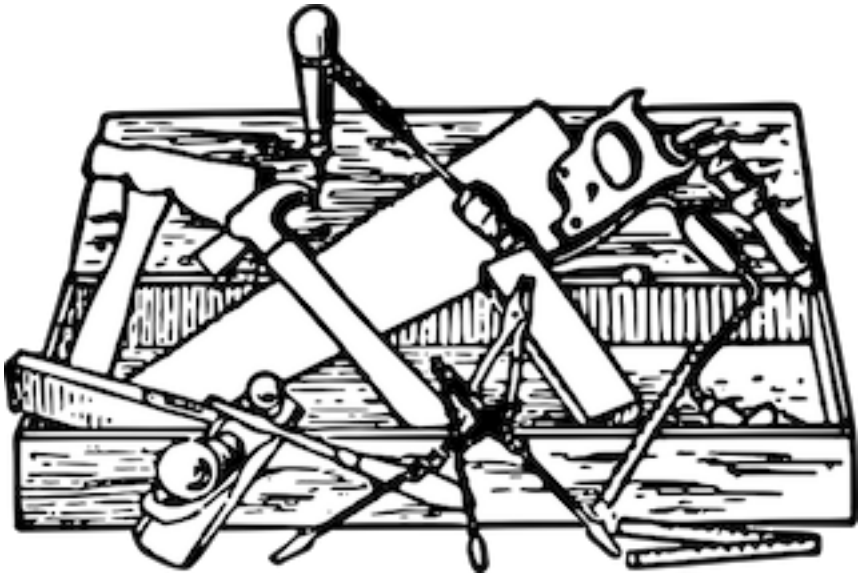

HCP Tools Documentation

Release 1.0.20

Thorsten Simons

May 03, 2022

Contents

HCP Tool (**hcpt**) offers some functions that might be useful when working with Hitachi Content Platform (HCP):

- Calculate the access token needed to access an authenticated namespace or the Management API
- Load HCP with test data
- List the content of a namespace (or parts of it)
- Change the retention of objects within HCP
- Delete objects from HCP, supporting Purge and Privileged Delete

Warning: Some commands may have severe impact on a production HCP:

load - if you load data into a namespace running in ‚compliance‘ mode while setting a retention for the ingested objects, you will not be able to delete these objects before the retention has expired!

retention - you might lock data in longer retention then wanted if you use an incorrect retention string - please refer to ‚Using a Namespace‘ on how to form this string!

unload - you might delete (purge) objects under retention in a namespace in ‚enterprise‘ mode.

CHAPTER 1

Setup

1.1 Pre-requisites

- Hitachi Content Platform (HCP)
- A Tenant-level user account having read/write/delete access to an HCP Namespace

1.2 Dependencies

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

It depends on the hcpsdk[1] to access 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 **hcpt**:

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

     ```
     $ pip install hcpt
     ```

   -or-

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

[1] http://hcpsdk.readthedocs.org/en/latest/
[2] https://www.python.org/downloads/
[3] https://gitlab.com/simont3/hcptool

– by cloning the repository:

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

- 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_hcpt
```

- Activate the virtual environment:

```
C:\>cd temp

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

- Install **hcpt**:

```
(_venv_hcpt) C:\temp>pip install hcpt
Collecting hcpt
  Downloading hcpt-1.0.19.tar.gz
Collecting hcpsdk>=0.9.3.post0 (from hcpt)
  Downloading hcpsdk-0.9.3.post4.tar.gz
Collecting dnspython3==1.12.0 (from hcpsdk>=0.9.3.post0->hcpt)
  Downloading dnspython3-1.12.0.zip (226kB)
    100% |################################| 229kB 1.3MB/s
Collecting alabaster>=0.7.1 (from hcpsdk>=0.9.3.post0->hcpt)
  Downloading alabaster-0.7.6.tar.gz
Installing collected packages: dnspython3, alabaster, hcpsdk, hcpt
  Running setup.py install for dnspython3
  Running setup.py install for alabaster
  Running setup.py install for hcpsdk
  Running setup.py install for hcpt
Successfully installed alabaster-0.7.6 dnspython3-1.12.0 hcpt-1.0.19 hcpsdk-0.9.
↪3.post4
```

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

```
C:\temp>hcpt
blablabla
```

Linux

- Create a fresh virtual environment:

```
$ pyvenv _venv_hcpt
```

- Activate the virtual environment:

```
$ source _venv_hcpt/bin/activate
```

- Install **hcpt**:

```
(_venv_hcpt) $ pip install hcpt
Collecting hcpt
  Downloading hcpt-1.0.19.tar.gz
```

(continues on next page)

(continued from previous page)

```
Collecting hcpsdk>=0.9.3.post0 (from hcpt)
  Downloading hcpsdk-0.9.3.post4.tar.gz
Collecting dnspython3==1.12.0 (from hcpsdk>=0.9.3.post0->hcpt)
  Downloading dnspython3-1.12.0.zip (226kB)
    100% |################################| 229kB 7.4MB/s
Collecting alabaster>=0.7.1 (from hcpsdk>=0.9.3.post0->hcpt)
  Downloading alabaster-0.7.6.tar.gz
Installing collected packages: dnspython3, alabaster, hcpsdk, hcpt
  Running setup.py install for dnspython3
  Running setup.py install for alabaster
  Running setup.py install for hcpsdk
  Running setup.py install for hcpt
Successfully installed alabaster-0.7.6 dnspython3-1.12.0 hcpt-1.0.19 hcpsdk-0.9.
↪3.post4
```

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

```
(_venv_hcpt) $ hcpt
blablabla
```

1.4 Verification

- Make sure that you have access to an HCP system and valid user credentials for an authenticated namespace at hand. The test may also be run against the default namespace, where's no need to have user credentials (simply provide random user / password).
- Get a command prompt
- If ever possible, switch of the local machine's DNS cache:

```
c:> net stop dnscache
```

- Run **hcpt** in test mode:

```
c:> hcpt -i5 --user <user> --password <password> test --cluster \
          ns.tenant.hcp.vm.loc --dir /rest/hcpt_test --file <filename>
          --structure 10 100
```

(if the target namespace has 'versioning' eabled, add –versionedNS to the command above!)

hcpt will...

1. calculate access tokens for both namespace and
2. MAPI access
3. feed file <filename> 100 times in each of 10 subdirectories of `/rest/hcpt_test` within namespace 'ns' in tenant 'tenant' within an HCP with a DNS name of 'hcp.vm.loc'
4. discover a list of all objects and directories, beginning at `/rest/hcpt_test`. This list will be provided as 2 files: hcpt_test.csv (to be loaded into MS Excel) and hcpt_list.<timestamp>.db, a Sqlite3-database that will be used as base for the nexts steps.
5. update the database file with a new retention setting to be processed in the next step.
6. update HCP with the new retention settings for the objects selected in the database in the prior step.
7. delete (purge) all the objects and the directories where they have been stored.

At the end, there will be a logfile (hcp_test.log) showing all the output that has been showed on the screen, an Excel-importable file (hcp_test.csv) containing a list of all objects and directories found in 4. and two Sqlite3 database files, containing the results of 4. and 7.

test run output:

```
02/12 09:25:49 [INFO    ] hcpt test v.0.9.3 (2011-07-07/Sm)
02/12 09:25:49 [INFO    ]     Logfile: hcpt_test.log
02/12 09:25:49 [INFO    ]      Target: https://test.m.hcp73.archivas.com/rest/hcpt_test15
02/12 09:25:49 [INFO    ] DataAccAcnt: n
02/12 09:25:49 [INFO    ]  Ingestfile: readme.txt
02/12 09:25:49 [INFO    ]  Started at: Fri, 12.02., 09:25:49
02/12 09:25:49 [INFO    ] ================================================
02/12 09:25:49 [INFO    ]
02/12 09:25:49 [INFO    ] ------------------------------------------------
02/12 09:25:49 [INFO    ] Test 1: generate a cookie for a DataAccessAccount
02/12 09:25:49 [INFO    ] ------------------------------------------------
02/12 09:25:49 [INFO    ] hcp-ns-auth=bg==:1dc7fed37e11b35093d311ef66928ad9
02/12 09:25:49 [INFO    ]
02/12 09:25:49 [INFO    ] ----------------------------------------
02/12 09:25:49 [INFO    ] Test 2: generate a cookie for MAPI access
02/12 09:25:49 [INFO    ] ----------------------------------------
02/12 09:25:49 [INFO    ] hcp-api-auth=bg==:1dc7fed37e11b35093d311ef66928ad9
02/12 09:25:49 [INFO    ]
02/12 09:25:49 [INFO    ] ------------------------------------
02/12 09:25:49 [INFO    ] Test 3: ingest some objects into HCP
02/12 09:25:49 [INFO    ] ------------------------------------
02/12 09:25:49 [INFO    ] hcpt load v.1.1.0 (2015-02-20/Sm)
02/12 09:25:49 [INFO    ]     Logfile: hcpt_test.log
02/12 09:25:49 [INFO    ] Loginterval: 5 sec.
02/12 09:25:49 [INFO    ]      Target: https://test.m.hcp73.archivas.com/rest/hcpt_test15/
02/12 09:25:49 [INFO    ] DataAccAcnt: n
02/12 09:25:49 [INFO    ]  Ingestfile: readme.txt
02/12 09:25:49 [INFO    ]    Filesize: 1012 Bytes (= 1012.00 Byte)
02/12 09:25:49 [INFO    ]   Retention: 0
02/12 09:25:49 [INFO    ]   Structure: [10, 100] = 1,000 PUTs (= 988.28 Kbyte)
02/12 09:25:49 [INFO    ]     Threads: 30
02/12 09:25:49 [INFO    ]  Started at: Fri, 12.02., 09:25:49
02/12 09:25:49 [INFO    ] ------------------------------------------------
02/12 09:25:49 [INFO    ] MonitorThread started monitoring Worker.Qbox.qsize()
02/12 09:25:54 [INFO    ] files sent: 976, errors: 0, PUTs/sec.: 194.5
02/12 09:25:55 [INFO    ] MonitorThread exits on Worker.Qbox.qsize() == 0
02/12 09:25:55 [INFO    ] ------------------------------------------------
02/12 09:25:55 [INFO    ]  Started at: Fri, 12.02., 09:25:49
02/12 09:25:55 [INFO    ]    Ended at: Fri, 12.02., 09:25:55
02/12 09:25:55 [INFO    ]     Runtime: 5.12 Sekunden
02/12 09:25:55 [INFO    ]     Success: 1000
02/12 09:25:55 [INFO    ]      Errors: 0
02/12 09:25:55 [INFO    ]  Statistics:
02/12 09:25:55 [INFO    ]   Obj./Sec.: 195.2
02/12 09:25:55 [INFO    ]    Transfer: 192.88 Kbyte/sec.
02/12 09:25:55 [INFO    ]       t-sum:   147,216 ms
02/12 09:25:55 [INFO    ]   t-average:       147 ms
02/12 09:25:55 [INFO    ]       t-min:        21 ms
02/12 09:25:55 [INFO    ]       t-max:       359 ms
02/12 09:25:55 [INFO    ]    t-median:       129 ms
02/12 09:25:55 [INFO    ]  t-90%-line:       234 ms
02/12 09:25:55 [INFO    ]
02/12 09:25:55 [INFO    ] ---------------------------------------
02/12 09:25:55 [INFO    ] Test 4: list the newly ingested objects
02/12 09:25:55 [INFO    ] ---------------------------------------
02/12 09:25:55 [STARTUP ] hcpt list v.1.4.6 (2015-05-20/Sm)
02/12 09:25:55 [STARTUP ]     Logfile: hcpt_test.log
02/12 09:25:55 [STARTUP ] Loginterval: 5 sec.
02/12 09:25:55 [STARTUP ]    Database: hcpt_list.20161202092549.db
02/12 09:25:55 [STARTUP ]              (will include deleted objects)
```

(continues on next page)

(continued from previous page)

```
02/12 09:25:55 [STARTUP ]      Target: https://test.m.hcp73.archivas.com/rest/hcpt_test15
02/12 09:25:55 [STARTUP ] DataAccAcnt: n
02/12 09:25:55 [STARTUP ]     Threads: 30
02/12 09:25:55 [STARTUP ]     FindQue: -1
02/12 09:25:55 [STARTUP ] dbWriterQue: -1
02/12 09:25:55 [STARTUP ]   Verbosity: 1
02/12 09:25:55 [STARTUP ]  Started at: Fri, 12.02., 09:25:55
02/12 09:25:55 [STARTUP ] ------------------------------------------------------------------
↪----------
02/12 09:25:55 [INFO    ] *** monitor started ***
02/12 09:25:55 [INFO    ] *** dbWriter started ***
02/12 09:25:56 [INFO    ] *** Discovery finished after 00:00:01.2 ***
02/12 09:25:56 [INFO    ] *** dbWriter finished after 00:00:01.3***
02/12 09:26:00 [INFO    ] Dirs:11; Objs:1,000; Err/Warn:0/0; dbWrites:1,011 [QF:0; Qdb:0]
02/12 09:26:00 [INFO    ] *** monitor exits ***
02/12 09:26:00 [STOPDOWN] ------------------------------------------------------------------
↪----------
02/12 09:26:00 [STOPDOWN]  Started at: Fri, 12.02., 09:25:55
02/12 09:26:00 [STOPDOWN]    Ended at: Fri, 12.02., 09:26:00
02/12 09:26:00 [STOPDOWN]     Runtime: 00:00:05.1 h:m:s.ms
02/12 09:26:00 [STOPDOWN]       Found: 11 Directories, 1,000 Objects
02/12 09:26:00 [STOPDOWN]    Warnings: 0
02/12 09:26:00 [STOPDOWN]      Errors: 0
02/12 09:26:00 [INFO    ]
02/12 09:26:00 [INFO    ] ------------------------------------------------------
02/12 09:26:00 [INFO    ] Test 5: prepare the database file for retention changes
02/12 09:26:00 [INFO    ] ------------------------------------------------------
02/12 09:26:00 [INFO    ] *** database update begins ***
02/12 09:26:00 [INFO    ] *** changing all object's retention to 'N+1s' ***
02/12 09:26:00 [INFO    ] *** database update finished ***
02/12 09:26:00 [INFO    ]
02/12 09:26:00 [INFO    ] ------------------------------------------------------------
02/12 09:26:00 [INFO    ] Test 6: change the retention of the newly ingested objects
02/12 09:26:00 [INFO    ] ------------------------------------------------------------
02/12 09:26:00 [STARTUP ] hcpt retention v.0.9.5 (2011-07-02/Sm)
02/12 09:26:00 [STARTUP ]     Logfile: hcpt_test.log
02/12 09:26:00 [STARTUP ] Loginterval: 5 sec.
02/12 09:26:00 [STARTUP ]    Database: hcpt_list.20161202092549.db (1.4.6/2015-05-20/Sm)
02/12 09:26:00 [STARTUP ]      Target: https://test.m.hcp73.archivas.com
02/12 09:26:00 [STARTUP ] DataAccAcnt: n
02/12 09:26:00 [STARTUP ]     Threads: 30
02/12 09:26:00 [STARTUP ]   Verbosity: 1
02/12 09:26:00 [STARTUP ]  Started at: Fri, 12.02., 09:26:00
02/12 09:26:00 [STARTUP ] ------------------------------------------------------------------
↪----------
02/12 09:26:00 [INFO    ] *** monitor started ***
02/12 09:26:00 [INFO    ] *** Changing retentions started ***
02/12 09:26:05 [INFO    ] Changed: 834, Errors=0, Warnings=0
02/12 09:26:07 [INFO    ] *** Changing retentions finished after 00:00:07.45 ***
02/12 09:26:10 [INFO    ] Changed: 1,000, Errors=0, Warnings=0
02/12 09:26:10 [INFO    ] *** monitor exits ***
02/12 09:26:10 [STOPDOWN] ------------------------------------------------------------------
↪----------
02/12 09:26:10 [STOPDOWN]  Started at: Fri, 12.02., 09:26:00
02/12 09:26:10 [STOPDOWN]    Ended at: Fri, 12.02., 09:26:10
02/12 09:26:10 [STOPDOWN]     Runtime: 00:00:10.1 h:m:s.ms
02/12 09:26:10 [STOPDOWN]     Changed: 1000
02/12 09:26:10 [STOPDOWN]    Warnings: 0
02/12 09:26:10 [STOPDOWN]      Errors: 0
02/12 09:26:10 [INFO    ]
02/12 09:26:10 [INFO    ] ----------------------------------------
```

(continues on next page)

(continued from previous page)

```
02/12 09:26:10 [INFO    ] Test 7: delete the newly ingested objects
02/12 09:26:10 [INFO    ] ----------------------------------------
02/12 09:26:10 [STARTUP ] hcpt unload v.1.1.12 (2015-02-27/Sm)
02/12 09:26:10 [STARTUP ]     Logfile: hcpt_test.log
02/12 09:26:10 [STARTUP ] Loginterval: 5 sec.
02/12 09:26:10 [STARTUP ]    Database: hcpt_unload.20161202092549.db
02/12 09:26:10 [STARTUP ]             (will be kept when finished)
02/12 09:26:10 [STARTUP ]      Target: https://test.m.hcp73.archivas.com/rest/hcpt_test15
02/12 09:26:10 [STARTUP ] DataAccAcnt: n
02/12 09:26:10 [STARTUP ]     Threads: 30
02/12 09:26:10 [STARTUP ]   QueueSize: -1
02/12 09:26:10 [STARTUP ]       Purge: yes
02/12 09:26:10 [STARTUP ]  Privileged: yes (Reason: 'hcpthcptcmds')
02/12 09:26:10 [STARTUP ]   Verbosity: 1
02/12 09:26:10 [STARTUP ]  DeleteMode: REQUESTED - Objects and Directories
02/12 09:26:10 [STARTUP ]  Started at: Fri, 12.02., 09:26:10
02/12 09:26:10 [STARTUP ] ------------------------------------------------------------------------
↪----------
02/12 09:26:10 [WARNING ] MonitorThread started - monitoring Worker Threads
02/12 09:26:10 [INFO    ] *** dbWriter started ***
02/12 09:26:10 [WARNING ] ***Discovery finished***
02/12 09:26:15 [INFO    ] Dirs: 11/0, Objects: 1000/605 (found/deleted), Errors: 0 [Q:0/365/0␣
↪(find/del/dbW)]
02/12 09:26:17 [WARNING ] ***Object deletion finished***
02/12 09:26:18 [INFO    ] Thread-1133: http (DELETE) error: 403:/rest/hcpt_test15 (re-
↪scheduled for later deletion)
02/12 09:26:18 [WARNING ] ***Directory deletion finished***
02/12 09:26:20 [INFO    ] Dirs: 11/11, Objects: 1000/1000 (found/deleted), Errors: 0 [Q:0/0/0␣
↪(find/del/dbW)]
02/12 09:26:20 [WARNING ] MonitorThread exits - all Worker Threads ended
02/12 09:26:20 [STOPDOWN] ------------------------------------------------------------------------
↪----------
02/12 09:26:20 [STOPDOWN]  Started at: Fri, 12.02., 09:26:10
02/12 09:26:20 [STOPDOWN]    Ended at: Fri, 12.02., 09:26:20
02/12 09:26:20 [STOPDOWN]     Runtime: 10.01 Sekunden
02/12 09:26:20 [STOPDOWN]       Found: 11 Directories, 1000 Objects
02/12 09:26:20 [STOPDOWN]     Deleted: 11 Directories, 1000 Objects
02/12 09:26:20 [STOPDOWN]      Errors: 0
02/12 09:26:20 [INFO    ]
02/12 09:26:20 [INFO    ] ==================================================
02/12 09:26:20 [INFO    ]  Started at: Fri, 12.02., 09:25:49
02/12 09:26:20 [INFO    ]    Ended at: Fri, 12.02., 09:26:20
02/12 09:26:20 [INFO    ]     Runtime: 30.19 Sekunden
```

CHAPTER 2

Usage

hcpt is a command line tool.

2.1 Syntax

```
$ hcpt [common arguments] subcommand [subcommand arguments]


usage: hcpt [-h] [--version] -u USER [-p PASSWORD] [-l LOGFILE]
            [-i seconds] [-t '# of threads'] [--nossl] [-v] [--gc t1.t2.t3]
            {cookie,load,list,retention,test,unload} ...
```

hcpt serves as a switchboard for the HCP Tools.
Required option is the subcommand to be used. Please note that arguments described here are additional to the subcommand's arguments.

Positional arguments:

```
{cookie,load,list,retention,test,unload}
    cookie              calculate HCP access token
    load                load bulk testdata into HCP
    list                list HCP content
    retention           change retention setting for selected objects within
                        HCP (see specific instructions)
    test                test-run all the subcommands
    unload              delete content from HCP
```

Optional arguments:

```
-h, --help            show this help message and exit
--version             show program's version number and exit
-u USER, --user USER  data access acount
-p PASSWORD, --password PASSWORD
                      password (will require manual input if not given)
```

(continues on next page)

(continued from previous page)

```
-l LOGFILE, --logfile LOGFILE
                      logfile (defaults to 'hcpt.py_subcmd.log')
-i seconds, --loginterval seconds
                      logging interval (defaults to 10 sec.)
-t '# of threads', --threads '# of threads'
                      no. of parallel threads (defaults to 30)
--nossl               use http instead of https
-v                    verbosity (-v = INFO, -vv = DEBUG, -vvv = garbage
                      collection statistics)
--gc t1,t2,t3         garbage collection thresholds (defaults to
                      '700,10,10'- see 'http://docs.python.org/py3k/
                      library/gc.html#gc.set_threshold')
```

2.2 Subcommands

2.2.1 cookie

Calculate the HCP access token required for http-requests.

```
usage: hcpt.py cookie [-h] [--version] {daac,mapi}
```

Calculate the HCP access token to be used in http-requests.

Positional arguments:

```
{daac,mapi}  account type (DataAccessAccount or MAPI)
```

Optional arguments:

```
-h, --help   show this help message and exit
--version    show subfunctions version and exit
```

2.2.2 load

Perform bulk data ingestion into HCP for testing (!) purposes.

load *does already use the* hcpsdk[4] *to make proper use of the resources of HCP, while maintaining perstistent http(s) sessions. Windows error 10048 doesn't apply here.*

```
usage: hcpt load [-h] [--version] -c CLUSTER -d directory -f ingestfile
                 [-r retention_string] --structure # [# ...] --reqlogfile
                 REQLOGFILE
```

hcpt load performs bulk data ingestion into HCP for testing purposes. It always uses https (or http if '–nossl' is given) and allows for multi-threaded ingestion.

Optional arguments:

```
-h, --help            show this help message and exit
--version             show subfunctions version and exit
-c CLUSTER, --cluster CLUSTER
                      target namespace (full qualified DNS-name)
-d directory, --dir directory
                      target directory ('/rest/...' or '/fcfs_data/...')
-f ingestfile, --file ingestfile
```

(continues on next page)

[4] http://hcpsdk.readthedocs.org/en/latest/

(continued from previous page)

```
                      file to be ingested
-r retention_string, --retention retention_string
                      retention (requires valid HCP retention string)
--structure # [# ...] directory structure to be build
--reqlogfile REQLOGFILE
                      log time needed per PUT into file
```

Controlled by *–structure [#_of_dirs [#_of_dirs [...]]] #_of_files'*, a directory structure is build and `#_of_files` copies of `ingestfile` will be ingested into each lowest level directory.

Example: `3 3 3` causes three directories to be created below `targetdir` (0000, 0001, 0002), with another three subdirectories (0000, 0001, 0002) in each of them and three copies of `ingestfile` to be written into each of these subdirectories.

Warning: Be cautious, you could use up a lot of capacity in HCP and generate a lot of network trafic while using it...

Example:

```
hcpt --user ns1 --password ns101 -v -i3 load \
     --cluster ns1.matrix.hcp1.vm.local \
     --dir /rest /hcpt_test1 --file c:\hitachi_logo.txt \
     --structure 10 10 1
```

2.2.3 list

Discover all objects in a given subdirectory within an HCP namespace while discovering the directory tree top/down. List the found objects and directories in a MS Excel usable file (*.csv) and in a Sqlite3 database file.

list *does already use the* hcpsdk[5] *to make proper use of the resources of HCP, while maintaining perstistent http(s) sessions. Windows error 10048 doesn't apply here.*

```
usage: hcpt list [-h] [--version] -c CLUSTER -d directory [--all]
                 [-B DATABASE] [--out {db,csv,both}] [--fatDB]
                 [--QF queuesize] [--Qdb queuesize] [--delay milliseconds]
                 [--outfile OUTFILE] [--showThreads]
                 [--pause_after PAUSE_AFTER]
                 [--pause_minutes PAUSE_MINUTES]
```

hcpt list lists all objects in a given subdirectory within an HCP namespace while discovering the directory tree top/down. Don't (!!!) run it against large directory trees on a production server - it may kill the server while eating up all resources...

Optional arguments:

```
-h, --help            show this help message and exit
--version             show subfunctions version and exit
-c CLUSTER, --cluster CLUSTER
                      target namespace (full qualified DNS-name)
-d directory, --dir directory
                      target directory ('/rest/...' or '/fcfs_data/...')
--all                 find deleted objects, too (if versioning is configured
                      for the namespace)
-B DATABASE, --database DATABASE
                      database file (defaults to
```

(continues on next page)

[5] http://hcpsdk.readthedocs.org/en/latest/

(continued from previous page)

```
                      'hcpthcptcmds.<timestamp>.[fat|slim].sqlite3')
--out {db,csv,both}   select the output format
--fatDB, --fat        include all available information in database
--QF queuesize        defines the allowed no. of items in FindQueue
--Qdb queuesize       defines the allowed no. of items in dbWriterQueue
--delay milliseconds  add a delay (pause) in ms between two requests
                      executed against HCP by a single thread
--outfile OUTFILE     filename for the resulting .csv file (defaults to
                      'hcpt_list.csv')
--showThreads         show info about running threads
--pause_after PAUSE_AFTER
                      pause discovery after <amt> files found
--pause_minutes PAUSE_MINUTES
                      pause discovery for <amt> minutes when
                      --pause_after triggers
```

Be aware: when discovering large directory trees, memory usage might become a problem, up to the point where this program might hang or even crash. You should monitor it by using `-v` or even `-vvv`. Best advice is to limit the number of threads (`-t`) to not more than 50 and limit the queues (`--QF` and `--Qdb`) to 10.000 and 20.000 respectively. You might encounter a deadlock situation, where `--QF` will be at max. and no object will be found. In this case, you'll need to unlimit `--QF` and maybe lower the threads. Speeding up the garbage collection by tuning `--gc` might help, too. But take care: this program might grab as many main memory as available, potentially affecting other applications - it's up to you to monitor that! Expect long (and I mean: really long) run times when discovering multi-million object directory trees! If you'd like to work with the database generated by this program, you could use tools provided at http://www.sqlite.org/download.html.

Example:

```
hcpt --user ns1 --password ns101 -v -i3 list \
     --cluster ns1.matrix.hcp1.vm.local \
     --dir /rest/hcpt_test1
```

2.2.4 retention

Change retention setting for selected objects. Takes a database generated by **hcpt list** as input to update the retention setting of the objects listed in the database. After generating the database with **hcpt list**, the field `flist.new_ret` must be updated with the new retention setting for each object (see description below).

```
usage: hcpt retention [-h] [--version] -B DATABASE [--delay DELAY]
```

hcpt retention takes a database generated by `hcpt list`, where the column `flist.new_ret` has been altered with a new retention string (see below). For every object (!) with a value in column `flist.new_ret`, **hcpt retention** tries to change the objects retention within HCP to the given value.

Optional arguments:

```
-h, --help            show this help message and exit
--version             show subfunctions version and exit
-B DATABASE, --database DATABASE
                      database file generated by 'hcp list' and altered as
                      described below
--delay DELAY         add a delay (pause) in ms between two requests
                      executed against HCP by a single thread
```

To alter the database, you can use the SQLite shell, available on Mac OS X, many Linux distributions, of from https://sqlite.org/download.html.

For example, if your database file is called `hcplist.sqlite3` and you want to add 1 year to every object's retention, you can follow these steps prior to running this tool:

```
$ sqlite3 hcplist.sqlite3
sqlite> UPDATE flist SET new_ret='R+1y' WHERE type='file' OR type='object';
sqlite> .quit
```

It is **YOUR** responsibility to specify a valid retention string - **hcpt retention** will not check it for validity!!!

Example:

```
hcpt --user ns1 --password ns101 -v -i3 retention \
     --database hcplist.sqlite3
```

2.2.5 test

Runs all subcommands agains HCP to verify the tools functionality.

```
usage: hcpt test [-h] [--version] -c CLUSTER -d directory -f ingestfile
                 [--versionedNS] [-r retention_string] --structure #
                 [# ...]
```

hcpt test runs all subcommands agains HCP, making sure that the program works.

Optional arguments:

```
-h, --help            show this help message and exit
--version             show subfunctions version and exit
-c CLUSTER, --cluster CLUSTER
                      target namespace (full qualified DNS-name)
-d directory, --dir directory
                      target directory ('/rest/...' or '/fcfs_data/...')
-f ingestfile, --file ingestfile
                      file to be ingested
--versionedNS         set this if the target namespace has versioning
                      enabled
-r retention_string, --retention retention_string
                      retention (defaults to 'N+1s')
--structure # [# ...] directory structure to be build
```

Example:

```
hcpt -i5 --user <user> --password <password> test \
     --cluster  ns.tenant.hcp.vm.loc --dir /rest/hcpt_test \
     --file <filename> --structure 10 100
```

2.2.6 unload

Perform deletion of data within HCP namespaces by discovering a directory tree top/down (alternatively, a list with objects to be deleted can be provided). Will find all directories and objects within that tree and will imediately begin with object deletion right after one has been found. Directory deletion will start down/up when the whole tree has been discovered. It will write a sqlite3 database file with a single record for each directory and object found, containing all the information available for it. This can grow quite large...

```
usage: hcpt unload [-h] [--version] -c CLUSTER -d directory
                   [--infile INFILE] [-B DATABASE] [--fatDB] [--keepDB]
                   [--QF queuesize] [--Qdb queuesize] [--objonly] [--purge]
                   [--privileged REASON] [--YES] [--versionedNS]
```

hcpt unload performs deletion of data within HCP namespaces by discovering a directory tree top/down. Will find all directories and objects within that tree and will imediately begin with object deletion right after one has been found. Directory deletion will start down/up when the whole tree has been discovered. It will write a sqlite3 database file with a single record for each directory and object found, containing all the information available for it. This can grow quite large...

Optional arguments:

```
-h, --help            show this help message and exit
--version             show subfunctions version and exit
-c CLUSTER, --cluster CLUSTER
                      target namespace (full qualified dns-name)
-d directory, --dir directory
                      target directory (/rest/... or /fcfs_data/...)
--infile INFILE       file holding a list of objects to be deleted (full
                      path: '/rest/.../object' or '/fcfs_data/.../object'.
                      If set, '--dir' will be used to determine the type of
                      namespace, only.
-B DATABASE, --database DATABASE
                      database file (defaults to
                      'hcpthcptcmds.<timestamp>.[fat|slim].sqlite3')
--fatDB               include all available information in database
--keepDB              do not delete the database file when finished
--QF queuesize        size of internal queue (defaults to unlimited)
--Qdb queuesize       defines the allowed no. of items in dbWriterQueue
--objonly             do not delete directories
--purge               purge versions (if not set, directory deletion will
                      fail if versioning is enabled)
--privileged REASON   perform privileged delete (requires a 'reason')
--YES                 ...if you really (!) want to delete the found
                      objects/directories (defaults to 'generate a list of
                      objects/directories only')
--versionedNS         set this if the target namespace has versioning
                      enabled
```

Be aware: if you have directories with a huge number (10.000++) of objects, main memory will become excessive used, even more the more threads you use. This could lead to runtime errors - in this case you will need to serialize the processing by limiting the number of threads down to 1 (one) depending on the available main memory. Of course, this will lead to a much longer runtime - monitor the processing by using the commandline switch `-v`.

Example:

```
hcpt --user ns1 --password ns101 -v -i3 unload --cluster \
     ns1.matrix.hcp1.vm.local --dir /rest/hcpt_test1 --YES
```

CHAPTER 3

Error Handling

Windows "10048 - Address already in use" errors

If **hcpt** is used effectively, it may send many requests per second to HCP. This may lead to *10048 - Address already in use* errors. This happens because Windows doesn't release used (and closed) TCP/IP-scockets until a timeout of 240 seconds (default) has elapsed. This adds to the fact, that the nummer of outgoing TCP/IP-ports is limited to less than 4.000. Taken as fact that every socket needs it's own outgoing port, it get's clear that we will run out of free sockets when doing multi-ten requests per second. **hcpt** catches the described error, waits a second and then tries again - but that's just a workaround slowing down the process substantially.

Tip: The **load** and **list** functions are already using the hcpsdk[6]. The fix described below isn't required if using these functions, only!

This can be fixed by:

- Lowering the „TCP/IP socket connection timeout" from 240 seconds (default) to e.g. 30 seconds. A Registry-Key needs to be added as DWORD:

  ```
  HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\TcpTimedWaitDelay =␣
  ↪30
  ```

  In the test environment, there was a gain of 20% in performance; the error disappeared.

- Increasing the number of dynamic available TCP/IP Ports (indirect by increasing the highst available dynamic Port). A Registry-Key needs to be added as DWORD:

  ```
  HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\MaxUserPort = 32768
  ```

  This seems to put more load on the systems – CPU-load is substancially higher and the system feels tenacious.

Please consider:

- Changing these Registry-Keys needs a system restart to activate them!
- Changing theses keys has effect for the whole system - other applications could be affected!

[6] http://hcpsdk.readthedocs.org/en/latest/

These Changes must be done with care - the author is not willing to take responsibility for whatever impact they might have!

See also: http://msdn.microsoft.com/en-us/library/aa560610%28v=bts.20%29.aspx

CHAPTER 4

HowTo

Compare the content of two namespaces (or two directory trees)?

Generate a list of all objects for each of the namespaces (or directory trees):

```
hcpt --user ns1 --password ns101 -v -l hcp1.log -i3 \
     list --cluster ns1.matrix.hcp1.vm.local --dir /rest \
     --database hcp1_db --keepDB --fatDB --nooutfile
hcpt --user ns1 --password ns101 -v -l hcp2.log -i3 \
     list --cluster  ns1.matrix.hcp2.vm.local --dir /rest \
     --database hcp2_db --keepDB --fatDB --nooutfile
```

You'll end up with two files, `hcp1_db` and `hcp2_db`, each containing a SQLite3 database. You'll also have two logfiles, `hcp1.log` and `hcp2.log`, at whose end you'll find the number of objects found.

Now, prepare a command file `cmd.sqlite` with this content:

```
attach database hcp2_db as hcp2;
.header on
.output compare.txt
select min(urlname) from
(Select urlname, size, retentionstring
 from main.flist
 Union all
 Select urlname, size, retentionstring
 from hcp2.flist
) tmp
group by urlname
having count(*) = 1
order by urlname;
```

Run `sqlite3` to find the differences between the namespaces (or directory trees):

```
sqlite3 -init cmd.sqlite hcp1_db
```

You'll end up with a file `compare.txt`, holding the paths/names of those objects available in only one namespace (or directory tree).

CHAPTER 5

Release History

1.0.20 - 2018-03-27

- fixed a bug in unload that caused a http 400 0error when –reason included spaces

1.0.19 - 2017-03-21

- Added pyinstaller build
- Changed distribution model to **pip**, only - shipment of pre-built Windows executables has been discarded.
- Set context to NOVERIFY in **retention** and **unload** to work around a restriction for HTTPS connections invented with Python 3.5

1.0.18 - 2015-05-20

- Fixed hcplist: re-queue an URL if an HcpsdkTimeoutError is raised during read()

1.0.17 - 2015-05-13

- Added more debugging output for *list* when using -vvv

1.0.16 - 2015-04-07

- Fixed hcplist: fixed a bug when discovering the default namespace

1.0.15 - 2015-03-29

- Now build with hcpsdk 0.9.2.21
- Fixed hcplist: improved error handling for http requests

1.0.14 - 2015-03-16

- Now build with hcpsdk 0.9.2.13
- Added in hcplist: –pause_after and –pause_minute to allow hcplist to stop discovery after a number of found folders and objects for some minutes. This is due to some older HCP systems having problems to keep pace with the tool
- Fixed hcplist: better error handling for http requests

1.0.13 - 2015-03-08

- Now build with hcpsdk 0.9.0.6

1.0.12 - 2015-03-01

- Fixed a situation where in 'list'a persistent connection was left in an unusable state due to not reading all received data from the response

1.0.11 - 2015-02-27

- Fixed a bug in 'unload' that lead to an exception being raised when unloading the default namespace
- Merged the 1.0.8 and 1.0.10 branches

1.0.9 - 2015-02-20

- 'list', 'load': now using hcpsdk to connect HCP
- 'list': now db, csv or both are written during the run, no more db dump at the end

1.0.8 - 2012-03-06

- Changed the licensing model to the MTI License, the documentation to the open office format, the setup scripts to allow for installation into Python's site-packages folder

1.0.7 - 2012-02-29

- Fixed a bug in 'unload' that prevented deletion of objects when –purge wasn't given.

1.0.6 - 2012-02-29

- Fixed a bug in dbwriter, crashing thread due to eroneous SQL command when doing a 'unload' with –fatDB enabled

1.0.5 - 2012-02-15

- For 'unload', a switch '–versionedNS' to be set when the target namespace allows versioning.

1.0.4 - 2011-12-01

- Changed in 'list': the behaviour when thread end, to assure that dbWriter can execute all its requests.

1.0.3 - 2011-11-30

- Changed in 'list': csvWriter now used 'UTF-8' encoding

1.0.2 - 2011-08-05

- Fixed another bug keeping the dbWriter from doing the last commit

1.0.1 - 2011-08-05

- Changed in 'list': fixed another bug keeping the program from ending if dbWriter thread dies

1.0.0 - 2011-08-04

- Changed in 'list': stop program when dbWriter thread fails

0.9.9 - 2011-07-07

- Added version of all subCmds are listet when calling 'hcpt –version'
- Minor changes in 'hcptest'

0.9.8 - 2011-07-06

- Fixed a false argument setting for 'unload' in hcpt.py

0.9.7 - 2011-07-05

- Fixed bug in subCmd 'test' - transfer of 'loginterval' to the subCmds
- Epilog of subCmd 'retention's help message
- Added a User Manual ;-)

0.9.6 - 2011-07-04

- Added hcpretention as subcommand into hcpt

- Fixed a bug in _unload.hcpunload.py

0.9.5 - 2011-07-01

- Added a test suite, available as subcommand 'test'

0.9.4 - 2011-07-01

- Added added hcpunload as subCmd 'unload' into hcp

0.9.3 - 2011-06-30

- Minor bug fixes in hcpargs.py and _cookie/hcpcookie.py
- Moved hcpArgs() into hcp.py, skipping hcpargs.py

0.9.2 - 2011-06-29

- Added hcpload as subCmd 'load' into hcpt

0.9.0 - 2011-06-29

- Initial Release - merging most of the HCP tools into one. 1st step is hcpcookie and hcplist

CHAPTER 6

License / Trademarks

6.1 The MIT License (MIT)

6.2 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[7], where it is declared under CC0 Public Domain[8] license.

[7] https://pixabay.com/en/toolbox-tool-box-hammer-tool-box-807845//
[8] https://pixabay.com/service/terms/#usage

CHAPTER 7

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