**Dustin Allen**  
**CYB333**  
**Midterm**

**Socket Programming and Port Scanning: A Reflection**

For this midterm, I built two security tools in Python: a client-server socket communication system and a port scanner. The goal was to simulate basic network communication and perform ethical port scanning on approved systems. These tools are foundational in penetration testing, vulnerability assessments, and understanding how attackers probe networks.

To implement the socket tools, I created two scripts—one for the server and another for the client.

* **Server**: Listens on a specific port, accepts a connection, receives a message, and sends a response.
* **Client**: Connects to the server, sends a message, and prints the response.

I tested the client-server interaction to ensure proper communication and clean disconnections. I also tested the client when the server was offline, which triggered a ConnectionRefusedError, confirming that the error handling worked as expected.

The port scanner checks a range of ports on a given host by attempting TCP connections. I tested it on 127.0.0.1 and scanme.nmap.org, a site specifically designated for safe scanning. I also experimented with invalid inputs such as negative port numbers and unreachable hosts. For example, attempting to scan a negative port threw an OverflowError, which taught me how the socket module enforces valid port ranges.

I used ChatGPT as an auxiliary tool throughout the project. I asked for an example of basic client-server socket communication and used it as a base. From there, I expanded it with additional logic, error handling, and refinements. I also used ChatGPT to optimize the port scanner’s logic, ensuring it followed Pythonic practices. While AI helped draft and debug, I made sure I understood every line of code and customized everything to meet the assignment’s requirements.

From an ethical standpoint, I kept all scans confined to localhost and scanme.nmap.org, as instructed. I avoided aggressive tactics, DoS attempts, or targeting unauthorized systems. As Skoudis and Liston (2006) emphasize, “A responsible port scan has permission, is aware of the impact of scanning, and makes no inadvertent attempts to trigger an incident or violate local policy.”

This project gave me insight into how automated tools simulate real-world attack vectors. More importantly, scripting those tools myself helped me better understand system vulnerabilities, input validation, and safe recon techniques. I now feel more capable of building defensive and offensive tools with confidence and integrity.

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