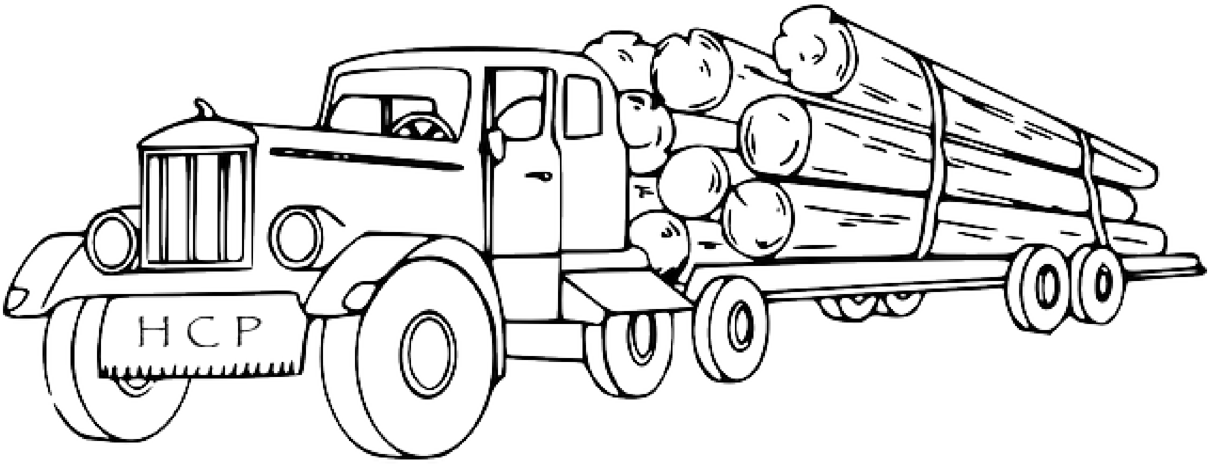


HCP access log collector Documentation

Release 2.0.9

Thorsten Simons

August 02, 2016

Contents

Hitachi Content Platform (HCP) logs access to its REST-based interfaces on a regular basis and keeps these logs for up to 90 days. While access to these logs was possible by downloading the internal logs using the System Management Console (SMC) since HCP version 7.0, it was a quite cumbersome manual task to extract the logs from the provided zip-file.

HCP version 7.2 invented a Management API (MAPI) endpoint that allows to selectively access parts (or all) of the internal logs.

The **HCP access log collector** tool concentrates on downloading the access logs only, enabling users to collect (and archive) these logs on a regular basis.

CHAPTER 1

Setup

1.1 Pre-requisites

- HCP running at least version 7.2
- An system-level admin account having at least the *Admin* and *Monitor* roles
- MAPI enabled at sytem-level
- If the logs are to be archived to an HCP Namespace: a data access user having at least write access to that Namespace

1.2 Dependencies

You need to have at least Python 3.4.3 installed to run **hcplogs**.

It depends on this package, which will be auto-installed if not already available:

- hcpsdk[1] - Used for access to HCP.

You might want to use a virtual environment to fence the dependency from your primary Python environment...

1.3 Installation

Make sure you have Python 3.4.3 (or better) installed

In case it's not installed, get it here[2].

There are two ways to install **hcplogs**:

1. system-wide
   - Install **hcplogs** by running:

```
$ pip install hcplogs
```

   -or-

   - Get the source from gitlab.com[3] either
     - by downloading the source archive, or
     - by cloning the repository:

[1] http://simont3.github.io/hcpsdk/
[2] https://www.python.org/downloads/
[3] https://gitlab.com/simont3/hcplogs

```
    $ git clone https://gitlab.com/simont3/hcplogs
```

- Install locally, including the dependency:

```
$ python setup.py install
```

2. in a virtual environment

   WINDOWS

   - Create a fresh virtual environment:

```
C:\>\Python35\Tools\scripts\pyvenv.py c:\temp\_venv_hcplogs
```

   - Activate the virtual environment:

```
C:\>cd temp

C:\temp>\temp\_venv_hcplogs\Scripts\activate.bat
(_venv_hcplogs) C:\temp>
```

   - Install **hcplogs**:

```
(_venv_hcplogs) C:\temp>pip install hcplogs
Collecting hcplogs
  Downloading hcplogs-2.0.2.tar.gz
Collecting hcpsdk>=0.9.3.post0 (from hcplogs)
  Downloading hcpsdk-0.9.3.post4.tar.gz
Collecting dnspython3==1.12.0 (from hcpsdk>=0.9.3.post0->hcplogs)
  Downloading dnspython3-1.12.0.zip (226kB)
    100% |################################| 229kB 1.3MB/s
Collecting alabaster>=0.7.1 (from hcpsdk>=0.9.3.post0->hcplogs)
  Downloading alabaster-0.7.6.tar.gz
Installing collected packages: dnspython3, alabaster, hcpsdk, hcplogs
  Running setup.py install for dnspython3
  Running setup.py install for alabaster
  Running setup.py install for hcpsdk
  Running setup.py install for hcplogs
Successfully installed alabaster-0.7.6 dnspython3-1.12.0 hcplogs-2.0.2 hcpsdk-0.9.3.post4
```

   Now you can run **hcplogs** as long as you have the virtual environment activated:

```
C:\temp>hcplogs
A configuration file is not available.
Do you want me to create a template for you (y/n)? y
Creation of template config file "hcplogs_config.ini" was successfull
    You need to edit it to fit your needs!
```

   Linux

   - Create a fresh virtual environment:

```
$ pyvenv _venv_hcplogs
```

   - Activate the virtual environment:

```
$ source _venv_hcplogs/bin/activate
```

   - Install **hcplogs**:

```
(_venv_hcplogs) $ pip install hcplogs
Collecting hcplogs
  Downloading hcplogs-2.0.4.tar.gz
Collecting hcpsdk>=0.9.3.post0 (from hcplogs)
  Downloading hcpsdk-0.9.3.post4.tar.gz
```

```
Collecting dnspython3==1.12.0 (from hcpsdk>=0.9.3.post0->hcplogs)
  Downloading dnspython3-1.12.0.zip (226kB)
    100% |################################| 229kB 7.4MB/s
Collecting alabaster>=0.7.1 (from hcpsdk>=0.9.3.post0->hcplogs)
  Downloading alabaster-0.7.6.tar.gz
Installing collected packages: dnspython3, alabaster, hcpsdk, hcplogs
  Running setup.py install for dnspython3
  Running setup.py install for alabaster
  Running setup.py install for hcpsdk
  Running setup.py install for hcplogs
Successfully installed alabaster-0.7.6 dnspython3-1.12.0 hcplogs-2.0.4 hcpsdk-0.9.3.post4
```

Now you can run **hcplogs** as long as you have the virtual environment activated:

```
(_venv_hcplogs) $ hcplogs
A configuration file is not available.
Do you want me to create a template for you (y/n)? y
Creation of template config file "hcplogs_config.ini" was successfull
        You need to edit it to fit your needs!
```

CHAPTER 2

Usage

HCP access log collector is a command line tool.

```
$ hcplogs --help
usage: hcplogs [-h] [--version] [-i INI]

hcplogs downloads access logs for the REST-based interfaces
(native/HS3/HSwift) from HCP systems. It keeps a timestamp of the last
successfully downloaded logs and excludes everything older than it when
running hcplogs again.

optional arguments:
  -h, --help  show this help message and exit
  --version   show program's version number and exit
  -i INI      path/name of the ini-file containing the configuration (defaults
              to the current directory)
```

A single argument, `-i configfile.ini` is relevant - it points to the configuration file that specifies what the tool shall do.

Tip: If you don't have a configuration file yet, just run the tool without arguments; it will create a template config file for you:

```
$ hcplogs
A configuration file is not available.
Do you want me to create a template for you (y/n)? y
Creation of template config file "hcplogs_config.ini" was successfull
        You need to edit it to fit your needs!
```

Make sure you edit the template file to fit your needs!

A run's output

```
11/08 13:01:30 [INFO  ] Started run (user "sm")
11/08 13:01:30 [INFO  ] log prepare started for admin.hcp72.archivas.com (10/8/2015 - 11/8/2015)
11/08 13:01:30 [INFO  ] log prepare started for admin.hcp73.archivas.com (10/8/2015 - 11/8/2015)
11/08 13:01:31 [INFO  ] status for target hcp72: preparing
11/08 13:01:31 [INFO  ] status for target hcp73: preparing
11/08 13:01:42 [INFO  ] status for target hcp72: preparing
11/08 13:01:42 [INFO  ] status for target hcp73: preparing
11/08 13:01:53 [INFO  ] status for target hcp72: preparing
11/08 13:01:53 [INFO  ] status for target hcp73: preparing
11/08 13:02:04 [INFO  ] status for target hcp72: preparing
11/08 13:02:05 [INFO  ] status for target hcp73: preparing
11/08 13:02:15 [INFO  ] status for target hcp72: preparing
11/08 13:02:16 [INFO  ] status for target hcp73: prepared
11/08 13:02:26 [INFO  ] status for target hcp72: preparing
```

```
11/08 13:02:37 [INFO  ] status for target hcp72: prepared
11/08 13:02:37 [INFO  ] starting download for target hcp72
11/08 13:02:48 [INFO  ] unpacking downloaded logs for target hcp72
11/08 13:02:59 [INFO  ] starting local transfer for node 192.168.0.178 of target hcp72
11/08 13:03:04 [INFO  ] done: transfer for node 192.168.0.178 of admin.hcp72.archivas.com (success/fail: 203/0)
11/08 13:03:16 [INFO  ] starting local transfer for node 192.168.0.179 of target hcp72
11/08 13:03:21 [INFO  ] done: transfer for node 192.168.0.179 of admin.hcp72.archivas.com (success/fail: 209/0)
11/08 13:03:33 [INFO  ] starting local transfer for node 192.168.0.176 of target hcp72
11/08 13:03:39 [INFO  ] done: transfer for node 192.168.0.176 of admin.hcp72.archivas.com (success/fail: 203/0)
11/08 13:03:51 [INFO  ] starting local transfer for node 192.168.0.177 of target hcp72
11/08 13:03:57 [INFO  ] done: transfer for node 192.168.0.177 of admin.hcp72.archivas.com (success/fail: 203/0)
11/08 13:03:57 [INFO  ] starting download for target hcp73
11/08 13:04:13 [INFO  ] unpacking downloaded logs for target hcp73
11/08 13:04:14 [INFO  ] starting local transfer for node 192.168.0.182 of target hcp73
11/08 13:04:14 [INFO  ] done: transfer for node 192.168.0.182 of admin.hcp73.archivas.com (success/fail: 20/0)
11/08 13:04:14 [INFO  ] starting local transfer for node 192.168.0.183 of target hcp73
11/08 13:04:14 [INFO  ] done: transfer for node 192.168.0.183 of admin.hcp73.archivas.com (success/fail: 20/0)
11/08 13:04:15 [INFO  ] starting local transfer for node 192.168.0.180 of target hcp73
11/08 13:04:15 [INFO  ] done: transfer for node 192.168.0.180 of admin.hcp73.archivas.com (success/fail: 21/0)
11/08 13:04:16 [INFO  ] starting local transfer for node 192.168.0.181 of target hcp73
11/08 13:04:16 [INFO  ] done: transfer for node 192.168.0.181 of admin.hcp73.archivas.com (success/fail: 20/0)
11/08 13:04:16 [INFO  ] Finished run (user "sm")
```

CHAPTER 3

Configuration

The tool is configurable using a basic configuration language providing a structure similar to what's found in Microsoft Windows INI files.

The order of the sections, and the order of the items within a section isn't relevant. Nevertheless it's suggested to leave all sections in the config file and simply switch unneeded features to `no` or `off`

> **Warning:** As the configuration file contains user credentials, make sure that it is stored at a secure location with propper permission settings, **not to disclose it to nosy eyes!**
> The config file must be write-able by the tool itself, as it uses it to remember state between its runs.

3.1 The configuration file explained

HCP system(s) to collect from

One or more HCP systems can be configured for log collection by adding more than a single `[target ...]` header. Make sure to replace `...` by an unique identifier for each target section!

- **fqdn** needs to start with `admin.`; using an IP address is *not* supported.
- **user** needs to be a system level user, having the *Admin* and *Monitor* role.
- **folder** is the target folder below the *path* specified in the `[... archive store]` section.
- **last collected** will be used by the tool to remember the last collection date

```
[target hcp72]
fqdn = admin.hcp72.domain.com
user = logmon
password = secret
folder = hcp72
last collected =

[target hcp73]
fqdn = admin.hcp73.domain.com
user = logmon
password = secret
folder = hcp73
last collected =
```

Log type selection

The `[access log]` section allows to select the required logs.

- **access** - user requests

- **admin** - access to the Management Consoles
- **mapi** - requests to the Management API
- **omit empty** - 0-byte files will be skipped if set to `yes`

```
[access logs]
access = yes
admin = yes
mapi = yes
# do not transfer empty logfiles
omit empty = yes
```

Storage locations

The `[local archive store]` section defines where to store the downloaded logs locally.

```
[local archive store]
enable = yes
# path should to be an absolute path
path = /tmp/hcplogs
```

The `[compliant archive store]` section defines where to archive the downloaded logs on an HCP system.

- **path** needs to be full qualified - including http or https and the target folder within the namespace
- **user** needs to have write permission for the namespace
- **retention** can be any valid retention string, as described in the *HCP - Using a Namespace* manual (0 means *deletion allowed*)

```
[compliant archive store]
enable = yes
# path needs to be a full qualified Namespace and folder
path = https://namespace.tenant.hcp72.domain.com/rest/accesslogs
# a user having write permission to the namespace
user = n
password = secret
# retention needs to be a valid HCP retention string
retention = 0
```

Logging

The `[logging ]` section defines where the tool writes its output to. If logging to file is enabled, logfiles are auto-rotated and limited to the given number of backups.

- **status query** tells after how many seconds a status update shall be logged while HCP is preparing the logs for download.
- **debug** enables more detailed and *very* chatty logging.

```
[logging]
log to stdout = yes
log to file = yes
logfile = /tmp/hcplogs/_hcplogs.log
rotateMB = 10
backups = 9
status query = 10
debug = no
```

Temporary storage

This section defines where temporary files will be stored. This is needed for unpacking and decompressing the downloaded file, and the preparation for archiving. The content will be deleted while the tool finishes.

```
[temporary store]
# used for download/de-compression and handling the logs
# will be cleaned up when the tool has finished its work
tempdir = .
```

CHAPTER 4

Archive Structure

The collected log files are stored in a way that makes it somewhat easy to process them later on.

```
[... archive store][path]/
    [target ...][folder]/
        <node IP address>/
            <year>/
                admin_request/
                    YYYYMMDD-HHMM.log.*
                    ...
                http_gateway_request/
                    YYYYMMDD-HHMM.log.*
                    ...
                mapi_gateway_request/
                    YYYYMMDD-HHMM.log.*
                    ...
```

Example:

```
/tmp/hcplogs/
    hcp72/
        192.168.0.176/
            2015/
                admin_request/
                    20151002-1124.log.0
                    ...
                    20151007-0532.log.0
                http_gateway_request/
                    20151002-1124.log.0
                    ...
                    20151007-0532.log.0
                mapi_gateway_request/
                    20151002-1124.log.0
                    ...
                    20151007-0532.log.0
        192.168.0.177/
        192.168.0.178/
        192.168.0.179/
    hcp73/
        192.168.0.180/
            2015/
                admin_request/
                http_gateway_request/
                mapi_gateway_request/
        192.168.0.181/
        192.168.0.182/
        192.168.0.183/
```

CHAPTER 5

Log record structure

5.1 Types of logs

hcplogs collects three different types of access logs (depending on how it is configured):

- *admin_request*

  Log records originated by the use of the System or Tenant Management Consoles

- *http_gateway_log*

  Log records originated by data access through one of the REST-based gateways (native/REST, HSwift, HS3)

- mapi_gateway_request

  Log records originated by requests made to the Management API REST gateway

5.2 Examples

These examples are from the *http_gateway_log*; examples from *admin_request* and *mapi_gateway_request* are not shown here, as they are very similar to the native/REST log records shown.

- native/REST request:

```
192.168.0.110 - filesUser [08/Dec/2015:07:29:05 +0100] \
    "GET /rest/files/testfile/1.1" \
    200 1024 files.aw21 9
```

- HSwift request:

```
192.168.0.220 - swiftuser [07/Dec/2015:21:14:38 +0100] \
    "GET /swift/v1/swifttest/test1?limit=10000 HTTP/1.1" \
    200 530000 test1.swifttest 5750
```

- HS3 request:

```
192.168.0.43 - s3user [08/Dec/2015:07:28:44 +0100] \
    "GET /73-created-01?prefix=&max-keys=1000&delimiter=%2F HTTP/1.1" \
    200 575 xyz@hs3 166
```

5.3 Columns explained

column	content	description
1	192.168.0.110	the clients IP address
2	-	not used, always '-'
3	filesUser	userId used by the client for this request
4	[08/Dec/...	timestamp
5	"GET ...	**the request made by the client** "/rest/..." –> native/REST "/swift/v1/<account>/<container>/..." –> HSwift <account> maps to Tenant (or Tenant-Id if Keystone is used for authentication), <container> maps to Namespace <bucket>/..." –> HS3 <bucket> maps to Namespace
6	200	http returncode
7	1024	size of returned content
8	files.aw21 xyz@hs3	<namespace>.<tenant< (native/REST) or <container>.<account> (HSwift) <tenant> @ HS3 gateway
9	9	HCP internal latency (ms)

Error Handling

Faulty read errors

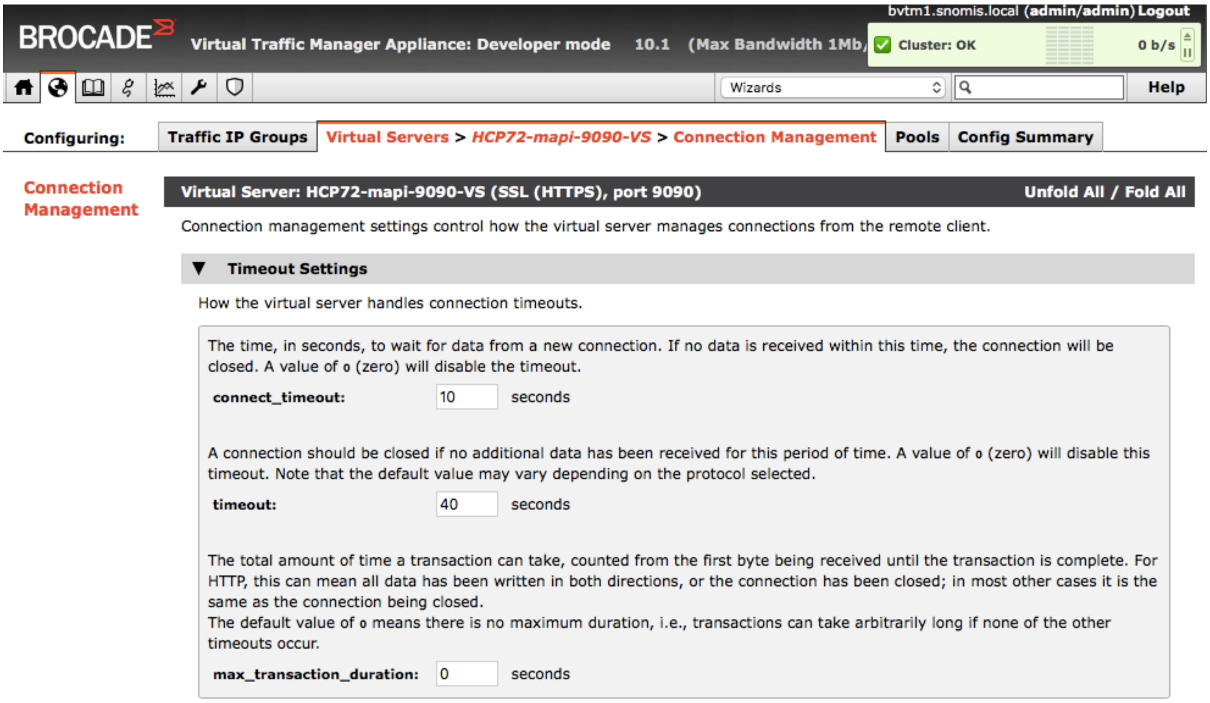
```
02/04 17:36:48 [INFO  ] starting download for target hcp72
02/04 17:37:20 [ERROR ] download for admin.hcp72.archivas.com failed
    hint: faulty read: 'NoneType' object has no attribute 'read'
```

This is a good indicator that the connection between **hcplogs** and HCP has been cut off by a third party while downloading the logs.

Check if your connection to HCP is routed through a *proxy* or *load balancer*. Most likely, there is a setting that cuts off connections that last longer than some seconds, even if in transaction.

As downloading the logs from HCP is not too fast, it is possible that these settings are too short. Look out for the proxy/load balancer configuration that is responisble for traffic to HCPs port 9090 (MAPI).

Load Balancer settings example


BROCADE
Virtual Traffic Manager Appliance: Developer mode 10.1 (Max Bandwidth 1Mb,
bvtm1.snomis.local (admin/admin) Logout
Cluster: OK
0 b/s
Wizards
Help
Configuring:
Traffic IP Groups
Virtual Servers > HCP72-mapi-9090-VS > Connection Management
Pools
Config Summary
Connection Management
Unfold All / Fold All
▼ Timeout Settings

Virtual Server: HCP72-mapi-9090-VS (SSL (HTTPS), port 9090)

Connection management settings control how the virtual server manages connections from the remote client.

How the virtual server handles connection timeouts.

The time, in seconds, to wait for data from a new connection. If no data is received within this time, the connection will be closed. A value of 0 (zero) will disable the timeout.

connect_timeout: 10 seconds

A connection should be closed if no additional data has been received for this period of time. A value of 0 (zero) will disable this timeout. Note that the default value may vary depending on the protocol selected.

timeout: 40 seconds

The total amount of time a transaction can take, counted from the first byte being received until the transaction is complete. For HTTP, this can mean all data has been written in both directions, or the connection has been closed; in most other cases it is the same as the connection being closed.
The default value of 0 means there is no maximum duration, i.e., transactions can take arbitrarily long if none of the other timeouts occur.

max_transaction_duration: 0 seconds

Fig. 6.1: The screenshot shows the relevant panel for **Brocade's VTM**

CHAPTER 7

Release History

2.0.9 2016-02-04

- Added info about error handling in the documenation

2.0.8 2016-01-13

- Changed default settings in the template configuration file
- Fixed a bug that caused hcplogs to fail in case there are Nodes with a backend IP address with a non-3 digit last octet

2.0.7 2015-12-08

- Added a documentation section that describes the log types and their columns

2.0.6 2015-11-08

- After downloads are finished, we now *cancel* the download facility within HCP

2.0.5 2015-10-20

- Documentation fixes

2.0.4 2015-10-15

- Added logo to documentation, trademark etc.

2.0.3 2015-10-14

- Changed documentation / installation

2.0.2 2015-10-11

- Removed unnecessary variables from hcplogs.init.py

2.0.1 2015-10-11

- Changed entrypoint script to allow for *console_ script* configuration, to allow the tool to be directly executable
- Updated documenation to explain source installation via pip and/or gitlab

2.0.0 2015-10-09

- Created the tool from scratch, as the existing version 1 got obsolete when the log download endpoint was invented in HCP 7.2

CHAPTER 8

License / Trademarks

8.1 The MIT License (MIT)

8.2 Trademarks and Copyrights of used material

Hitachi is a registered trademark of Hitachi, Ltd., in the United States and other countries. Hitachi Data Systems is a registered trademark and service mark of Hitachi, Ltd., in the United States and other countries.

Archivas, Hitachi Content Platform, Hitachi Content Platform Anywhere and Hitachi Data Ingestor are registered trademarks of Hitachi Data Systems Corporation.

All other trademarks, service marks, and company names in this document or web site are properties of their respective owners.

The logo used in the documentation is based on the picture found at pixabay.com[4], where it is declared under CC0 Public Domain[5] license.

[4] https://pixabay.com/en/logging-log-truck-hauling-logs-36093/

[5] https://pixabay.com/service/terms/#usage

CHAPTER 9

About

About the Developer

The developer serves for Hitachi Data Systems since 2007, with a main focus on **Hitachi Content Platform** and its family of related products. Being a presales consultant with HDS Germany for more than six years, he actually works for the HCP engineering department as an HCP Technologist for the EMEA region.

Prior to HDS, he served for eight years as a presales manager for a major storage vendor in Germany. Before that, he worked for ten years as a software developer, system programmer, project manager and technical architect for a major German manufacturing company.

In his spare time, he develops tools around HCP that make his own (and hopefully) others life easier.

You can contact him per email at sw@snomis.de