WPF Application Overview: VLAN Management Tool

1. Application Purpose and Technology Stack

This WPF application, developed using C# and .NET Core 3.1 in Visual Studio, is designed for network administrators to manage VLANs by interacting with HP and Extreme switches. It provides a GUI that facilitates the scanning of network interfaces, identifies connected switches, retrieves relevant data via LLDP packets, and allows VLAN changes via SSH connections.

2. Network Scan and LLDP Packet Parsing

The application allows the user to initiate a network scan through the GUI. It listens for LLDP packets on the Ethernet interface to identify the IP address, port, and VLAN ID. For HP switches, all this data is directly available in the LLDP packets. For Extreme switches, however, the VLAN ID is not present, prompting additional steps.

3. Handling HP vs Extreme Switches

A key logic flag, `is_extreme_switch`, determines the subsequent processing flow. If set to `false` (HP), the application parses the LLDP packet directly. If `true` (Extreme), the application uses the SSH.NET library to connect to the switch using RADIUS server authorization. It queries the VLAN ID for the specific port based on the known switch IP and port number.

4. VLAN Modification Feature

The application includes a secondary GUI window that allows users to change the VLAN of a specific switch port. Users input the IP address and port number, select the switch type (HP or Extreme), and choose a VLAN from a dropdown menu. Depending on the switch type, different commands are executed using the SSH.NET library. HP switches use local credentials, while Extreme switches authenticate via RADIUS using the user's credentials.

5. Custom Command Execution and Output Display

The tool provides an interface to execute custom commands on the switches and displays the command output in a designated output screen within the GUI, enhancing administrative flexibility and feedback.

6. Asynchronous Execution and UI Updates

All operations that interact with the network or perform lengthy processing are handled asynchronously using separate threads. Output and UI updates are dispatched to the main thread using the Dispatcher mechanism to maintain responsiveness and avoid UI freezing.