# Test Coverage

Since this was a fairly small program I had 6 unit tests and 5 integration tests. The unit tests were fairly comprehensive covering every function and so were the integration tests. Though, to fill out the requirements of 5 integration tests I had to stretch the functionality of them. I.E if I had a function that already used a different function inside of it, I would test those functions separately within the bounds of the main function.

## Unit Tests:

```
def testCountPacket(self): Test to see if each packet marking is correct,
each call should increment the graph G by 1

def testCountEdgePacket(self): Same as testCountPacket except this counts
an edge. Since it takes an edge we check for tuples

def testSendPacket(self): Sends a packet to router 20 in the graph, should
only send 1 packet per call

def testSendEdgePacket(self): Same as above, except it increments a
different attribute of router 20

testNodeSampling(self): Tests if the malicious actor using the node
sampling algorithm is correct

def testEdgeSampling(self): Same as above, using the edge sampling
algorithm instead
```

## Integration Tests:

```
def testSendNodePacketsIntegration(self): tests the node send function
within the main function, the output is a little random based on the
probability of each router, so checks for outliers

def testSendEdgePacketsIntegration(self): Same as above, except it tests a
different variable in router 20
```

```
def testNodeSamplingIntegration(self): Tests to see if the sendNodePackets
function + the node sampling algorithm has correct data

def testEdgeSamplingIntegration(self): Same as above using the edge send
function and edge sampling algorithm

def testNodeCountAndEdgeCountIntegration(self): Tests to see if the
combination of the two sending functions work together in the main
functionality
```

## **Automated Testing**

The automated testing was fairly easy. I used an example workflow supplied by GitHub and then modified it to use Coverage.py to run my tests automatically on push. It’s written in yaml and can be found in CS_333_Final_Repo/.github/workflows/python-package.yml

## **Automated Deployment**

This was the hardest part of the project. I tried AWS, Docker, Azure, Jenkins. Eventually I had to use GitHub pages because it was the simplest to learn and deploy. The deployment page is very simple, but it took all my spirit to learn how to do it. Since it’s built with Github Pages, it deploys automatically on push of the main branch.
You can find it at https://dhuynhvo.github.io/CS_333_Final_Repo/ With a download link for the main repo