

NetFixGo: Automated Network Troubleshooting and Validation

NetFixGo (*Network Fix Go*) is a Python-based CLI tool for automated network troubleshooting and validation. It focuses on comparing network device configurations and OSPF neighbor states against predefined golden states and golden configuration, resolving detected issues dynamically, and notifying you via Slack. It simplifies the management of network reliability and ensures compliance with golden configurations.

The primary functionalities include:

- Dynamic comparison of device configurations with golden configurations.
- Validation of OSPF neighbor states.
- Automated resolution of detected issues, such as:
 - Fixing mismatched OSPF timers.
 - Resolving OSPF shutdown configurations.
- Real-time Slack notifications for issues and resolutions.

Note: While the tool currently only supports OSPF automated resolution, this can be further enhanced to include more protocols and features.

Features

1. Configuration Comparison: NetFixGo compares the current device configuration against the golden configuration. It detects any divergences, removes unwanted configurations, and restores compliance by applying the golden configuration.

```
student@csci5840-vm2-snr8112:~/git/csci5840$ netfixgo
2024-12-02 15:49:21.366 | DEBUG | __main__:main:417 - Connecting to r1 (192.168.100.2)...
2024-12-02 15:49:21.996 | WARNING | __main__:compare_configs:91 - Differences found for r1:
--- Golden Config
+++ Current Config
@@ -106,6 +106,7 @@
!
router ospf 1
+shutdown
network 11.0.0.0/24 area 0.0.0.0
network 12.0.0.0/24 area 0.0.0.0
network 100.0.0.0/29 area 0.0.0.0
2024-12-02 15:49:23.455 | SUCCESS | __main__:compare_configs:113 - Reverted the config back to golden config
```

Screenshot 1: Netflix identifying divergences on device R1

2. OSPF Neighbor Validation: The tool validates OSPF neighbor states against the golden state. It identifies and logs missing or unexpected neighbors, mismatched OSPF timers, or incorrect neighbor states (e.g., *non-FULL/2WAY* states).

```
2024-12-02 15:40:04.821 | DEBUG | __main__:check_ospf_neighbors:371 - Checking OSPF neighbors on r1...
2024-12-02 15:40:04.888 | ERROR | __main__:compare_ospf_neighbors:275 - Missing OSPF neighbors on r1:
2024-12-02 15:40:04.888 | ERROR | __main__:compare_ospf_neighbors:278 - 200.0.0.1
2024-12-02 15:40:04.888 | ERROR | __main__:compare_ospf_neighbors:278 - 200.0.0.2
2024-12-02 15:40:04.888 | ERROR | __main__:compare_ospf_neighbors:278 - 192.168.100.3
```

Screenshot 2: NetfixGo identifying missing neighbors

3. Automated Issue Resolution: NetFixGo dynamically resolves common OSPF issues, such as:

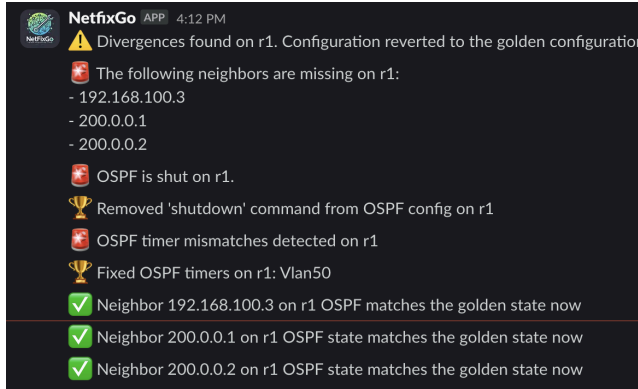
- Removing shutdown from OSPF configurations.
- Correcting OSPF interface timer mismatches.

```
2024-12-02 15:35:35.894 | DEBUG | __main__:compare_ospf_config:217 - Comparing OSPF configuration between r1 and neighbor 200.0.0.1...
2024-12-02 15:35:36.086 | WARNING | __main__:compare_ospf_config:222 - OSPF is shut on r1. Fixing...
200
2024-12-02 15:35:36.334 | DEBUG | __main__:compare_ospf_config:226 - Removed 'shutdown' from OSPF configuration on r1.
200
2024-12-02 15:35:46.519 | DEBUG | __main__:compare_ospf_neighbors:305 - Re-checking OSPF neighbor: 200.0.0.1 on r1 after fixing 'shutdown' in 10s..
200
2024-12-02 15:35:56.708 | WARNING | __main__:compare_ospf_neighbors:326 - Timer mismatches detected on r1. Fixing timers...
200
2024-12-02 15:35:56.981 | DEBUG | __main__:compare_ospf_neighbors:337 - Fixed timers for interface Vlan50 on r1.
200
2024-12-02 15:35:57.153 | DEBUG | __main__:compare_ospf_neighbors:340 - Re-checking OSPF neighbors on r1 after fixing timers in 10s...
2024-12-02 15:36:07.230 | SUCCESS | __main__:compare_ospf_neighbors:350 - Neighbor 200.0.0.1 on r1 matches the golden state after fixing timers.
200
2024-12-02 15:36:07.403 | DEBUG | __main__:compare_ospf_config:217 - Comparing OSPF configuration between r1 and neighbor 192.168.100.3...
2024-12-02 15:36:07.711 | DEBUG | __main__:compare_ospf_config:251 - All OSPF timers on r1 are correctly configured.
2024-12-02 15:36:07.711 | DEBUG | __main__:compare_ospf_neighbors:305 - Re-checking OSPF neighbor: 192.168.100.3 on r1 after fixing 'shutdown' in 10s...
2024-12-02 15:36:17.783 | SUCCESS | __main__:compare_ospf_neighbors:315 - Neighbor 192.168.100.3 on r1 matches the golden state after fixing 'shutdown'.
200
2024-12-02 15:36:17.956 | DEBUG | __main__:compare_ospf_config:217 - Comparing OSPF configuration between r1 and neighbor 200.0.0.2...
2024-12-02 15:36:18.189 | DEBUG | __main__:compare_ospf_config:251 - All OSPF timers on r1 are correctly configured.
2024-12-02 15:36:18.190 | DEBUG | __main__:compare_ospf_neighbors:305 - Re-checking OSPF neighbor: 200.0.0.2 on r1 after fixing 'shutdown' in 10s..
2024-12-02 15:36:28.256 | SUCCESS | __main__:compare_ospf_neighbors:315 - Neighbor 200.0.0.2 on r1 matches the golden state after fixing 'shutdown'.
```

Screenshot 4: NetFix go dynamically detecting and fixing OSPF config and timer issues

4. Slack Integration The tool integrates with Slack for real-time notifications:

- Reports missing neighbors or mismatched states.
- Notifies successful resolutions and unresolved issues.
- Uses emoji annotations to highlight severity and status.



Screenshot 5: Slack notifications

Usage

NetFixGo can be run directly via the CLI. It reads device information from a CSV file (*sshInfo.csv*), establishes SSH connections, and validates device configurations.

Key Functions

1. `compare_configs`

- Compares the running configuration of a device with its golden configuration.
- Resolves mismatches by applying golden configurations.
- Sends Slack notifications for detected differences and applied fixes.

2. `compare_ospf_neighbors`

- Compares OSPF neighbor states against the golden state.
- Logs and resolves:
 - Missing neighbors.
 - Mismatched neighbor states.
 - Unexpected neighbors.
- Automatically fixes detected issues using the *compare_ospf_config* function.

3. `compare_ospf_config`

- Validates OSPF configurations for:
 - Shutdown conditions.
 - Timer mismatches (*hello, dead, retransmit intervals*).

- Applies necessary corrections and rechecks OSPF states.

4. send_slack_notification

- Sends formatted messages to a predefined Slack webhook URL.
 - Supports emoji-based annotation for urgency and success statuses.
-