## QT & C++

**Install QT and pre-requisites:**

WINDOWS: Download and install QT Creator (install in C:\\Qt)

<https://www.qt.io/download>

WINDOWS: Add qmake to environment variables

# locate your qmake.exe

C:\Program Files\Microsoft SQL Server\140\PYTHON\_SERVER\Library\bin

Check that qmake is installed and which version is being used

qmake -v

If successful, you should see the following:

A close up of a logo

Description automatically generated

WINDOWS: Install g++ and add to environment variables

<http://www.cs.utah.edu/~alee/g++/g++old.html>

Assuming the following file structure…

./

├── main.cpp

└── main.h

And assuming terminal application (no qt) Compile using the following:

g++ -c -ansi -pedantic -Wall .\main.cpp -I main.h ; g++ main.o; ./a.exe

**Lesson 1**

How to read UML diagrams:

A close up of a logo

Description automatically generatedA screenshot of a cell phone

Description automatically generated

**Lesson 2**

Use ctrl + r to execute (build + run) your code.

#include <iostream> // Standard c++ library.

using std //Included in iostream.

#include “fact.h” //User-defined header

int main() {

int factArg = 0 ;

cout << "Factorial of: ";

cin >> factArg;

findFact(factArg);

return 0;

}

using std

* Lets us use cout without the prefix std::

**3 - Directories**

~/workspace/dataanalytics/Realtime\_Analytics/modules/salt.utils

├── **salt\_config/**

├── common/ Contains config files (.sls) that apply to the server (url’s, files, directories, install commands)

Configure “states” that the minions

should be in.

├── networking/ Network config

├── pki/ Contains ssh keys/ (.pem/.pub)

├── power/ Login config

├── rap/ Contains template launchers, tools

and.ini files (core, proxy, output)

├── security/ Config for linux security policies

├── testing/ Install script for ??? dependencies

inside container ???

├── users/ Contains user roles & credentials

├── visualization/ Virtualization config

├── master Sets directory root to etc/salt which is

a volume mounted into the salt container

├── minion ???

├── minion\_id minion id

└── top.sls Calls ALL configs (RAP, PROXY, CORE, OUTPUT, EDGE)

├── **salt\_ssh/**

├── ??? ???

├── ??? ???

└── ??? ???

├── **masterless\_minion\_deploy/**

├── ??? ???

├── ??? ???

└── ??? ???

**4 - Salt**

SaltStack is a config management and orchestration tool which to standardize the process of configuring servers. It automates admin and code deployment tasks to reduce errors. It can also be used in devops to push config remotely to servers (CI/CD).

Salt uses a slave-master setup that enables push/pull execution.

**salt\_config/** (Salt config for dataanalytics)

* Run in masterless minion mode
* Config is done statically with yaml files
* Salt states contains a list of managed functions (system functions)

salt roster

* User-made config file.
* Contains list of servers you want to connect to (targets)
* The targets data structure is the list of target systems and attributes on how to connect to these systems

masterless\_minion\_deploy.py

* Pulls a datica minion image from docker hub
* Runs a container using the above image
* Binds folders (salt\_config, salt\_ssh, salt\_common) to /etc directories inside container

**5 – Configure VM to run solution**

Create VM (CentOS) on azure

A screenshot of a cell phone

Description automatically generated

Ssh and log into VM

ssh luyanda@40.127.5.199

Install pre-requisites

(<https://phoenixnap.com/kb/how-to-install-docker-centos-7>)

sudo yum check-update

sudo yum install git

~~# install docker dependencies~~

~~sudo yum install -y yum-utils device-mapper-persistent-data lvm2~~

~~# add docker repo to centOS~~

~~sudo yum-config-manager --add-repo~~ [~~https://download.docker.com/linux/centos/docker-ce.repo~~](https://download.docker.com/linux/centos/docker-ce.repo)

#INSTALL PIP

sudo yum install epel-release

sudo yum install python-pip

~~# INSTALL DOCKER~~

~~sudo yum install docker~~

~~sudo systemctl start docker~~

~~sudo systemctl enable docker~~

~~sudo groupadd docker~~

~~sudo gpasswd -a $USER docker~~

~~sudo service docker restart~~

#INSTALL DOCKER

sudo yum check-update

curl -fsSL https://get.docker.com/ | sh

sudo systemctl start docker

sudo systemctl enable docker

sudo usermod -aG docker $(whoami)

#INSTALL PYTHON LIBRARIES

pip install docker

Close VM and re-enter to allow docker functionality

Download Source code

mkdir ~/workspace

cd ~/workspace

git clone [https://<USERNAME>@bitbucket.org/ddsp/dataanalytics.git](https://%3cUSERNAME%3e@bitbucket.org/ddsp/dataanalytics.git)

git checkout salt\_reorg

Config roster file

cd ~/workspace/dataanalytics/Realtime\_Analytics/modules/salt.utils/salt\_config

vim roster

Add the following config to your roster file

RAP\_PROXY\_TEST:

host: saltybob.southafricanorth.cloudapp.azure.com

user: luyanda

passwd: 1Q2W3e4r5t6y7u!!

sudo: True

tty: True

(NB: roster file should be saved with no file extension. Be sure to check that the tab spacing is correct)

Config common.sls file

cd ~/workspace/dataanalytics/Realtime\_Analytics/modules/salt.utils/salt\_config/rap/

vim common.sls

Change ‘mc-t’ to the first 4 letters of your VM/host. In this case, the VM is called ‘SALT01’ and the first 4 letters are ‘SALT’:

A screen shot of a computer

Description automatically generated

Config nifi-shutdown.sh file

cd ~/workspace/dataanalytics/Realtime\_Analytics/modules/salt.utils/salt\_config/rap/

vim nifi-shutdown.sh

Like the above, change ‘mc-t’ to the first 4 letters of your VM/host

A screen shot of a computer

Description automatically generated

Run the python script. Use two windows to allow for debugging:

Terminal window 1

# build salt container (salt-ssh-temp)

cd ~/workspace/dataanalytics/Realtime\_Analytics/modules/salt.utils

python masterless\_deploy.py

Terminal window 2

# debug “salt-ssh-temp” container

docker logs salt-ssh-temp --follow