LDP_CHECK-V1.py Documentation

Purpose
This script runs on Extreme EXOS (Python 2.7) to check OSPF + LDP health for two primary VLAN paths and ensure the VPLS next-hop is correct. If the next-hop is wrong, it toggles MPLS LDP on selected VLANs to recover adjacency.

Environment
- **Platform**: Extreme EXOS (uses `exsh.clicmd`).
- **Python**: 2.7 compatible.
- **Execution**: Intended to run on the switch/router where `clicmd` is available.

Configuration
- **Primary VLAN**
  - `PRIMARY_VLAN`: VLAN name
  - `PRIMARY_VLAN_IP`: OSPF neighbor IP
- **Secondary VLAN**
  - `SECONDARY_VLAN`: VLAN name
  - `SECONDARY_VLAN_IP`: OSPF neighbor IP
- **Backup VLAN**
  - `BACKUP_VLAN`: VLAN name
  - `BACKUP_VLAN_IP`: Expected VPLS next-hop in full failover
- **VPLS**
  - `VPLS_PEER`: Peer IP (not used in logic)
  - `VPLS_SERVICE`: VPLS service name used to read next-hop
- **Cooldown**
  - `COOLDOWN_SEC`: Minimum delay between recovery actions
  - `COOLDOWN_FILE`: File storing the last action timestamp
- **Logging**
  - `SYSLOG_TAG`: Syslog tag
- **Safety**
  - `DRY_RUN`: If `True`, log only without making changes

Functional Flow
1) Logging and Cooldown
- `log(msg)` writes a syslog message using `clicmd`.
- `cooldown_active()` checks if the last action is within the cooldown window.
- `set_cooldown()` records the current time to prevent repeated toggles.

2) Health Checks
- **OSPF neighbor state**
  - `ospf_neighbor_is_full(ip)` runs `show ospf neighbor` and checks if the neighbor is `FULL`.
- **LDP adjacency**
  - `ldp_adjacency_up(vlan)` runs `show mpls ldp interface` and checks adjacency count for the VLAN.
- **VPLS next-hop**
  - `get_vpls_nexthop()` runs `show vpls <service> detail` and extracts `Next Hop Addr`.

3) Recovery Action
- `toggle_vlan(vlan_name)`:
  - `disable mpls ldp vlan <vlan>`
  - waits 60 seconds
  - `enable mpls ldp vlan <vlan>`
  - Uses `DRY_RUN` to avoid actual changes if enabled.

Decision Logic (Main)
1. **Read current VPLS next-hop**
   - If not found → log error and exit.
2. **Check PRIMARY** (OSPF + LDP):
   - If **PRIMARY up** and **next-hop is correct** → all OK.
   - If **PRIMARY up** but **next-hop wrong**:

     - If cooldown active → skip.
     - Otherwise **toggle SECONDARY and BACKUP** VLANs.
3. **PRIMARY down, SECONDARY up**:
   - If **SECONDARY up** and **next-hop correct** → all OK (failover to secondary).
   - If **SECONDARY up** but **next-hop wrong**:
     - If cooldown active → skip.
     - Otherwise **toggle BACKUP** VLAN only.
4. **Both PRIMARY and SECONDARY down**:
   - Log warning and check if VPLS next-hop equals BACKUP.
   - If backup next-hop correct → log failover to backup.
   - If not → log issue.

Usage
Run on an EXOS device:
```bash
python LDP_CHECK-V1.py
```

For testing without changes:
- Set `DRY_RUN = True`.

Notes and Behavior
- The cooldown prevents rapid repeated toggles.
- Recovery is done by toggling MPLS LDP on VLANs to reestablish adjacency.
- `VPLS_PEER` is defined but not used in current logic.