sum file by running the following command, substituting the proper <ip address>:
   ```
   pscp -scp SHA1SUM keyxfer@<ip address>:.
   ```
6. Wait for 15 minutes to make sure that the files have been processed by the CM130 analyzer.
7. Run the following command to test if the public key(s) file has been updated, substituting the proper <ip address>:
   ```
   plink keyxfer@<ip address>: public-key-datetime
   ```
   The server will return a formatted date/time string in the format described in the previous section.
8. Compare the returned datetime with the earlier obtained datetime, they should be different. If the datetime values are the same, the key transfer process has failed or you need to wait a bit longer.

This completes the key update process.

Downloading measurement data from the CM130 analyzer

An XML file that contains CM130 measurements (measurement-data.xml) is available on the CM130 analyzer. The client does a remote copy of the XML file to collect measurement data.

- Open a Windows command (cmd.exe) window.
- Change the directory to the directory where you want the data stored.

Securely copying measurement data

Now it is time to copy a file from that server back to the client.

1. Perform a *pscp.exe* file transfer:
   pscp -scp -i <complete path to private key file xxxx.ppk> nobody@<ip address>:measurement-data.xml <destination directory path>

   Example:
   ```
   pscp -scp -i c:\keyStorage\CM-130PrivateKey.ppk
   nobody@192.168.1.2:measurement_data.xml .
   ```

The user will see the file name copied with a progress status (for large files). At this point, the file from the remote server has been copied to the client computer.

Note: *If an error occurs where the key is not accepted by the server, the most probable reason is that the key file above was not stored in the proper format. Re-run the key transfer process, including the generation of the keys. You may reload the private key so that you do not need to recreate the entire key.*

Timestamp process

The timestamp file, *timestamp.txt*, lets the client provide a notice to the host's application of the time and date that the measurement data was transferred to the client. The indication to the host that the client has completed data transfer is a change to the last modified time of *timestamp.txt*.

The host to client transfer indicator is provided to the host when the client executes the following *touch* command while logged into the host:

```
plink -i c:\keyStorage\CM-130PrivateKey.ppk nobody@192.168.1.2: touch timestamp.txt
```

Acknowledgement process

The acknowledgement process lets the client retrieve an indication of when the host application has acknowledged the transfer and purged the old data from the measurement file. The host side application will periodically query the last access time of the *timestamp.txt* file and compare that to the measurement timestamps in the measurement data.

In order for the client to determine if the host has purged old data, a user or script must perform the following:

1. Log into the host via SSH.
2. Run the *acknowledgement* application.
3. Capture the output string [-1, 0..3000]

   - ERROR - (-1): analyzer error - on-site attention needed
   - IDEAL - (0): ideal transfer - there are 0 (zero) messages queued between the data pull request completion and the touch request
   - PLAUSIBLE - (1): there is 1 message queued between the data pull request completion and the touch request (a delay of +0 .. 299 seconds)
   - ATTENTION - (2 .. 3000): there are 2 to 3000 messages queued between the data pull request completion and the touch request; a delay of (((N-1) * 5 minutes) + [0 .. 299]seconds). Where 'N' is the number of queued messages between the data pull completion and the receipt of the 'touch' request

   Example *acknowledgement* execution:

   ```
   plink -i c:\keyStorage\CM-130PrivateKey.ppk
   nobody@192.168.1.10: acknowledgment
   1
   ```

   In the previous example, the client remotely executed the *acknowledgement* command and received a positive integer value, which indicates that the old data was purged since the last time the data was transferred from the host to the client, and any measurements that were buffered have now been written to the next file.

Change the password for the keyxfer user account (optional)

To change the password, execute one of the following command lines:

```
plink -ssh -pw "current-password" keyxfer@cm130-ip-address reset-keyxfer-password
"new-password"
```

or

```
plink -ssh -i nobody-private-key-file nobody@cm130-ip-address reset-keyxfer-password
"new-password"
```

The new password must have:

- At least 8 characters
- Include three different types of characters: lower case letters, numbers, special characters (e.g., @#$%^*, ").

Note: *Empty passwords are not permitted.*

The new password must not include "! & \ / ( ) < > |".

The password types that follow **should not** be used:

- All numbers (e.g., birth dates, social security number, license plate, phone numbers)
- All letters (uppercase, lowercase or mixed) as palindromes, consecutive or repetitive letters or adjacent letters on the keyboard
- Username, real name, host name, company name or address in any form (e.g., reversed, capitalized or doubled)
- Common and obvious letter-number replacements (e.g., replace the letter O with number 0), such as "M1cr0$0ft" or "P@ssw0rd"

Reset the password for the keyxfer user account

If you have forgotten the current password, execute the following command line to reset the password to the default password:

```
plink -ssh -i nobody-private-key-file nobody@cm130-ip-address reset-keyxfer-password
```

The default password is 1@34%^YtrEwq

HACH COMPANY
100 Dayton Ave, Ames, IA 50010 U.S.A.
Tel. (970) 669-3050
(800) 227-4224 (U.S.A. only)
Fax (970) 669-2932
orders@hach.com
www.hach.com