Midterm Reflection: Socket Programming and Port Scanning

CYB 333 | Sean Ayersman

Isaiah Caparoula

6/15/2025

For this project, I built a basic socket connection system and a Python-based port scanner using VSCode. I started by creating a server that listens for incoming TCP connections on localhost, and a client that connects to the server and sends messages. My goal was to ensure clean communication between both scripts, including proper disconnection and error handling. I also added logic so that typing “exit” on the client side would trigger a clean shutdown on both ends.  
  
One challenge I faced was dealing with connection errors when the server wasn't running. Initially, the client would crash if it couldn’t find the server. To fix this, I added a try-except block specifically for ConnectionRefusedError, which allowed me to display a friendly message instead of an unhandled crash. Another challenge was ensuring the port scanner didn’t run too fast, so I added a short delay (time.sleep(0.1)) between scans to avoid triggering rate limits or simulating a denial-of-service beh...  
  
I used ChatGPT as a coding assistant in parts of this assignment. For example, I asked: “Write a TCP socket server in Python with graceful shutdown and client message handling.” The initial code helped me set up the structure, but I customized the logic to handle my test cases, like echoing back responses and using with blocks for auto-closing sockets. I also asked ChatGPT to explain how connect\_ex() works in a port scanner so I could avoid false positives when detecting open ports.  
  
Security-wise, I made sure to only scan 127.0.0.1 and scanme.nmap.org, which are both authorized for testing. I avoided scanning random public IPs and added input validation for ports to prevent errors. According to Skoudis and Liston (2006), ethical port scanning is critical in penetration testing, but only when permission is granted. This project reinforced how important it is to respect scanning boundaries and prevent misuse of scanning tools.  
  
Overall, this project helped me understand how network communication works at a basic level and how to automate parts of it with Python. I learned how to build simple socket apps, catch exceptions, and use input validation to secure my code. I also saw how tools like port scanners are essential in security automation for identifying open ports and potential vulnerabilities—when used responsibly.

Reference

Skoudis, E., & Liston, T. (2006). \*Counter Hack Reloaded: A Step-by-Step Guide to Computer Attacks and Effective Defenses\* (2nd ed.). Prentice Hall.