Conclusion

Although I originally believed that a random number generator could be made using images cached from social media. When I began to develop my applications I quickly learned that social media sites put strict limits on the amount of images that can be requested per hour. Although it could be done, it would be far too difficult for me to accomplish with my experience. I found another alternative that came close to what I was looking to accomplish, I decided to use tweets, which are readily available to be streamed to an application in real time. I decided to stick to testing that and only that because it would be best if I focused my efforts on one specific application. I developed multiple versions of this application to compare with the PHP rand() Function. When I did compare the data I found that my final version of the twitter generator had the highest p-value, but all of the p-values were within around 20% of each other. The p-value of Twitter Generator v4 was 6 trillion more than that of the PHP rand() Function. I think that although my the Twitter Generator v4 worked the best (Twitter Generator v1 being the worst) none of the random number generators did poorly. The generation of random numbers is not a large issue in security now, but once quantum computers are developed and the prime factorization of a 256 digit number can be found in a reasonable amount of time, that little extra bit from a better random number generator might be the difference between secure and security breach.