|  |
| --- |
| Final Sotera Logo w ds 300dpi (2) |
| Datawake Dev Environment Installation Guide 1.2 |
| DARPA-Memex |
| Datawake-Dev-InstallationGuide.docx |
| Brodie McDougald, Michael Frame |
| 2/16/2015 |

Document Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Revision | Reviser | Description |
| 02/11/2015 | 1.0 | Brodie McDougald | Initial draft – Windows Deployment |
| 02/13/2015 | 1.1 | Michael Frame | Added EC2 and Developer Use sections |
| 02/16/2015 | 1.2 | Michael Frame | Added EC2 Digest Authentication |

Table of Contents

[1 Setup for Windows 5](#_Toc411591930)

[1.1 Prerequisites 5](#_Toc411591931)

[1.2 Installation 5](#_Toc411591932)

[1.2.1 Install & Prepare Docker 5](#_Toc411591933)

[1.2.2 Install Datawake 5](#_Toc411591934)

[1.2.3 Check To See That Everything is Up and Running & Save State 6](#_Toc411591935)

[2 Setup for Linux 7](#_Toc411591936)

[2.1 Prerequisites 7](#_Toc411591937)

[2.2 Installation 7](#_Toc411591938)

[2.2.1 Install & Prepare Docker 7](#_Toc411591939)

[2.2.2 Install Datawake 7](#_Toc411591940)

[2.2.3 Check To See That Everything is Up and Running 8](#_Toc411591941)

[3 Setup for Linux on Amazon EC2 9](#_Toc411591942)

[3.1 Prerequisites 9](#_Toc411591943)

[3.2 Installation 9](#_Toc411591944)

[3.2.1 Launch new EC2 instance 9](#_Toc411591945)

[3.2.2 Install & Prepare Docker 9](#_Toc411591946)

[3.2.3 Install Datawake 10](#_Toc411591947)

[3.2.4 Check To See That Everything is Up and Running 10](#_Toc411591948)

[3.2.5 Create an image of the VM for later duplication 11](#_Toc411591949)

[3.2.6 Stopping and Starting EC2 Image 11](#_Toc411591950)

[3.2.7 Starting a previously configured EC2 Image 11](#_Toc411591951)

[4 Developer use of a Linux instance on Amazon EC2 12](#_Toc411591952)

[4.1 Prerequisites 12](#_Toc411591953)

[4.2 Environment 12](#_Toc411591954)

# Setup for Windows

This section describes the setup of the Datawake dev environment under Windows. The environment consists of various system components (Kafka, Datawake, MySQL, Tangelo, etc) configured and running in containers utilizing Docker and Virtual Box.

## Prerequisites

There are a few prerequisites required to run Datawake in a Windows dev environment. These include the following:

* Microsoft Windows
* Boot2Docker (software and instructions available @ <https://docs.docker.com/installation/windows/> )
* A connection to the internet / Sourceforge

## Installation

The following steps need to be performed to get the Windows dev environment installed, configured, and working.

### Install & Prepare Docker

* + Install Boot2Docker via the downloadable version found at <https://docs.docker.com/installation/windows/>
  + Launch the Boot2docker application from the icon on your desktop.
  + When it finishes loading it will be at a ‘*docker@boot2docker:~$*’ prompt
  + Type ‘*git config --global core.autocrlf false*’
  + Create a directory for the GiT repository: ‘*sudo mkdir <whatever you want to call it>*’
  + cd to that directory

### Install Datawake

* + Type ‘*sudo git clone* [*https://github.com/Sotera/Datawake*](https://github.com/Sotera/Datawake)’ (this should clone the Datawake repo to your folder)
  + From the directory you just cloned Datawake into, type ‘*cd Datawake/dev-env*’
  + Create a fig.yml from the fig.yml template file
    - Type ‘*sudo cp fig.yml.template fig.yml* ‘
  + Next, edit the newly created ‘*fig.yml*’ file by typing ‘*sudo vi fig.yml*’.
    - Change the IP addresses listed as 192.168.59.103 to whatever the IP is of your Docker (this is shown when you first start Docker, or you can go to a Windows Command Prompt and type ‘*boot2docker ip’*). To enter edit mode, use the insert key.
    - Locate the “*volumes:*” section of **dwstream** and **datawake** in the fig.yml file. These start with *‘- /Users*’. These lines need to be changed to reflect the correct path of the GiT repo we cloned earlier. Following the steps above, this will likely be *‘- ~/<whatever you called the folder>/Datawake/…*’
    - Save your changes, and exit vi (esc key to exit insert mode, then *‘:wq’*)
  + Install Installation Support Files
    - Type ‘*tce-load -wi python.tcz*’
    - Type ‘*curl https://bootstrap.pypa.io/get-pip.py -o - | sudo python*’
    - Type ‘*curl https://bootstrap.pypa.io/ez\_setup.py -o - | sudo python*’
    - Type ‘*sudo pip install fig*’
  + Bring up the mysql container by typing ‘fig up –d mysql’
  + Insert the tables and mock test data into mysql by typing ‘*./init\_db.sh*’
  + Bring up all of the containers for Datawake by typing ‘*fig up –d*’

### Check To See That Everything is Up and Running & Save State

* + Open your browser and navigate to ‘*http://<your docker ip>/domain/loader*’
    - You should get the domain upload page
    - For the domain name fill in “memex.program”, for description type “*Memex Default domain*”
    - Browse to the default domain file from your local computer’s Datawake/etc directory and click the Submit button
    - Your new domain entry should appear in the table at the bottom of the page.
  + Open Oracle VM VirtualBox Manager (notice your boot2docker-vm is running)
    - Click on Snapshots on the upper right
    - Right Click on the ‘*Current State*’ listed in the snapshot tree, and choose ‘*Take Snapshot*’
    - Right Click on the ‘*boot2docker-vm*’ listed in the left-hand pane, and choose ‘*Close > Save State*’. These steps should save your docker vm in its current state with everything set up. It will also provide a clean starting point if you ever need to revert to a “clean slate”.

# Setup for Linux

## Prerequisites

There are a few prerequisites required to run Datawake in a Linux dev environment. These include the following:

* A Linux Distribution (Ubuntu, CentOS, etc). This guide is from a Ubuntu 14.04 perspective.
* Docker (software and instructions available @ https://docs.docker.com/installation/ )
* A connection to the internet / Sourceforge

## Installation

This installation assumes that you have already installed a base install of Ubuntu or other Linux distribution, and know the credentials that are required (including those of root).

### Install & Prepare Docker

* + Type ‘*ifconfig*’ at the Linux prompt to get your IP address (take note of it for later).
  + Type ‘*sudo apt-get update*’ to update apt-get if necessary.
  + Install the new version of Docker (not Docker.io)
    - Type ‘*curl –sSL* [*https://get.docker.com/ubuntu/*](https://get.docker.com/ubuntu/) *| sudo sh*’
  + Install apparmor (needed by docker to run properly on Ubuntu)
    - Type ‘*sudo apt-get install apparmor apparmor-utils -y*’
  + Install python pip
    - Type ‘*sudo apt-get install python-pip -y*’
  + Create a source folder for the Datawake project
    - From your user home directory (usually ‘~’), type ‘*mkdir src*’
    - Type ‘*cd src*’

### Install Datawake

* + Pull the latest Datawake code to the newly created src directory
    - Type ‘*git clone* [*https://github.com/Sotera/Datawake*](https://github.com/Sotera/Datawake)’
  + Change directory to the Datawake/dev-env folder
    - Type ‘*cd Datawake/dev-env*’
  + Create a fig.yml from the fig.yml template file
    - Type ‘*cp fig.yml.template fig.yml* ‘
  + Next, edit the newly created ‘*fig.yml*’ file by typing ‘*vi fig.yml*’.
    - Change the IP addresses listed as 192.168.59.103 to whatever your Linux IP is (the one you got when you ran ‘*ifconfig*’ earlier). To enter edit mode, use the insert key.
    - Locate the “*volumes:*” section of **dwstream** and **datawake** in the fig.yml file. These start with *‘- /Users*’. These lines need to be changed to reflect the correct path of the GiT repo we cloned earlier. Following the steps above, this will likely be *‘- ~/src/Datawake/…*’
    - Save your changes, and exit vi (esc key to exit insert mode, then *‘:wq’*)

### Check To See That Everything is Up and Running

* + Confirm Docker is running
    - Type ‘*sudo docker ps’*
    - *If it’s not, start it by typing ‘sudo service docker start’*
    - if Docker failed to start, check to see if it still has a ‘*docker.pid*’ in /var/run
      * Type *‘- cat /var/run/docker.pid’*
      * If it exists, delete it by typing ‘*sudo rm /var/run/docker.pid*’
  + Install Fig
    - Type ‘*sudo pip install fig*’
  + Setup your MySQL container in docker
    - Type ‘*sudo fig up -d mysql*’
  + Setup your MySQL database, and create the test user
    - Type ‘*sudo ./init\_db.sh*’
  + Startup Docker and all its containers (using the settings in fig.yml that you created)
    - Type ‘*sudo fig up -d*’
  + Open your browser and navigate to ‘*http://<your ip address>/domain/loader*’
    - You should get the domain upload page
    - For the domain name fill in “*memex.program*”, for description type “*Memex Default domain*”
    - Browse to the default domain file from your local computer’s Datawake/etc directory and click the Submit button
    - Your new domain entry should appear in the table at the bottom of the page.

# Setup for Linux on Amazon EC2

## Prerequisites

There are a few prerequisites required to run Datawake in a Linux dev environment on the EC2. These include the following:

* A Linux Distribution (Ubuntu, CentOS, etc). This guide is from a Ubuntu 14.04 perspective running on a t2.micro (free tier eligible) instance to minimize costs.
* Docker (software and instructions available @ https://docs.docker.com/installation/ )
* A connection to the internet / Sourceforge

## Installation

This installation assumes that you have an Amzon EC2 account and the appropriate permissions to launch new instances.

### Launch new EC2 instance

* + Select the Ubuntu Server 14.04 LTS (HVM), SSD Volume Type AMI (ami-9a562df2) 64 bit image.
  + Select General Purpose, t2.micro (Free tier eligible) Instance Type.
  + Select the default Instance Detail configuration.
  + Set ‘Add Storage’, ‘Size(GiB) to 15
  + Tag the Instance with a meaningful tag such as KEY: Name VALUE: <your desired instance name>
  + Configure the security group by creating a new one with SSH/TCP/22 and HTTP/TCP/80 or selecting an existing group with these inbound rules.
  + Launch the instance.
  + Select an existing key pair or create a new key pair and Launch
  + View your Dashboard to see the running instances. You’ll need both the Internal and Public IP Addresses to proceed.

### Install & Prepare Docker

* + SSH to the Public IP address/Port 22 using your tool of choice (Putty)
  + User: ubuntu
  + Type ‘*ifconfig*’ at the Linux prompt to get your IP address (take note of it for later).
  + Type ‘*sudo apt-get update*’ to update apt-get if necessary.
  + Install the new version of Docker (not Docker.io)
    - Type ‘*curl –sSL* [*https://get.docker.com/ubuntu/*](https://get.docker.com/ubuntu/) *| sudo sh*’
  + Install apparmor (needed by docker to run properly on Ubuntu)
    - Type ‘*sudo apt-get install apparmor apparmor-utils -y*’
  + Install python pip
    - Type ‘*sudo apt-get install python-pip -y*’
  + Make a swap file for the image (avoid out of memory issues in install later)
    - Type ‘sudo fallocate -l 4G /swapfile’ to create the file
    - Type ‘sudo chmod 600 /swapfile’ to enable the file
    - Type ‘sudo mkswap /swapfile’ to set it up
    - Type ‘sudo swapon /swapfile’ to enable the swap space
    - Type ‘sudo vi /etc/fstab’ to edit the file to make it permanent on the image
    - Add a line 2 ‘/swapfile none swap sw 0 0’, save and exit vi
  + Create a source folder for the Datawake project
    - From your user home directory (usually ‘~’), type ‘*sudo mkdir src*’
    - Type ‘*cd src*’

### Install Datawake

* + Pull the latest Datawake code to the newly created src directory
    - Type ‘*git clone* [*https://github.com/Sotera/Datawake*](https://github.com/Sotera/Datawake)’
  + Change directory to the Datawake/dev-env folder
    - Type ‘*cd Datawake/dev-env*’
  + Create a fig.yml from the fig.yml template file
    - Type ‘*cp fig.yml.template fig.yml* ‘
  + Next, edit the newly created ‘*fig.yml*’ file by typing ‘*vi fig.yml*’.
    - Change the IP addresses listed as 192.168.59.103 to whatever your Linux IP is (the one you got when you ran ‘*ifconfig*’ earlier). To enter edit mode, use the insert key.
    - Locate the “*volumes:*” section of **dwstream** and **datawake** in the fig.yml file. These start with *‘- /Users*’. These lines need to be changed to reflect the correct path of the GiT repo we cloned earlier. Following the steps above, this will likely be *‘- ~/src/Datawake/…*’
    - Save your changes, and exit vi (esc key to exit insert mode, then *‘:wq’*)

### Check To See That Everything is Up and Running

* + Confirm Docker is running
    - Type ‘*sudo docker ps’*
    - *If it’s not, start it by typing ‘sudo service docker start’*
    - if Docker failed to start, check to see if it still has a ‘*docker.pid*’ in /var/run
      * Type *‘- cat /var/run/docker.pid’*
      * If it exists, delete it by typing ‘*sudo rm /var/run/docker.pid*’
  + Install Fig
    - Type ‘*sudo pip install fig*’
  + Setup your MySQL container in docker
    - Type ‘*sudo fig up -d mysql*’
  + Setup your MySQL database, and create the test user
    - Type ‘*sudo ./init\_db.sh*’
  + Startup Docker and all its containers (using the settings in fig.yml that you created)
    - Type ‘*sudo fig up -d*’
  + Open your browser and navigate to ‘*http://<your ip address>/domain/loader*’
    - You should get the domain upload page
    - For the domain name fill in “*memex.program*”, for description type “*Memex Default domain*”
    - Browse to the default domain file from your local computer’s Datawake/etc directory and click the Submit button
    - Your new domain entry should appear in the table at the bottom of the page.

### Create an image of the VM for later duplication

* + TODO: document this

### Stopping and Starting EC2 Image

* + If Docker is not properly stopped before shutting down the image it can be left in a bad state. To prevent this, do the following steps before shutting down the VM
    - Type ‘sudo fig stop’
    - Type ‘sudo service docker stop’
    - Stop the vm from the ECs Instances page

TODO: create a script that does this

### Starting a previously configured EC2 Image

* + The EC2 will most likely assign a new IP address to your VM. Therefore you will need to verify your internal and Public IP before using the image. If your MySQL database has not been set to persist your data across sessions you will also need to recreate your test user account
    - Type ‘ifconfig’, compare your IP to the IP address in the Datawake/dev-env/fig.yml file to modify it if necessary
  + Start your MySQL container in docker
    - Type ‘*sudo fig up -d mysql*’
  + Setup your MySQL database, and create the test user
    - Type ‘*sudo ./init\_db.sh*’
  + Startup Docker and all its containers (using the settings in fig.yml that you created)
    - Type ‘*sudo fig up -d*’

TODO: create a script that does this

### To Enable Digest Authentication on an EC2 Image instance

It is fairly straightforward to implement authentication on the Tangelo webserver container by placing an .htaccess file and password file in the root of the site you wish to protect. To enable authentication on the default stack perform the following steps:

* Make sure the docker containers are not running, type ‘sudo fig stop’
* Create an ‘.htaccess’ file in the ~/src/Datawake/server/domain/loader folder. Edit it to have the following lines
  + AuthType digest
  + AuthRealm MemexDatawake
  + AuthPasswordFile ~/src/Datawake/server/domain/loader/memdwpw.txt
* Create the password memdwpw.txt file using tangelo-passwd program
* Startup the docker containers, type ‘sudo fig up –d’
* Browse to http://<your ip>/domain/loader, you should now be prompted to enter the user and password for “MemexDatawake”.
* You can alter or add accounts to the password file using the tangelo-passwd program that is installed with tangelo.

# Developer use of a Linux instance on Amazon EC2

## Prerequisites

There are a few prerequisites required to run Datawake in a Linux dev environment on the EC2 with developer acess. These include the following:

* Git, Github, and appropriate Github permissions to the Datawake project
* Internet access
* A configured SSH key in your OS
* An EC2 account and the appropriate permissions

## Environment

Use of a Datawake instance on the Amazon EC2 greatly simplifies the development experience by allowing the developer to use a remote instance rather than running Boot2Docker or a Linux VM locally on their machine. The developer utilizes the instance by working on and checking in code on their local OS and Git checkout. They then log into an EC2 instance, perform a git pull to the desired branch, and then run ‘sudo fig up –d’ to build and deploy their latest changes.