ZeroMQ Pi to PC Client/Server

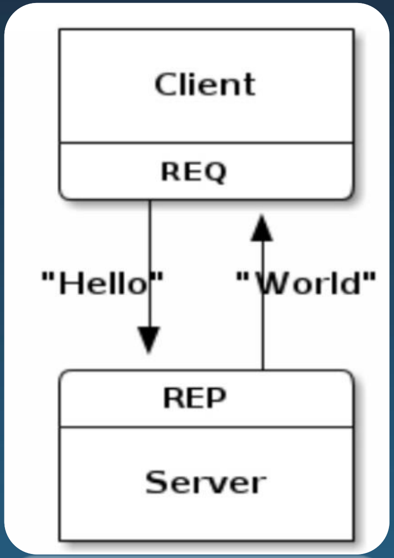
Setup and implementation notes for the “Hello World” program

VOGELECTRIC

ZeroMQ Pi to PC Client/Server

- ZMQ Overview
  - The purpose of this test program was to develop an internet based, cross platform , data communication method to be used in higher level applications. ZeroMQ (also known as MQ or ZMQ) was chosen as the data transport protocol. MQ is a high-performance asynchronous TCP based socket library for messaging with the following features;
    - Many kinds of connection patterns.
    - Multiplatform, multi-language (30+).
    - Fast (8M msg/sec, 30usec latency)
    - Small (20K lines of C++ code)
    - Open source

The test program is based on the published “Hello World” program but it has had some additional functionality added. Both the client and server are written in Python with the Client hosted on a Raspberry PI and the Server hosted on a Windows PC. The client sends "Hello" to the server, which replies with "World". Operation hinges on the server which opens a ➢MQ socket on port 5555, listens for and reads requests from client(s), and replies with "World" to each request. The communication is accomplished using the MQ REQ-REP pattern which is essentially a TCP socket pair is in lockstep.

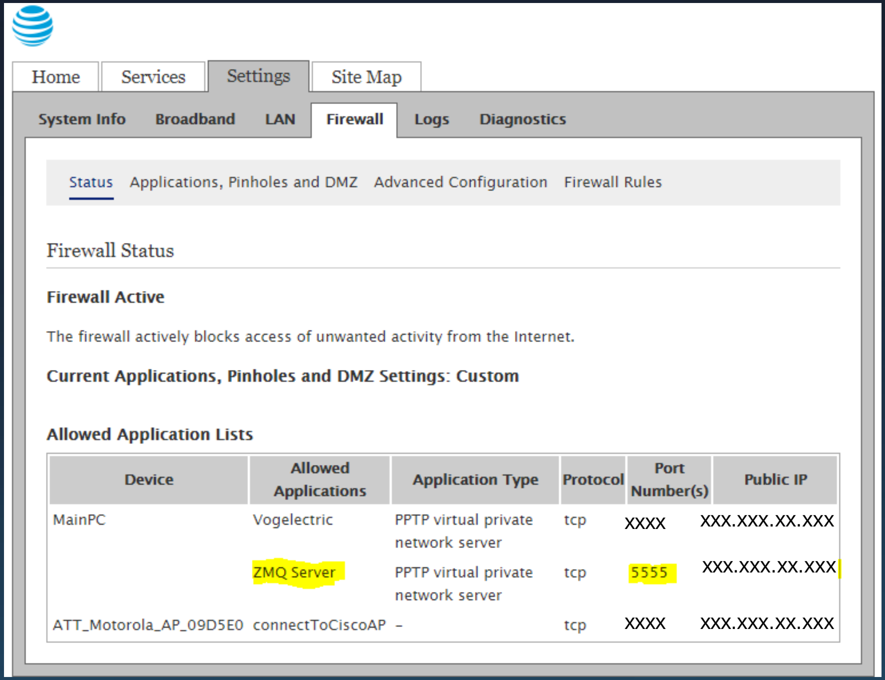


Software Installation

- Software Installation
  - Pyzmq is located at pyzmq github and was installed on both platforms using: "pip install pyzmq" (for python version 2.7)
- Hardware Environments
  - Client
    - Raspberry Pi Client (LINUX)
    - Python 2.7 development environment
    - ZMQ
    - ZMQ Wrapper
  - Server
    - PC (Windows 10)
    - Python 2.7 development environment

Server set-up

- The server on the PC is assigned to port 5555. No conflicts were found although it is a registered port. In order for the server port to be reached port forwarding needed to be enabled on the router.
- The port may also have to be added to the safe list of the antivirus / internet security firewall program i.e. McAfee, Norton, etc.



Firewall Status

Firewall Active

The firewall actively blocks access of unwanted activity from the Internet.

Current Applications, Pinholes and DMZ Settings: Custom

Allowed Application Lists

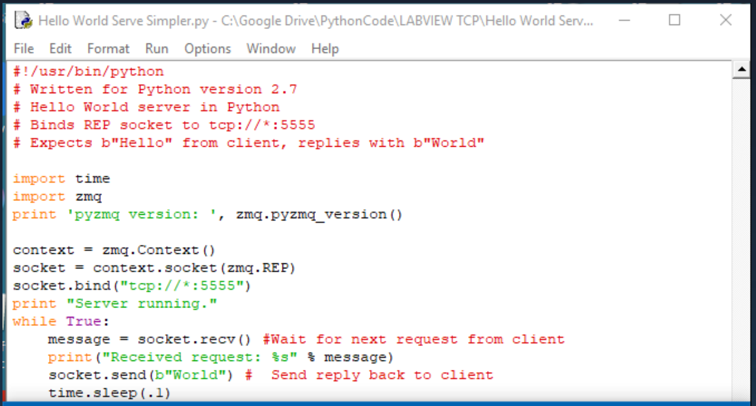
Device	Allowed Applications	Application Type	Protocol	Port Number(s)	Public IP
MainPC	Vogelectric	PPTP virtual private network server	tcp	XXXX	XXX.XXX.XX.XXX
	ZMQ Server	PPTP virtual private network server	tcp	5555	XXX.XXX.XX.XXX
ATT_Motorola_AP_09D5E0	connectToCiscoAP	–	tcp	XXXX	XXX.XXX.XX.XXX

VOGELECTRIC

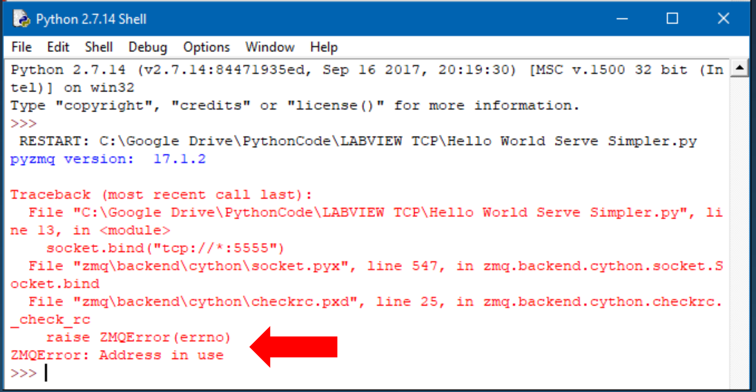
Server Issues

- On the simple server program if the program is terminated either by error or intentionally with Ctrl-C, the port remains bound and trying to run the program again fails with an “Address in use error. There is a workaround to free the port (below) but the better solution was to provide a graceful shutdown by ensuring the port is released.
- Workaround
  1. Open a command prompt window and type in netstat -a -n -o
  2. Find the TCP listing for the target (5555) port [IP address]:[port number] the corresponding [target_PID]).
  3. CTRL+ALT+DELETE and choose "start task manager”.
  4. Click on "Processes" tab
  5. Enable the "PID" column by going to: View > Select Columns > and Check the box for PID.
  6. Find the PID of interest and "END PROCESS"

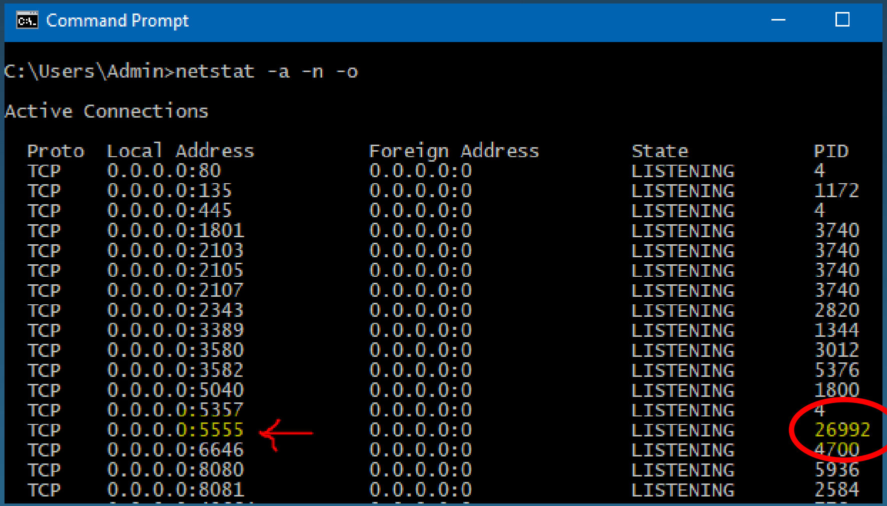
  Now you can rerun the server on [the IP address]:[port number] without a problem.


Hello World Serve Simpler.py - C:\Google Drive\PythonCode\LABVIEW TCP\Hello World Serv...

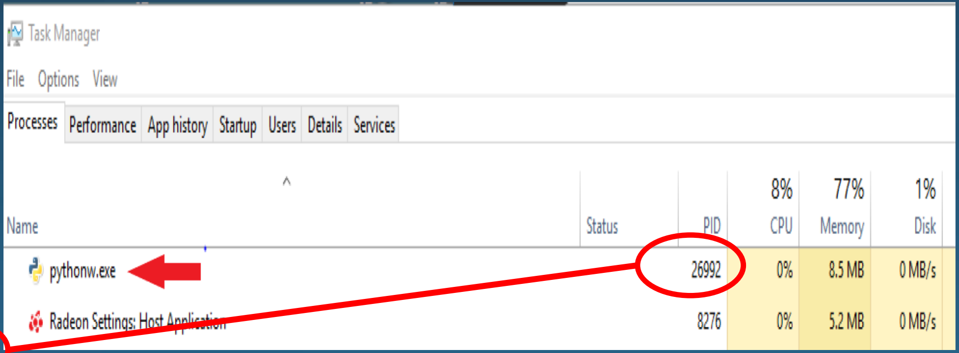
```
#!/usr/bin/python
# Written for Python version 2.7
# Hello World server in Python
# Binds REP socket to tcp://*:5555
# Expects b"Hello" from client, replies with b"World"

import time
import zmq
print 'pyzmq version: ', zmq.pyzmq_version()

context = zmq.Context()
socket = context.socket(zmq.REP)
socket.bind("tcp://*:5555")
print "Server running."
while True:
    message = socket.recv() #Wait for next request from client
    print("Received request: %s" % message)
    socket.send(b"World") #  Send reply back to client
    time.sleep(.1)
```


Python 2.7.14 Shell

```
Python 2.7.14 (v2.7.14:84471935ed, Sep 16 2017, 20:19:30) [MSC v.1500 32 bit (In
tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:\Google Drive\PythonCode\LABVIEW TCP\Hello World Serve Simpler.py
pyzmq version:  17.1.2

Traceback (most recent call last):
  File "C:\Google Drive\PythonCode\LABVIEW TCP\Hello World Serve Simpler.py", li
ne 13, in <module>
    socket.bind("tcp://*:5555")
  File "zmq\backend\cython\socket.pyx", line 547, in zmq.backend.cython.socket.S
ocket.bind
  File "zmq\backend\cython\checkrc.pxd", line 25, in zmq.backend.cython.checkrc.
_check_rc
    raise ZMQError(errno)
ZMQError: Address in use
>>>
```


Command Prompt

```
C:\Users\Admin>netstat -a -n -o

Active Connections

  Proto  Local Address          Foreign Address        State           PID
  TCP    0.0.0.0:80             0.0.0.0:0              LISTENING       4
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING       1172
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING       4
  TCP    0.0.0.0:1801           0.0.0.0:0              LISTENING       3740
  TCP    0.0.0.0:2103           0.0.0.0:0              LISTENING       3740
  TCP    0.0.0.0:2105           0.0.0.0:0              LISTENING       3740
  TCP    0.0.0.0:2107           0.0.0.0:0              LISTENING       3740
  TCP    0.0.0.0:2343           0.0.0.0:0              LISTENING       2820
  TCP    0.0.0.0:3389           0.0.0.0:0              LISTENING       1344
  TCP    0.0.0.0:3580           0.0.0.0:0              LISTENING       3012
  TCP    0.0.0.0:3582           0.0.0.0:0              LISTENING       5376
  TCP    0.0.0.0:5040           0.0.0.0:0              LISTENING       1800
  TCP    0.0.0.0:5357           0.0.0.0:0              LISTENING       4
  TCP    0.0.0.0:5555           0.0.0.0:0              LISTENING       26992
  TCP    0.0.0.0:6646           0.0.0.0:0              LISTENING       4700
  TCP    0.0.0.0:8080           0.0.0.0:0              LISTENING       5936
  TCP    0.0.0.0:8081           0.0.0.0:0              LISTENING       2584
```


Task Manager

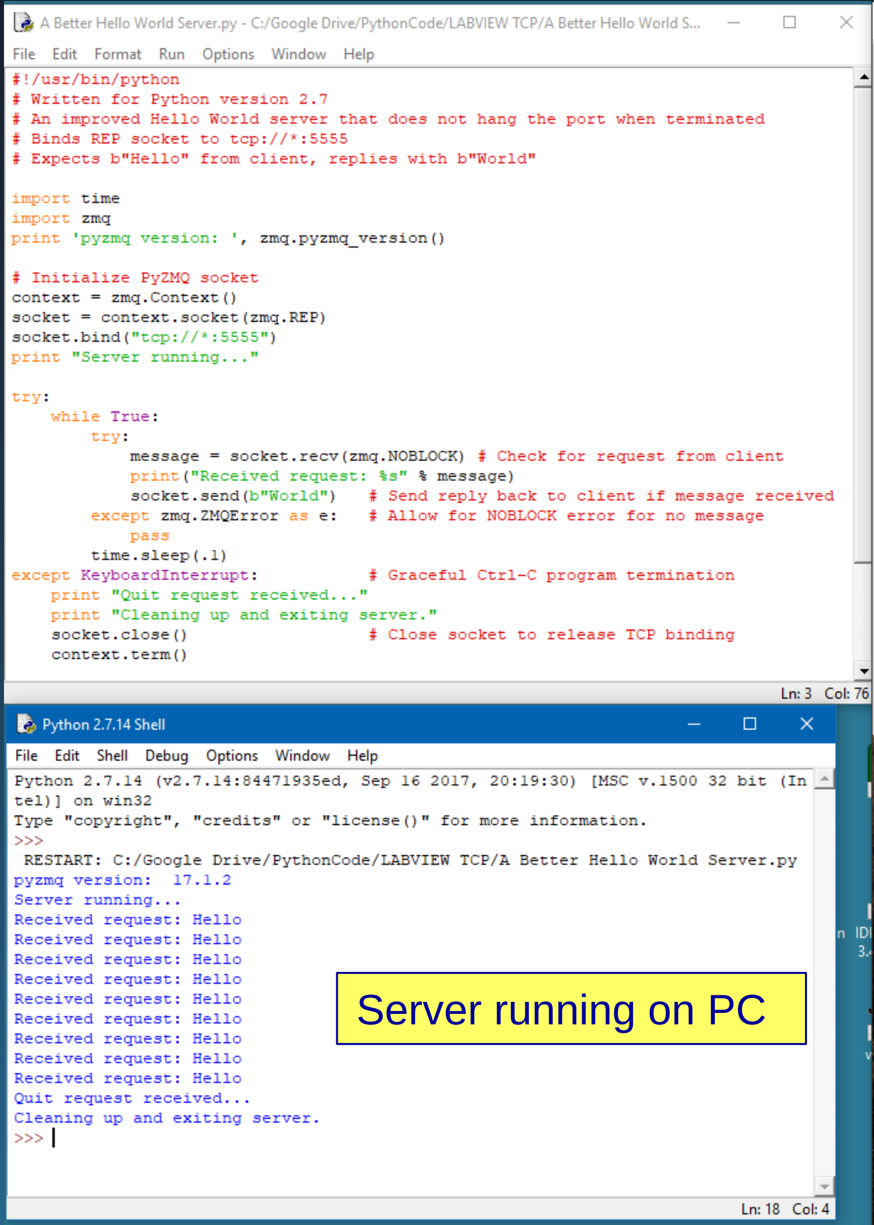
File Options View
Processes Performance App history Startup Users Details Services

Name	Status	PID	8% CPU	77% Memory	1% Disk
pythonw.exe		26992	0%	8.5 MB	0 MB/s
Radeon Settings: Host Application		8276	0%	5.2 MB	0 MB/s

Code Solution Notes

- Client Program Modifications
  - The original client Hello World” program was modified to test several addressing schemes.
    - Local network address
    - Internet numeric address
    - A domain name server address
- Server Program Modifications
  - The original server “Hello World” program was modified by adding two nested levels of “try” clauses to allow for error processing. In order for a Ctrl-C program termination to be caught the while loops needs to run freely. The original server uses the receive message statement in blocking mode. This means the program is paused listening for messages and the while loop does not run in order to catch the Ctrl-C. The while loop will run if the receive message is used in non-blocking mode but this results in a EAGAIN error “Non-blocking mode was requested and no messages are available at the moment”. This is more of a behavior than an error so the “try” clause lets it pass through allowing the program to loop. Thus the outer “try” clause is able to look for the program termination and close the socket and exit the program without error.

Final Code



A Better Hello World Server.py - C:/Google Drive/PythonCode/LABVIEW TCP/A Better Hello World S...

```
#!/usr/bin/python
# Written for Python version 2.7
# An improved Hello World server that does not hang the port when terminated
# Binds REP socket to tcp://*:5555
# Expects b"Hello" from client, replies with b"World"

import time
import zmq
print 'pyzmq version: ', zmq.pyzmq_version()

# Initialize PyZMQ socket
context = zmq.Context()
socket = context.socket(zmq.REP)
socket.bind("tcp://*:5555")
print "Server running..."

try:
    while True:
        try:
            message = socket.recv(zmq.NOBLOCK) # Check for request from client
            print("Received request: %s" % message)
            socket.send(b"World")    # Send reply back to client if message received
        except zmq.ZMQError as e:    # Allow for NOBLOCK error for no message
            pass
        time.sleep(.1)
except KeyboardInterrupt:            # Graceful Ctrl-C program termination
    print "Quit request received..."
    print "Cleaning up and exiting server."
    socket.close()                   # Close socket to release TCP binding
    context.term()
```

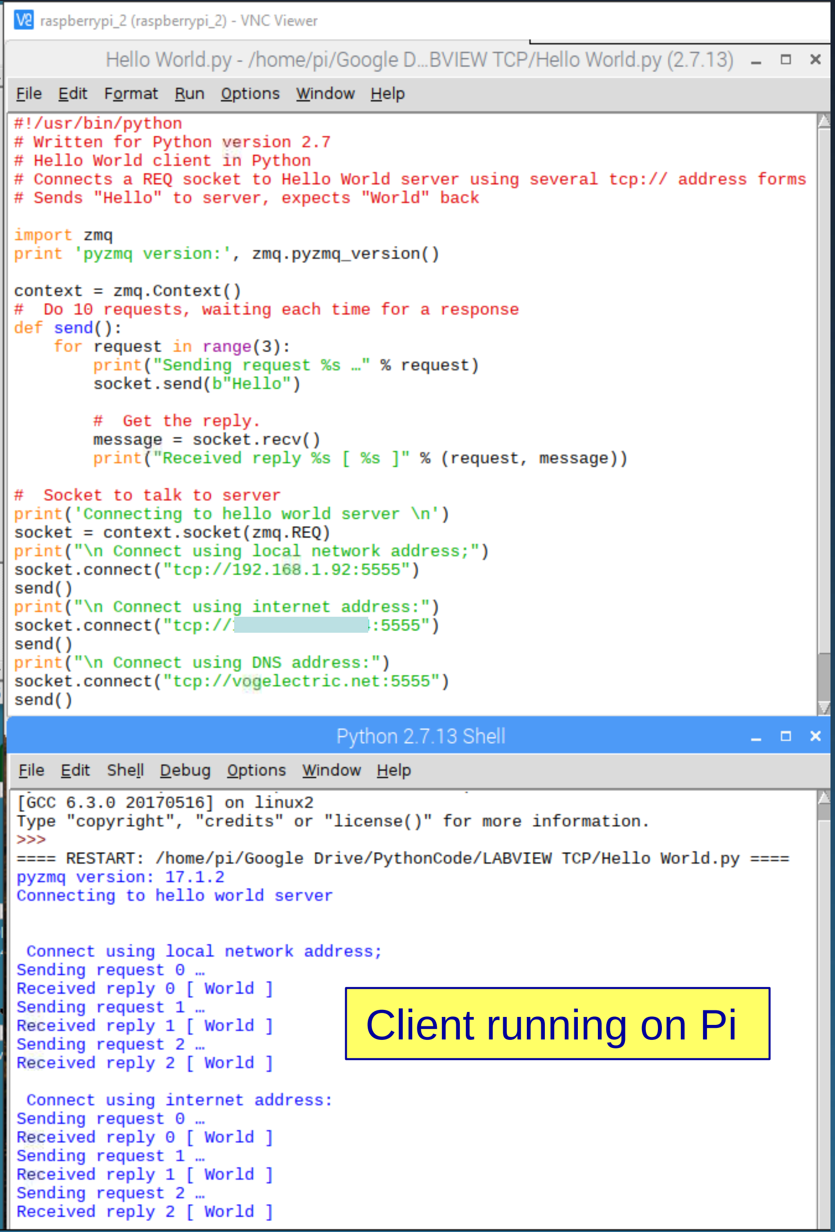
Ln: 3 Col: 76

Python 2.7.14 Shell

```
Python 2.7.14 (v2.7.14:84471935ed, Sep 16 2017, 20:19:30) [MSC v.1500 32 bit (In
tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:/Google Drive/PythonCode/LABVIEW TCP/A Better Hello World Server.py
pyzmq version:  17.1.2
Server running...
Received request: Hello
Received request: Hello
Received request: Hello
Received request: Hello
Received request: Hello
Received request: Hello
Received request: Hello
Received request: Hello
Received request: Hello
Quit request received...
Cleaning up and exiting server.
>>> |
```

Ln: 18 Col: 4

Server running on PC



raspberrypi_2 (raspberrypi_2) - VNC Viewer

Hello World.py - /home/pi/Google D...BVIEW TCP/Hello World.py (2.7.13)

```
#!/usr/bin/python
# Written for Python version 2.7
# Hello World client in Python
# Connects a REQ socket to Hello World server using several tcp:// address forms
# Sends "Hello" to server, expects "World" back

import zmq
print 'pyzmq version:', zmq.pyzmq_version()

context = zmq.Context()
#  Do 10 requests, waiting each time for a response
def send():
    for request in range(3):
        print("Sending request %s …" % request)
        socket.send(b"Hello")

        #  Get the reply.
        message = socket.recv()
        print("Received reply %s [ %s ]" % (request, message))

#  Socket to talk to server
print('Connecting to hello world server \n')
socket = context.socket(zmq.REQ)
print("\n Connect using local network address;")
socket.connect("tcp://192.168.1.92:5555")
send()
print("\n Connect using internet address:")
socket.connect("tcp://[redacted]:5555")
send()
print("\n Connect using DNS address:")
socket.connect("tcp://vogelectric.net:5555")
send()
```

Python 2.7.13 Shell

```
[GCC 6.3.0 20170516] on linux2
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: /home/pi/Google Drive/PythonCode/LABVIEW TCP/Hello World.py ====
pyzmq version: 17.1.2
Connecting to hello world server


 Connect using local network address;
Sending request 0 …
Received reply 0 [ World ]
Sending request 1 …
Received reply 1 [ World ]
Sending request 2 …
Received reply 2 [ World ]

 Connect using internet address:
Sending request 0 …
Received reply 0 [ World ]
Sending request 1 …
Received reply 1 [ World ]
Sending request 2 …
Received reply 2 [ World ]
```

Client running on Pi