**November 4th, 2016**

BigchainDB Error

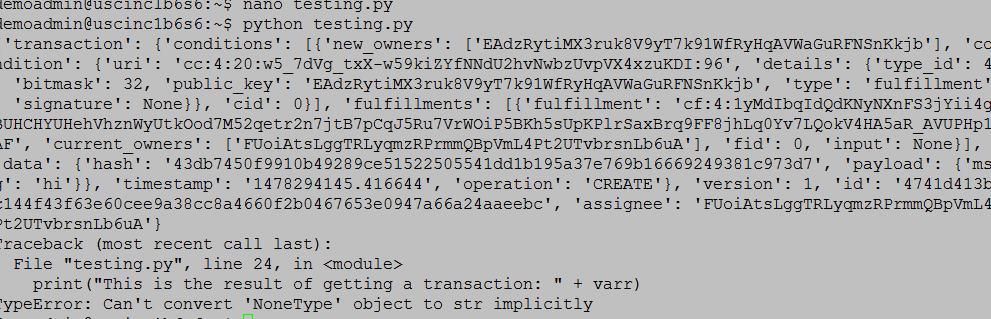
As discussed with Venketa on Tuesday, November 2nd and with Gaurav on Wednesday, November 3rd, the root cause for the driver not working is that the following command is not working:

*chain.get\_transaction(txID)*

This was identified to be the error. Commenting out this line of code in the driver caused the driver to return a response.

Additionally, to verify that this was the error, a python script was made that creates a transaction and retrieves the transaction.

This is the output of the script (for reference, see (\*) to view the code for the script):

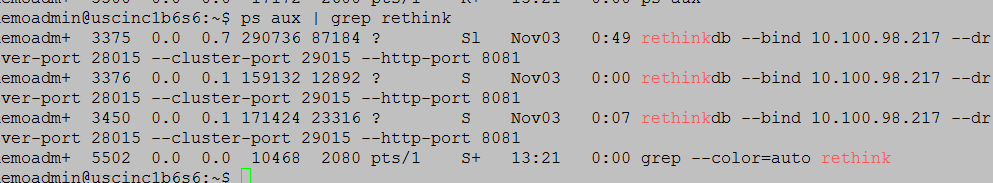


Notice that we can see the output of creation of the transaction. But we can’t get the transaction. It gives us a TypeError. Retrieving the transaction gives us ‘NoneType’. For some reference, note that ‘NoneType’ in python is an object that has no value.

(For reference, see: <http://stackoverflow.com/questions/21095654/what-is-a-nonetype-object>)

However, normally, getting a transaction should either give the transaction, or it should say that the transaction doesn’t exist. It should return a ‘NoneType’ object.

As Gaurav told us, this is not related to rethinkdb. If it were related to rethinkdb, then we would not be able to see the output. Running the linux command “ps grep | aux rethink” shows us that we have rethinkdb running.



Note that killing all processes and running the rethinkdb command once causes all three to start. So we know that having three instances to work does not cause the problem.

Additionally, we made a script to test the functionality of rethinkdb from the web interface, as advised by Venketa. The script runs fine and lets us retrieve and access data from rethinkdb. The script is visible at (\*\*) for reference.

We have also spoken directly to bigchaindb. They cannot help us further without environment configuration information that only infrastructure has access to. We will need additional infrastructure support.

Reference Scripts

(\*) Testing get\_transaction:

from bigchaindb import Bigchain

from bigchaindb import crypto

from bigchaindb import util

import cryptoconditions as cc

#generate a public and private key

priv,pub = crypto.generate\_key\_pair()

#instantiate your chain

b = Bigchain()

digital\_asset\_payload = {"msg": "hi"}

tx = b.create\_transaction(b.me, pub, None, 'CREATE',

payload = digital\_asset\_payload)

tx\_signed = b.sign\_transaction(tx, b.me\_private)

b.write\_transaction(tx\_signed)

print(tx\_signed)

varr = b.get\_transaction(tx\_signed['id'])

print("This is the result of getting a transaction: " + varr)

(\*\*) Testing the rethinkdb web interface:

import rethinkdb as r

#when you first start rethinkdb, the server opens a port for the client drivers (28015 by default)

#open a connection:

r.connect('localhost',28015).repl()

#by default, rethinkdb creates a database test

#let's create a table authors within this database

#print(r.db("test").table\_create("authors").run())

#retrieve all documents from the table authors

#print(r.table('authors').insert([{"msg":"hi"}]).run())

#below was trying to access a table from the rethinkdb data folder:

#print(r.table('b787b578-2ca3-43a3-b4b1-0c18bd199026').run())

#print(list(r.db('rethinkdb').table('server\_status').run()))