Dynamic Analysis for OpenThread Network Interoperability Testing

Attempted Approaches and Limitations

1. **Multi-container OpenThread Implementation**
   I explored a Docker-based approach to simulate a Thread network using multiple containers. However, no official examples or complete setups were available. Despite several iterations, I was unable to achieve inter-container communication using OpenThread CLI — the nodes could not discover each other or form a network. This approach was ultimately abandoned due to time constraints and lack of community support.

2. **Device Emulation for OpenThread-Capable Hardware**
   I also attempted to emulate OpenThread-compatible devices using various methods (e.g., ZephyrOS builds and Renode and Openthread Supported devices). These efforts were blocked by firmware image build failures and runtime issues, preventing successful boot or CLI access in emulated nodes.

- I spent approximately **two weeks** attempting to resolve these issues. In order to continue progress on interoperability metric evaluation, I switched to using **OTNS**, OpenThread's official simulator, which provides reliable simulation of node communication, though without RTOS-level execution or hardware emulation.
- I plan to revisit these approaches at a later stage to address the issues in more depth. If the problems can be resolved, a version using emulated devices in a multi-container setup with ZephyrOS may be developed and evaluated alongside the current OTNS-based simulation.

Current Status

- OTNS is installed and functioning under WSL (Ubuntu 22).
- A CLI wrapper is implemented for communication with OpenThread CLI nodes.
- All 7 baseline metric functions are in the implementation stage using Python.
- InfluxDB and Grafana setup for real-time metric logging and visualization is in progress.

Next Steps

- Run controlled simulations with 20 nodes to collect baseline data.
- Integrate InfluxDB logging and Grafana visualization.
- Begin Step 2: Dynamic Topology Scaling experiments.

References

1. OpenThread Documentation. https://openthread.io
2. OTNS Simulator. https://github.com/openthread/ot-ns

Dynamic Analysis for Interoperability (DAfI)

Dynamic analysis evaluates a system's real-time behavior and performance during execution.
This approach, also known as Dynamic Program Analysis or Dynamic Application Security Testing (DAST), is essential for identifying network stability, role transition behaviors, and

protocol efficiency in a **simulated OpenThread environment using OTNS**.

By focusing on a consistent simulation platform (OTNS with OpenThread CLI simulation), this study ensures fair evaluations of OpenThread's network behavior, IPv6 routing efficiency, RPL stability, and dynamic role transitions — **without the need for hardware emulation or physical devices**.

The testing framework enables controlled topology scaling and role transitions, allowing for comprehensive analysis while maintaining interoperability within a standardized simulation environment.

Focus

Inspect how the OpenThread mesh network dynamically adapts and maintains stable operations under: - Node role transitions (e.g., Router ↔︎ End Device, Leader elections). - Topology changes (scaling up/down, adding/removing nodes) within an **OTNS-based simulation environment**.

Simultaneously, we evaluate OpenThread's protocol-level performance:

- **IPv6 Handling**: Validate OpenThread's ability to correctly assign, route, and forward IPv6 packets while optimizing overhead through 6LoWPAN header compression.
- **RPL Routing Stability**: Ensure routing decisions remain adaptive and consistent under network dynamics.
- **6LoWPAN Efficiency**: Evaluate how OpenThread compresses and efficiently transmits IPv6 packets over constrained networks.
- **Seamless Communication**: Measure OpenThread's ability to maintain stable network connectivity when nodes join, leave, or change roles dynamically.

Simulated Node Types (OTNS)

Type	Role
FTD	Leader
FTD	Router
FTD	REED (Router-Eligible ED)
FTD	FED (Full End Device)

Simulation Requirements:

- Use **OTNS with OpenThread simulation binaries** (`ot-cli-ftd`).
- Simulate only **FTDs** — No MTDs (Minimal Thread Devices).
- Utilize **REEDs** to allow for dynamic role transitions (Router ↔︎ End Device).
- **Border Router functionality is excluded** from this simulation.

Stages

Step 1: Baseline Measurements

Establish a stable OpenThread network using OTNS before introducing topology changes by evaluating OpenThread's fundamental network behavior and protocol-level stability.

Process (Step 1):

1. Deploy an initial OpenThread network with a fixed number of nodes (Routers, End Devices, and a Leader).
2. Allow the network to stabilize by ensuring:

    - Leader election occurs successfully.
    - Routing tables are fully synchronized across all nodes.
    - IPv6 communication is functional and bidirectional.
3. Monitor and log simulation-level metrics via OTNS and CLI commands.
4. Record baseline performance across all network and protocol metrics to establish a reference for later topology changes and scaling.

Baseline Metrics (Step 1)

#	Inter. Test	Measurement	Description
1	(N, T)	Leader Election Time	Measure the time from network initialization to Leader election.
2	(N, T)	RPL Route Stability & Update Efficiency	Check if all routing tables converge and remain synchronized.
3	(N, T)	Packet Delivery Ratio (PDR) During Steady State	Measure packet loss or delivery success using ping/UDP traffic.
4	(N, T)	Topology Convergence Time	Time taken for routing information to stabilize after initial setup.
5	(N, T)	Multicast Propagation Delay	Measure propagation time of multicast control messages (e.g., DIO, DAO).
6	(N, H)	IPv6 Packet Forwarding Efficiency	Validate proper packet forwarding paths and address usage.
7	(N, H)	6LoWPAN Compression Efficiency	Observe IPv6 compression behavior via CLI or packet size inspection.

Step 2: Topology Scaling

Evaluate OpenThread's ability to dynamically scale the network while maintaining IPv6 routing, RPL consistency, and 6LoWPAN compression efficiency within an OTNS-based simulation.

Process (Step 2):

1. Incrementally add OTNS nodes and measure how quickly they integrate into the mesh network.
2. Remove nodes one by one to observe how OpenThread reconfigures routing and communication paths.
3. Analyze network updates and protocol behaviors:
    - **IPv6**: Ensure newly added nodes receive proper IPv6 addresses and can communicate without disruption.
    - **RPL**: Validate that routes are properly created, updated, and remain stable.
    - **6LoWPAN**: Monitor changes in header compression efficiency as the network grows or shrinks.
    - **Seamless Communication**: Confirm stable message transmission when nodes join or leave the network.

Measurements (Step 2)

#	Inter. Test	Measurement	Description
2	(T, N)	RPL Route Stability & Update Efficiency	Ensure all simulated nodes update their routing tables correctly and adapt to topology changes.
3	(N, T)	Packet Delivery Ratio (PDR) During Changes	Observe any increase in packet loss during node additions/removals.
4	(T, N)	Topology Convergence Time	Measure how long it takes for new nodes to fully integrate.
5	(N, T)	Multicast Propagation Delay	Measure delay in RPL DIO/DAO message propagation after topology changes.
6	(N, H)	IPv6 Packet Forwarding Efficiency	Validate correct packet routing paths and address reachability.
7	(N, H)	6LoWPAN Compression Efficiency	Observe changes in header compression efficiency as node count scales.

Identifying Failure Thresholds Process (Step 2)

#	Inter. Test	Failure Category	Description	Threshold Definition
2	(T, N)	RPL Routing Failure	Routing tables fail to converge after topology changes.	RPL remains unstable beyond **X sec** after a change.
3	(N, T)	Packet Delivery Failure (PDR)	Significant packet loss during node additions/removals.	PDR drops below **X%**.
4	(T, N)	Topology Scaling Breakdown	New nodes fail to integrate or cause excessive routing latency.	Integration exceeds **X sec**, or route churn occurs.
5	(N, T)	Multicast Propagation Delay Failure	RPL messages are delayed, affecting route updates.	DIO/DAO propagation exceeds **X sec**.
6	(N, H)	IPv6 Packet Forwarding Failure	Packets fail to route correctly.	Packet loss or misrouting above acceptable levels.
7	(N, H)	6LoWPAN Compression Failure	Inefficient compression due to scaling.	Compression ratio worsens by more than **X%**.

Step 3: Role Changes

Evaluate how OpenThread maintains network stability, communication consistency, and routing behavior when simulated nodes dynamically change roles (e.g., Leader, Router, REED, End Device) within an OTNS environment.

Process (Step 3):

1. Force a Leader re-election by removing the current Leader node from the network.
2. Demote a Router to an End Device and observe effects on routing and stability.
3. Promote an End Device (REED) to a Router and measure how quickly it integrates and updates routing information.
4. Observe protocol behavior during role transitions:
    - **IPv6**: Ensure nodes retain proper addressing and forwarding capabilities.
    - **RPL**: Validate that routes are recalculated correctly after transitions.
    - **6LoWPAN**: Check if header compression remains efficient after role changes.
    - **Seamless Communication**: Confirm that traffic flows without major disruption during transitions.

Measurements (Step 3)

#	Inter. Test	Measurement	Description
1	(T, N)	Leader Election Time	Measure how quickly the network elects a new Leader after a forced change.
2	(T, N)	RPL Route Stability & Update Efficiency	Ensure routing tables update promptly and correctly after role transitions.
3	(N, T)	Packet Delivery Ratio (PDR) During Role Changes	Check for any traffic loss during node promotions or demotions.
5	(N, T)	Multicast Propagation Delay	Time required for RPL messages (e.g., DIO) to propagate role changes.
6	(N, H)	IPv6 Packet Forwarding Efficiency	Ensure forwarding continues correctly across role transitions.
7	(N, H)	6LoWPAN Compression Efficiency	Evaluate any degradation in compression performance during role changes.

Identifying Failure Thresholds (Step 3)

Determine when OpenThread fails to maintain stable operations during dynamic role transitions in a simulated OTNS environment.

#	Inter. Test	Failure Category	Description	Threshold Definition
1	(T, N)	Leader Election Failure	Leader election takes too long, disrupting overall network operation.	Election time exceeds **X seconds**.
2	(T, N)	RPL Routing Failure	Routing tables fail to update after role changes.	RPL remains unstable beyond **X seconds** after role switch.
3	(N, T)	Packet Delivery Failure (PDR)	Packet loss increases significantly during role transitions.	PDR drops below **X%**.
5	(N, T)	Multicast Propagation Delay Failure	RPL control messages are delayed, affecting routing updates.	Propagation delay exceeds **X seconds**.
6	(N, H)	IPv6 Packet Forwarding Failure	IPv6 forwarding fails due to improper route recalculations.	Packet misrouting or delivery failure exceeds acceptable limits.
7	(N, H)	6LoWPAN Compression Failure	Compression becomes inefficient after role changes.	Compression ratio worsens by more than **X%**.