Asset Discovery Using Post Quantum Cryptography Demo Report

Objective:
To simulate how asset discovery agents can be protected against quantum threats by replacing classical key exchange and authentication methods (e.g., RSA, ECC) with post-quantum algorithms specifically, ML-KEM (Kyber) for key encapsulation and Dilithium for digital signatures. The system ensures that before any device scanning or communication begins, a PQC-secured connection is established between the discovery agent and the remote machine.

System Architecture

1. Target Device (Ubuntu VM)
   Acts as a simulated enterprise asset running:
   a. SSH service (OpenSSH)
   b. TLS service (optional OpenSSL endpoint)
   c. SNMP agent (snmpd)
   d. Flask API that returns a list of connected devices
   e. PQC secure server using ML-KEM_512 and Dilithium2
2. Host Machine (Windows/Linux/macOS)
   Runs the discovery system:
   a. PQC utility module (pqc_client_connection.py) for key exchange and verification
   b. Discovery agent script (discovery_agent.py) that performs:
      i. TLS check
      ii. SSH banner grab
      iii. SNMP query
      iv. Retrieval of mock internal asset data All operations are blocked until the PQC handshake succeeds.

Demo Flow

1. Setup the Ubuntu VM
   a. Install essential services (SSH, SNMP, Flask)
   b. Create a mock JSON file of connected devices
   c. Start the Flask API to serve this data
   d. Launch the PQC server which:
      i. Receives a Kyber public key
      ii. Sends back a ciphertext, shared secret, and Dilithium signature
2. Run Discovery from Host Machine
   a. The agent first connects to the PQC server
   b. After a successful ML-KEM decapsulation and Dilithium signature verification:
      i. Begins discovery operations on the VM
      ii. Saves results to a structured JSON report file

Results:

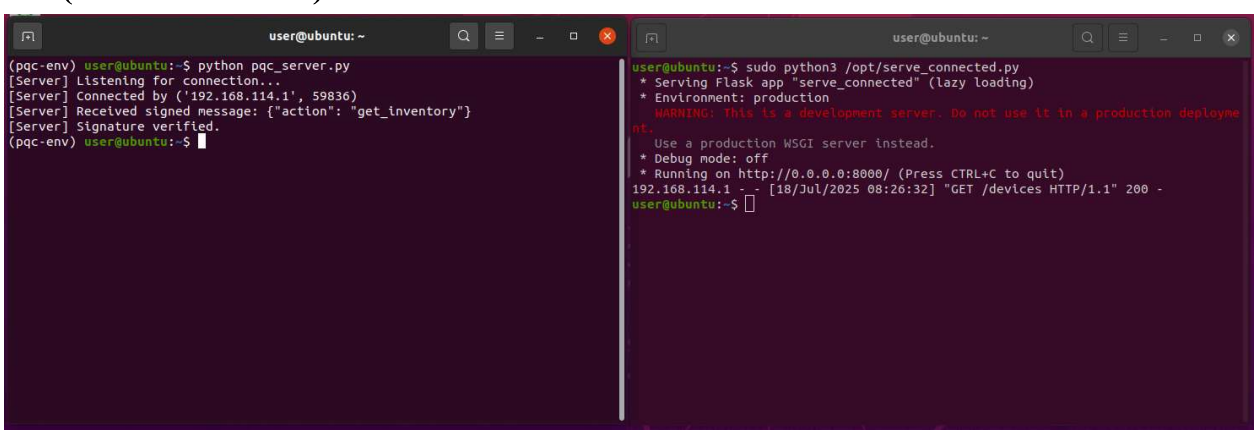
The demo produces a report (pqc_secure_discovery.json) containing:

1. TLS scan status (if reachable)
2. SSH banner from the target
3. SNMP query result
4. List of connected internal assets from the Flask API All actions are done only after a successful PQC handshake

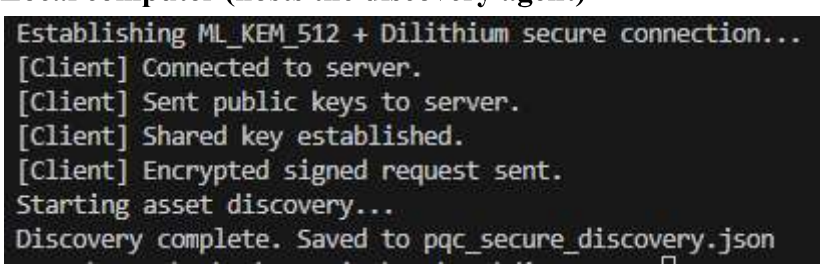
Note:
The demo can also be performed using just a regular computer. This can be done by changing the IP address in the scripts to local host and then setting up a terminal for the server and the other for the client and discovery agent.

Demo Visuals:

VM (Acts as the server)



Local computer (hosts the discovery agent)



Result:
Results can be seen in the pqc_secure_discovery.json file.