Notes

Cisco Configuration Deployment Script

Introduction
This Python script is designed to facilitate the copying of templates to Cisco routers or switches via console connection. It provides a simple interface for selecting the COM port, baud rate, and entering the password required for accessing privileged EXEC mode on the device. The script then copies the specified template file to the device and generates a log file to track the process.

How it works
The script establishes a serial connection with the Cisco device using the provided COM port and baud rate. It then sends commands to enter privileged EXEC mode and authenticate using the provided password if necessary. Once in privileged mode, the script reads a template file line by line and sends each line as a command to the Cisco device. After sending all commands, it waits for the device to process them and generates a log file containing the output received from the device.

Features
- Easy-to-use script for copying templates to Cisco devices via console connection.
- Supports selection of COM port and baud rate.
- Password authentication for privileged EXEC mode access.
- Generates a log file to track the template copying process.

Dependencies
- Python 3.x pyserial (included in Python standard library)

License
This project is licensed under the MIT License.

GitHub
Share Link: https://github.com/Anthony-Constant/Cisco-Configuration-Deployment

PYTHON COPY & PASTED LOCAL SOURCE CODE

```
############################################################
# Script to deploy configs/templates to cisco devices i.e. switches/routers              #
# Author: Anthony Constant                                                              #
# Date: 22/04/2024                                                                       #
############################################################

import serial
import time
import serial.tools.list_ports

def read_until_prompt(ser, prompt, timeout=5):
    start_time = time.time()
    response = b""
    while True:
        if time.time() - start_time > timeout:
            break
        data = ser.read()
        if data:
            response += data
        if prompt in response:
            break
    return response

def copy_template(com_port, baud_rate, password, template_file, log_file):
    try:
        # Serial connection setup
        ser = serial.Serial(com_port, baud_rate, timeout=1)
        ser.write(b"\r\n")
        print("Initiating template copy process…")
        # Give some time for the serial connection to stabilize
        time.sleep(2)
        # Send command to the device to enter privileged EXEC mode
        print("Entering privileged EXEC mode...")
```

```
        ser.write(b"enable\r\n")
        # Check if already in enable mode
        output = read_until_prompt(ser, b"#")
        if b"Password:" in output:
            print("Sending password...")
            ser.write(password.encode('utf-8') + b"\r\n")
            read_until_prompt(ser, b"#")
        # Send commands to copy template
        print(f"Copying template from {template_file}...")
        with open(template_file, "r") as template:
            for line in template:
                ser.write(line.encode('utf-8') + b"\r\n")
                time.sleep(0.5)  # Adjust delay between commands if necessary
        # Read the output until no data is received for a short period
        copy_output = ser.read_all().decode()
        # Save the copy log to a file
        with open(log_file, "w") as file:
            file.write(copy_output)
        print(f"Template copied successfully. Log saved to {log_file}")
        # Close serial connection
        ser.close()
    except serial.SerialException as se:
        print(f"Serial port error: {se}")
    except Exception as e:
        print(f"An error occurred: {e}")

if __name__ == "__main__":
    try:
        available_ports = list(serial.tools.list_ports.comports())
        print("Available COM ports:")
        for idx, port in enumerate(available_ports):
            print(f"{idx + 1}. {port.device}")
        selection = int(input("Enter the number corresponding to the desired COM port: "))
        selected_port = available_ports[selection - 1].device
        baud_rate = int(input("Enter the BAUD rate: "))
        password = input("Enter your password: ")
        template_file = input("Enter the filename of the template to copy: ")
        log_file = input("Enter the filename for the copy log (e.g., copy_log.txt): ")
```

```
        copy_template(selected_port, baud_rate, password, template_file, log_file)
    except (IndexError, ValueError):
        print("Invalid selection. Please enter a valid number.")
```