**DataHub客户端安装和使用方法**

## DataHub客户端安装方法

DataHub客户端有四种安装方法，其中，前三种适用于64位Unix/Linux操作系统，第四种适用于64位Windows操作系统。

### 使用脚本安装

支持64位DEBIAN，UBUNTU，FREEBSD，CentOS操作系统。

1、安装：

打开终端，输入：

curl -sSL https://hub.dataos.io/install.sh | sh -s xxxxxxxxxxxxxxxxxxxxxxx

安装过程约持续1分钟，完成安装后DataHub Daemon服务会自动启动。

2、运行Client

执行命令：datahub --help 查看可执行的命令。

[https://hub.dataos.io/clientDownload.html](http://hub.dataos.io/clientDownload.html)，查看当前用户的token。

3、手动停止DataHub Daemon服务:

sudo datahub stop

4、手动启动DataHub Daemon服务:

sudo datahub --daemon --token xxxxxxxxxxxxxxxxxxxxxxxxxxx

注：xxxxxxxxxxxxxxxxxxxxxxxxxxx是用户个人唯一的口令，每个用户不同。在DataHub网页上登陆后，访问安装Client客户端页面，即可看到个人口令。

### 使用源码安装

在安装GO(go1.4以上版本)语言和设置完GOPATH环境变量之后：

1. 下载源码

go get github.com/asiainfoLDP/datahub

编译安装：

在$GOPATH/github.com/asiainfoLDP/datahub内执行go install

1. 启动DataHub Daemon服务

sudo $GOPATH/bin/datahub --daemon --token xxxxxxxxxxxxxxxxxxxxxxxx

1. 停止DataHub Daemon服务

sudo $GOPATH/bin/datahub stop

注：xxxxxxxxxxxxxxxxxxxxxxxxxxx是用户个人唯一的口令，每个用户不同。在DataHub网页上登陆后，访问安装Client客户端页面，即可看到个人口令。

### 使用docker镜像

安装条件：已安装GO(go1.4以上版本)语言和设置完GOPATH环境变量。

1、构建docker镜像：

git clone https://github.com/asiainfoLDP/datahub

cd datahub

docker build -t datahub .

2、启动Docker镜像:

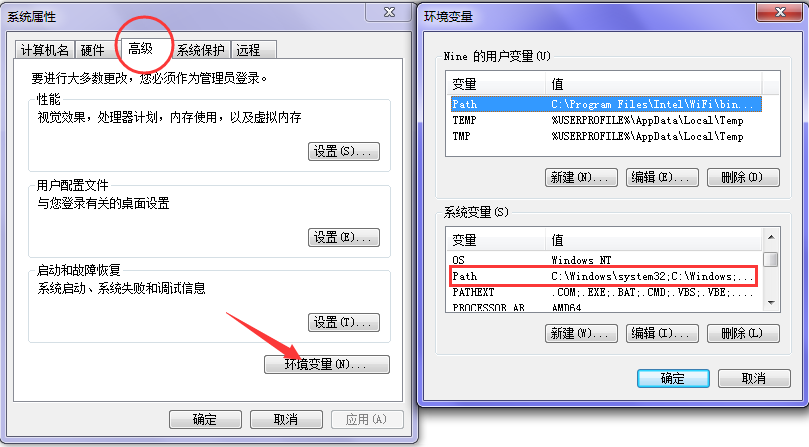
docker run -d -e "DAEMON\_TOKEN= xxxxxxxxxxxxxxxxxxxxxxxxxxx " datahub

注：xxxxxxxxxxxxxxxxxxxxxxxxxxx是用户个人唯一的口令，每个用户不同。在DataHub网页上登陆后，访问安装Client客户端页面，即可看到个人口令。

### Windows系统安装

安装条件：Windows 64位操作系统

1. 下载Datahub工具datahub.exe，放到本地目录
2. 配置环境变量，在Path后加入“datahub.exe”所属路径，分号分隔。例如datahub.exe在D:\Datahub目录下，则在Path中增加D:\Datahub。



1. 启动cmd，输入start datahub --daemon --token xxxxxxxxxxxxxxxxxxxxxxxxxxx

注：xxxxxxxxxxxxxxxxxxxxxxxxxxx是用户个人唯一的口令，每个用户不同。在DataHub网页上登陆后，访问安装Client客户端页面，即可看到个人口令。

### 作为数据提供方的网络策略要求

1、需提供ENTRYPOINT，同时修改防火墙，使得主机的35800端口可以被外部访问。设置entrypoint方法：sudo $GOPATH/bin/datahub ep https://example.com:35800。

2、在使用docker镜像时，需要保证 Docker镜像的35800映射到的主机端口可以被外部访问。

例如:

docker run -d -e "DAEMON\_TOKEN= xxxxxxxxxxxxxxxxxxxxxxxxxxx " –e

"DAEMON\_ENTRYPOINT=https://example.com:35800" -p 35800:35800 datahub

注：xxxxxxxxxxxxxxxxxxxxxxxxxxx是用户个人唯一的口令，每个用户不同。在DataHub网页上登陆后，访问安装Client客户端页面，即可看到个人口令。

## DataHub Client命令介绍

DataHub Client是DataHub的命令行客户端，用来执行DataHub相关命令。

1、常用相关命令：

dp：datapool管理

repo：repository管理

subs：subscrption管理

login：登录到hub.dataos.io

pull：下载数据

pub：发布数据

job：显示任务列表

ep： 设置Entrypoint

logout：登出

help：帮助命令

2、常用专业术语解释：

datapool（dp）：数据数据池

repository（repo）：数据仓库区

subscription（subs）：我的订购

location: datapool下级子目录

jobid：任务id

ep：entrypoint管理

comment：DataItem或者Tag描述

注：

1、命令在没有错误发生时，会记录到日志中（/var/log/datahub.log）。错误信息会打印到终端。

2、日志级别分为[TRACE] [DEBUG] [INFO] [WARNNING] [ERROR] [FATAL]。

3、参数支持全名和简称两种形式，例如--type等同于-t。详情见命令帮助。

4、参数赋值支持空格和等号两种形式，例如--type=file等同于--type file。

### DataPool相关命令

如果需要pull数据，则必须创建相应的datapool。

#### 列出所有数据池

列出所有数据池（datapool）

命令：

datahub dp

举例：

输入：

datahub dp

输出：

DATAPOOL TYPE

--------------------

dp1 file

dp2 file

dphdfs hdfs

dps3 s3

#### 列出数据池详情

指定一个datapool，列出此datapool详情。

datahub dp $DPNAME

举例：

输入：

datahub dp dp1

输出：

$ DATAPOOL:dp1 file /var/lib/datahub/dp1

----------------------------------------------------------------------------------------------------------------------

repo1/item1:tag1 2015-10-23 03:57:42 pub repo1\_item1 tag1.txt Size:232.00KB

repo1/item1:tag2 2015-10-23 03:59:49 pub repo1\_item1 tag2

repo1/item2:jinrong-40 2015-10-23 04:01:22 pull item2location jinrong\_40.txt 金融信息

cmcc/beijing:jiangsu-lac-ci 2015-11-19 10:57:21 pull cmcc\_beijing jiangsu-lac-ci.txt 位置区编码

$

说明：cmcc\_beijing为dataitem beijing在datapool dp1中的位置， jiangsu-lac-ci.txt为tag存储到dp1中的文件名，“位置区编码”为详细信息

#### 创建数据池

新建一个datapool。

datahub dp create $DPNAME [[file://][ABSOLUTE PATH]] | [[s3://][BUCKET]] | [[hdfs://][USERNAME:PASSWORD@HOST:PORT]]

其中：[file://][s3://][hdfs://]为datapool类型；[absolute path]为datapool保存路径（且保存类型必须设置）

举例：

1、在命令中，datapool类型与datapool保存路径均设置：

输入：

datahub dp create dp123 file:///var/lib/datahub/dp123

输出：

DataHub : Datapool create success,Name:dp123 Type:file Path: /var/lib/datahub/dp123

2、在命令中，只设置保存路径:

此时默认为file，且路径为/var/lib/datahub/dp123。

输入：

datahub dp create dp123 /var/lib/datahub/dp123

输出：

DataHub : Datapool create success,Name:dp123 Type:file Path: /var/lib/datahub/dp123

3、在命令中，文件类型和保存路径均不设置：

此时默认为file，且路径默认为/var/lib/datahub/dp123。

输入：

datahub dp create dp123

输出：

DataHub : Datapool create success,Name:dp123 Type:file Path:/var/lib/datahub/dp123

4.创建datapool类型[s3://]

$ datahub dp create s3dp s3://mybucket

DataHub : s3dp already exists, please change another name.

$

说明：mybucket是s3上已存在的bucket。另外，需要在启动daemon的系统中设置环境变量：AWS\_SECRET\_ACCESS\_KEY， AWS\_ACCESS\_KEY\_ID， AWS\_REGION。

5. 创建datapool类型[hdfs://]

$ datahub dp create hdfsdp hdfs://user123:admin123@x.x.x.x:9000

说明：“hdfs://”后需要接hdfs的连接串。

#### 删除数据池

删除数据池不会删除目标数据池已保存的数据。该datapool有发布的数据项时，不能被删除。删除是在sqlite中标记状态，不真实删除。

datahub dp rm $DPNAME

举例：

输入：

datahub dp rm testdp

输出：

DataHub : Datapool testdp with type:file removed successfully!

### login命令

login命令支持被动调用，用于datahub client与datahub server交互时作认证。并将认证信息保存到环境变量，免去后续指令重复输入认证信息。

datahub login

举例：

输入：

datahub login

输出：

login as: datahub

password: \*\*\*\*\*\*\*

DataHub : Login success.

### logout命令

logout命令支持被动调用，用于datahub client与datahub server交互时作认证。并将认证信息保存到环境变量，免去后续指令重复输入认证信息。

datahub logout

举例：

输入：

datahub logout

输出：

DataHub : Logout success.

### subs相关命令

列出所有已订阅项。

datahub subs

举例：

输入：

datahub subs

输出：

REPOSITORY/DATAITEM TYPE

-------------------------------

cmcc/beijing file

Internet\_states/Film\_and\_television file

#### 列出用户在某个Repository下已订阅的DataItem

datahub subs $REPONAME

举例：

输入：

datahub subs cmcc

输出：

REPOSITORY/DATAITEM TYPE

------------------------------

cmcc/Beijing file

cmcc/Tianjin file

cmcc/Shanghai file

#### 列出已订购DataItem详情

datahub subs $REPO/$DATAITEM

举例：

输入：

datahub subs cmcc/Beijing

输出：

REPOSITORY/DATAITEM:TAG UPDATETIME COMMENT

----------------------------------------------

cmcc/beijing:chaoyang 2015-12-12 07:16:56｜21天前 600M

cmcc/beijing:daxing 2015-12-12 07:16:56｜21天前 435M

cmcc/beijing:shunyi 2015-12-12 07:16:56｜21天前 324M

cmcc/beijing:haidian 2015-12-12 07:16:56 ｜21天前 988M

### pull命令

pull命令用来下载您已订阅的DataItem下的Tag到你所指定的Datapool。

##### 4.3.4.1 下载Tag命令

datahub pull $REPO/$DATAITEM:$TAG $DATAPOOL[://$LOCATION] [--destname，-d]

其中参数[--destname, -d ]作用是重命名需要下载的Tag，可以缺省。缺省状态下默认名称为pull的Tag名字。

举例：

1、在缺省命名参数的情况下：

输入：

datahub pull cmcc/beijing:transportation dp1://cmccbj

输出：

DataHub : cmcc／beijing:transportation will be pulled soon and can be found in /var/lib/datahub/dp1/cmccbj

2、在有命名参数的情况下：

输入：

datahub pull cmcc/beijing:transportation dp1://cmccbj --destname “abc”

datahub pull cmcc/beijing:transportation dp1://cmccbj –d “abc”

输出：

DataHub : cmcc／beijing:transportation will be pulled into /var/lib/datahub/dp1/cmccbj/

##### 4.3.4.2 针对repo／DataItem的自动（取消自动）下载

datahub pull $REPO/$DATAITEM:$TAG $DATAPOOL[://$LOCATION] [--automatic，-a]

datahub pull $REPO/$DATAITEM:$TAG $DATAPOOL[://$LOCATION] [--cancel，-c]

其中参数[--automatic，-a] 表示自动下载已订阅的DataItem新增的Tag；参数[--cancel，-c]表示取消自动下载Tag。

举例：

1、repo／DataItem自动下载：

输入：

datahub pull cmcc／beijing dp1 –-automatic

datahub pull cmcc／beijing dp1 –a

输出：

DataHub : cmcc／beijing will be pulled automatically.

2、 repo／DataItem取消自动下载：

输入：

datahub pull repo/DataItem dp1 --cancel

datahub pull repo/DataItem dp1 -c

输出：

DataHub : Cancel the automatical pulling of repo/DataItem successfully.

注：

1、拉取某个DataItem的Tag，即pull一个Tag，需指定$DATAPOOL, 可再指定$DATAPOOL下的子目录$LOCATION。若没有指定，则默认下载到：$DATAPOOL://$REPO\_$DATAITEM. 可选参数。

2、参数支持全名和简称两种形式，例如--type等同于-t。详情见命令帮助。

3、参数赋值支持空格和等号两种形式，例如--type=file等同于--type file

### pub相关命令

pub命令包括：1、发布一个DataItem 2、发布一个Tag。

#### 发布一个DataItem

发布DataItem。须指定DATAPOOL和DATAPOOL下的子路径LOCATION 。

datahub pub $REPOSITORY/$DATAITEM $DATAPOOL://$LOCATION --accesstype=public [private] [--comment, -m]

其中，参数[--accesstype, -t]指定DataItem属性：public, private, 默认private；

参数[--comment, -m]描述DataItem或者Tag。

举例：

1、DataItem属性、描述均设置：

输入：

datahub pub music\_1/migu mydp://dirmigu --accesstype="public" --comment="migu music desc"

输出：

DataHub : Successfully published

2、选择设置DataItem属性，不设置描述：

输入：

datahub pub music\_1/migu mydp://dirmigu --accesstype="public"

输出：

DataHub : Successfully published

3、选择设置DataItem描述，不设置DataItem属性（此时属性默认为private）：

输入：

datahub pub music\_1/migu mydp://dirmigu --comment="migu music desc"

输出；

DataHub : Successfully published

4、DataItem属性、描述均不设置（此时默认属性为private）：

输入：

datahub pub music\_1/migu mydp://dirmigu

输出：

DataHub : Successfully published

注：

1、参数支持全名和简称两种形式，例如--type等同于-t。详情见命令帮助。

2、参数赋值支持空格和等号两种形式，例如--type=file等同于--type file

#### 发布一个Tag

发布Tag。须指定TAGDETAIL , 用来指定Tag对应文件名，该文件必须存在于$DATAPOOL://$LOCATION内。

datahub pub $REPO/$DATAITEM:$Tag $TAGDETAIL --comment=" "

其中，参数[--comment, -m=]用于描述DataItem或者Tag

举例：

1、不设置Tag描述：

输入：

datahub pub music/migu:migu\_user migu\_user\_info.txt

输出：

DataHub:successfully published

2、设置Tag描述：

输入：

datahub pub music/migu:migu\_user migu\_user\_info.txt --comment=“a”

输出：

DatahHub:successfully published

注：

1、参数支持全名和简称两种形式，例如--type等同于-t。详情见命令帮助。

2、参数赋值支持空格和等号两种形式，例如--type=file等同于--type file

### repo命令

repo命令可分三个方面：

#### 查询自己具有写权限的所有Repository

datahub repo

举例：

输入：

datahub repo

输出：

REPOSITORY

--------------------

Location\_information

Internet\_stats

Base\_station\_location

#### 查询自己具有写权限DataItem

datahub repo reponame（要查询的repo名称）

举例：

输入：

datahub repo Location\_information

输出：

REPOSITORY/DATAITEM

--------------------

Location\_information/Cell

Location\_information/Cell01

Location\_information/Cell02

Location\_information/Cell03

Location\_information/Cell04

#### 查询自己创建的Tag

datahub repo reponame（要查询的repo名称）/itemname（要查询的item名称）

举例：

输入：

datahub repo Location\_information/Cell

输出：

REPOSITORY/ITEM:TAG UPDATETIME COMMENT

--------------------

Location\_information/Cell:BR201511250013\_23GCELL\_00 2016-02-23 09:28:41|21天前 Size:393.81 MB

Location\_information/Cell:BR201511250013\_23GCELL\_01 2016-03-10 09:47:39|5天前 Size:393.82 MB

Location\_information/Cell:BR201511250013\_23GCELL\_02 2016-03-11 13:04:11|4天前 Size:393.89 MB

Location\_information/Cell:BR201511250013\_23GCELL\_03 2016-03-14 09:38:03|1天前 Size:393.90 MB

### job命令

job命令分为三个方面：

#### job查看所有任务列表，包括数据下载和发送的任务

datahub job

举例：

输入：

datahub job

输出：

JOBID STATUS DOWN TOTAL PERCENT TAG

---------------------------------------------

72cbddes downloaded 893 893 100.0% Base\_station/Location:location

#### job查看某个任务id对应的信息

datahub job &JOBID

举例：

输入：

datahub job 72cbddes

输出：

JOBID STATUS DOWN TOTAL PERCENT TAG

---------------------------------------------

72cbddes downloaded 893 893 100.0% Base\_station/Location:location

#### job rm删除某个job

datahub job rm &JOBID

### 删除一个Tag

删除自己发布的Tag。

datahub repo rm reponame/DataItemneme:tagname

举例：

输入：

datahub repo rm music/migu:migu\_user

输出：

DataHub : After you delete the Tag, data could not be recovery,Are you sure to delete the current Tag?[Y or N]:

输入：Y

输出：DataHub : OK

输入：N

输出：返回

### 删除一个item

删除自己创建的DataItem。

datahub repo rm reponame/itemname

举例：

输入：

datahub repo rm music/migu

输出：

Datahub : After you delete the DataItem, data could not be recovery, and all tags would be deleted either.Are you sure to delete the current DataItem?[Y or N]:

输入：Y

输出：DataHub : OK

输入：N

输出：返回

### help命令

help提供datahub所有命令的帮助信息。

列出帮助

datahub help [$CMD] [$SUBCMD]

举例：

$datahub help dp

Usage:

datahubdp [DATAPOOL]

List all the datapools or one datapool

Usage of datahubdp create:

datahub dp create DATAPOOL [file://][ABSOLUTE PATH] [[s3://][BUCKET]]

e.g. datahubdp create dptest file:///home/user/test

datahub dp create s3dp s3://mybucket

Create a datapool

Usage of datahub dp rm:

Datahub dp rm DATAPOOL

Remove a datapool

$

### ep命令

若需要向DataHub发布数据，需提供ENTRYPOINT。

datahub ep [http://HOST:PORT]

举例：

输入：

datahub ep http://10.1.235.98:8080

输出：

entrypoint: http://10.1.235.98:8080

DataHub : OK. your entrypoint is: http://10.1.235.98:8080