Offensive C++

Duration: 5 Days

Day 1: Introduction to C++ Programming & Networking Fundamentals

- **A. Introduction to C++ Programming**
    - Overview of C++ and its applications
    - Basic syntax, variables, data types, and operators
    - Control flow: if-else, loops (for, while), switch-case
    - Functions: definitions, parameters, return types
    - Introduction to classes, objects, and constructors
- **B. Working with C++ Data Structures**
    - Arrays and strings
    - Vectors, lists, and maps (Key-Value pairs)
    - Sets and iterators
- **C. Introduction to Networking in C++**
    - Basics of IP addresses, ports, and protocols (TCP/UDP)
    - Understanding sockets and the client-server model
    - Networking libraries in C++ (e.g., Boost.Asio)
- **D. Hands-On:**
    - Develop a simple networking program using sockets to send and receive data between two devices.

Day 2: Advanced Networking & Client-Server Communication

- **A. Advanced Socket Programming**
    - Socket creation: socket(), bind(), listen(), accept(), connect()

    - Sending and receiving data using send() and recv()
    - Handling multiple clients with non-blocking sockets
- **B. Creating the Configurator Structure**
    - Project structure and directory layout
    - Designing the Master-Slave communication flow
    - Choosing appropriate data structures for device management (e.g., maps for storing device configurations)
- **C. Client-Server Communication (Master-Slave Model)**
    - Designing the protocol for Master and Slave communication
    - Handling device configuration and querying status
    - Implementing server and client functionality for Master-Slave communication
- **D. Hands-On:**
    - Build a client-server application for basic Master-Slave communication, simulating device interactions.

Day 3: Packet Analysis, Transmission, and Debugging

- **A. Packet Structure and Transmission**
    - Introduction to network packet structure (headers, payload)
    - How to capture and analyze packets (Wireshark)
    - Sending custom packets in C++
- **B. Packet Analysis**
    - Capturing and analyzing packets exchanged between Master and Slave devices
    - Filtering packets based on IP address and port number
    - Analyzing errors and timeouts during communication
- **C. Debugging and Error Handling**
    - Using a debugger (e.g., GDB) to inspect program flow
    - Handling connection errors and invalid data
    - Using exception handling in C++ (try, catch)
- **D. Hands-On:**

- Implement packet transmission and error handling mechanisms, and use Wireshark to analyze packets.

Day 4: Multi-Threading, Real-Time Networking, and Device Communication

- **A. Multi-threading in C++**
  - Basics of multi-threading and thread synchronization
  - Handling concurrent connections from multiple slave devices
- **B. Real-Time Networking**
  - Techniques for low-latency, real-time communication
  - Optimizing network communication for real-time updates
- **C. Device Communication Protocol**
  - Defining a custom protocol for Master-Slave communication
  - Sending configuration data, receiving status updates
- **D. User Interface for Configurator (CLI/GUI)**
  - **Text-based Interface (CLI)**: Using libraries like ncurses for terminal-based UI.
  - **GUI (Graphical User Interface)**: Introduction to simple GUI using Qt or GTK for managing devices.
  - Designing a simple interface to allow users to interact with the configurator (add/remove devices, configure settings, view device status)
- **E. Hands-On:**
  - Implement a multi-threaded server to manage multiple slave devices.
  - Build a simple **CLI** interface for interacting with the configurator.
  - **GUI**: Build a basic GUI (e.g., using Qt) for managing devices and viewing their status.

Day 5: Final Project and Wrap-Up

- **A. Final Project Work**
  - Setting up communication between Master and Slave devices.
  - Building a simple interface (CLI or GUI) for the configurator.
  - Implementing real-time updates and configuration management.

- **B. Project Testing and Debugging**
    - Testing the full system: Master-Slave communication, packet handling, and configuration management.
    - Troubleshooting and optimizing the system.
- **C. Final Hands-On:**
    - Complete the full configurator software, ensuring it can communicate with and manage multiple slave devices.
    - Implement and test the **CLI**/ **GUI** for device management (adding/removing devices, sending configuration commands).
    - Display logs for packets sent/received.
- **D. Review of Advanced Topics (Optional)**
    - Introduction to SSL/TLS for secure communication.
    - Cross-platform development (Windows/Linux compatibility).