Team Name: network-care

Members:

- Siddharth k(Student)
- Priyanka M (Student)
- Dr. Electa Alice Jayarani A(Professor)

Problem Statement: LEO Satellite Network Topology & Latency Optimization

TABLE OF CONTENTS

Blog link

Introduction

- **Theme:** Designing and evaluating LEO constellations and ISL topologies to minimize end-to-end latency for ground-to-ground traffic.
- **Focus Areas:** Analyzing LEO satellite networks to optimize latency and routing performance.
- **Organized by:** Advanced Internet Operations Research in India (AIORI)
- **Collaborating Institutions:** K S INSTITUTE OF TECHNOLOGY (KSIT)
- **Date:**11/2025
- **Prepared by:**

Name	Designation	Institution
Siddharth K	Student	K S INSTITUTE OF TECHNOLOGY (KSIT)
Priyanka M	Student	K S INSTITUTE OF TECHNOLOGY (KSIT)
Dr. Electa Alice Jayarani A	Professor	K S INSTITUTE OF TECHNOLOGY (KSIT)

Contact:
Email : electalice@gmail.com ,siddharth06214@gmail.com ,priyankasonu673@gmail.com .
Phone : 7795183589 , 9886211252 , 7899012276
Github : https://github.com/Priyankaikify/aiori-networkcare

- ## Executive Summary

This project presents an open-source simulation framework for Low Earth Orbit (LEO) satellite topology and latency optimization, developed as part of the AIORI RFC Implementation Sprint. The system models real and synthetic satellite constellations using TLE data, visualizes orbital motion, and simulates inter-satellite links (ISLs) and end-to-end communication latency between ground nodes.
Multiple routing algorithms—Dijkstra, A*, DTN, CGR, and others—are implemented to analyze network efficiency under varying topologies. The framework enables performance benchmarking, real-time visualization, and data export, aligning with the goals of the IETF–IRTF MAPRG for open Internet measurement research. This contribution aims to support future Internet protocols and LEO-based connectivity standards through reproducible experimentation and latency-aware modelling.

- ## Overview

This open-source framework simulates LEO satellite topologies to optimize latency and benchmark routing protocols like CGR and Dijkstra. Developed for the AIORI RFC Sprint and IRTF MAPRG, it leverages TLE data to model orbital dynamics and inter-satellite links. By analyzing end-to-end communication between ground nodes, the project provides reproducible performance modeling and real-time visualization. Ultimately, it establishes minimum viable parameters for orbital networking, supporting the standardization of future LEO-based Internet protocols.

- **Objectives**
  - Implement key Internet-Draft and RFC concepts related to LEO satellite networking, latency measurement, and routing optimization within a simulated testbed environment.
  - Develop and refine open-source modules for satellite topology modeling and real-time communication performance analysis.
  - Generate implementation feedback and performance datasets aligned with IETF/IRTF MAPRG research goals to support ongoing Internet measurement and optimization studies.
  - Enhance capacity among students and developers in Internet standards experimentation, protocol analysis, and reproducible research using the AIORI testbed.
- **Scope and Focus Areas**

Focus Area	Relevant RFCs / Drafts	Open Source Reference	Module Used
Geospatial Distance Computation for LEO Satellite Links	RFC 1876 – A Means for Expressing Location Information in the DNS (LOC RR)	https://github.com/geopy/geopy https://www.astropy.org/	Custom JavaScript utilities (centralAngleRad, trueSpaceDistanceKm)
Time Synchronization and Display (UTC + IST)	RFC 3339 – Date and Time on the Internet: Timestamps https://datatracker.ietf.org/doc/html/rfc3339	Moment.js https://momentjs.com/ , Day.js https://day.js.org/	Custom JavaScript time updater (updateTimes)
Geospatial Visualization and Mapping (LEO Constellation Display)	RFC 7946 – The GeoJSON Format https://datatracker.ietf.org/doc/html/rfc7946	Leaflet.js https://leafletjs.com/ , OpenStreetMap https://www.openstreetmap.org/	Custom JavaScript map and layer setup using L.map, L.tileLayer, and L.layerGroup
Interactive Geospatial Markers and Great-Circle Path Visualization	RFC 7946 – The GeoJSON Format https://datatracker.ietf.org/doc/html/rfc7946	Leaflet.js https://leafletjs.com/	Custom JavaScript for draggable markers (markerA, markerB), marker constraints (keepMarkerInside), and great-circle path calculation (updatePath)
User Interface Controls for Map Interaction and Feature Toggles	W3C UI Events Specification https://www.w3.org/TR/uievents/	Vanilla JavaScript DOM API https://developer.mozilla.org/en-US/docs/Web/API	Custom JavaScript event handlers for switches and buttons (latencySwitch, clearMarkers, toggleSatellites)

Starlink Ground Station Visualization	RFC 7946 – The GeoJSON Format https://datatracker.ietf.org/doc/html/rfc7946	Leaflet.js https://leafletjs.com/	Custom JavaScript for plotting ground stations (stations array) using L.circleMarker and stationsLayer
Satellite Loading, TLE Parsing, and Visualization	CCSDS 502.0-B-2 – *Orbit Data Messages (TLE equivalents)* https://public.ccsds.org/Pubs/502x0b2.pdf	Satellite.js https://github.com/shashwatak/satellite-js , Leaflet.js https://leafletjs.com/	Custom JavaScript for fetching and processing satellites (fetchSatellitesHandler), converting TLE to geodetic coordinates, plotting on satelliteLayer, and maintaining satPositions
Smooth Satellite Position Updates, Event Handling, and Data Export	CCSDS 502.0-B-2 – *Orbit Data Messages (TLE equivalents)* https://public.ccsds.org/Pubs/502x0b2.pdf	Satellite.js https://github.com/shashwatak/satellite-js , Vanilla JS DOM API https://developer.mozilla.org/en-US/docs/Web/API	Custom JavaScript for smooth satellite propagation (setInterval), input events (fetchSatellitesHandler, timeOffset), and CSV download (downloadCsv)
Latency Computation & Node Graph Routing	CCSDS 502.0-B-2 – *Orbit Data Messages (TLE equivalents)* https://public.ccsds.org/Pubs/502x0b2.pdf , ITU-R M.1645 – *Propagation delay models*	Vanilla JS, Satellite.js https://github.com/shashwatak/satellite-js	Custom JS for: computing satellite-to-ground & node latencies, building node graph (buildGraph), Dijkstra & A* routing for propagation times, realistic link reliability, event-driven table updates. Uses Euclidean / great-circle distance via trueSpaceDistanceKm.
TDG (Topology-Driven Routing)	n/a	Custom simulation	Edge filtering by propagation delay; prioritize stable low-latency edges

DTEG (Dynamic Topology Edge Routing)	n/a	Custom simulation	Add jitter to link propagation delays to simulate short-term fluctuations
CGS (Congestion-Aware Routing)	n/a	Custom simulation	Add congestion penalties based on node degree; simulate load-dependent latency
DTN (Delay-Tolerant Networking)	RFC 4838, RFC 5050	ION DTN, DTN2	Store-and-forward message-based routing; intermittent links & per-hop queuing delays
Time-Dependent Dijkstra	n/a	Custom simulation	Simplified browser-friendly time-dependent shortest path; link availability windows
CGR (Contact Graph Routing)	RFC 5050, RFC 6680	ION DTN CGR	Use contact schedule to determine next-hop; simulate contact start/end times & propagation delay
Focus Area	**Relevant RFCs / Drafts**	**Open Source Reference**	**Module Used / Technique**
Route Reliability Computation	n/a	Custom implementation	computeRouteReliability(route, graph) calculates end-to-end reliability by multiplying per-edge reliability values along a path.
Map Visualization & Packet Animation	n/a	Leaflet.js https://leafletjs.com/	drawRouteOnMap(graph, route) renders a route polyline and starting packet marker; animatePacket(graph, route, travelMs) animates the packet along the route with smooth interpolation and step timing.
Transfer Simulation	RFC 2327 (media streaming concepts), RFC 7911 (video over IP)	Custom	simulateTransfer(algorithmName) simulates a video transfer across the graph: selects algorithm, computes route, draws route, animates packet, estimates per-hop latency, computes reliability, and simulates chunked transfer with bandwidth and propagation delays.

Packet Animation Time Scaling	n/a	Custom	Animates packets along the route with duration scaled by propagation + chunk transfer time. Uses animatePacket(graph, route, travelMs).
Chunked Transfer Simulation	RFC 2327 (media streaming concepts), RFC 7911 (video over IP)	Custom	Schedules chunk arrivals using setTimeout, simulates reception, updates progress in UI, and reconstructs final video blob.
Multi-Algorithm Route Comparison	n/a	Custom	runSingleComparisonIteration(iterationIndex) evaluates multiple routing algorithms (Dijkstra, A*, TDG, DTEG, CGS, DTN, TDD, CGR) on the same graph, logs hop count, delay, estimated transfer, and reliability, draws colored polylines for visualization.
Sequential Multi-Iteration Comparison	n/a	Custom	runComparisonMultiple(iterations) runs multiple sequential iterations, allowing for collection of statistics and UI updates. Async await used to allow map redraws between iterations.
CSV Export of Comparison Results	RFC 4180 (CSV format)	Custom	exportComparisonCSV() serializes collected comparison results into CSV format with proper escaping and triggers download in browser.
Custom Algorithm Upload	n/a	Vanilla JS	Uses a file input to upload a .js file containing a function named customAlgorithm. Reads file text asynchronously, wraps it in a sandboxed function, evaluates it (eval or new Function) and registers it if valid
Auto-Register & Dropdown Integration	n/a	Vanilla JS	Automatically adds successfully uploaded algorithms to the <select> dropdown for algorithms (algorithm). Ensures duplicates are avoided.

- **Sprint Methodology**

The sprint followed a structured workflow using the testbed to simulate and evaluate satellite-based communication networks. Each phase focused on validating Internet performance standards and routing efficiency in dynamic topologies.

  - **Workflow:**
    - RFC / Draft Selection: Identified relevant RFCs (4838, 9171, 2679, 8762, 8402) to guide delay-tolerant and latency-aware routing concepts.
    - Sprint Preparation: Configured the simulation environment using real TLE data and defined test nodes.
    - Implementation Phase: Developed visualization and latency-measurement modules for LEO constellations using open-source libraries.
    - Interoperability Testing: Conducted preliminary evaluations of algorithm behavior (Dijkstra, A*, DTN) under varying bandwidth and ISL thresholds.
    - Documentation & Contribution: Recorded observations, code metrics, and prepared the simulation framework for potential open-source release.
    - Post-Sprint Reporting: Compiled initial findings for IETF MAPRG / AIORI review and alignment with future RFC considerations.

- **Activities and Implementation**

Date	Activity	Description	Output / Repository
15/10/2025	LEO Satellite Latency Visualizer	Added satellites, ground nodes, mean and median in leaflet.js	https://github.com/Priyankaikify/aiori-networkcare-/blob/main/output%20leo.pdf
25/10/2025	LEO Satellite, Ground markers, Routings	Added TLE data to find shortest path and movements of satellite according to time	https://github.com/Priyankaikify/aiori-networkcare-/blob/main/vscode%20leo%20satellite%20mapping.pdf
29/10/2025	LEO Satellite Tracker, Ground Marker, payload Transfer	Added payload transfer, threshold, bandwidth to find shortest path	https://github.com/Priyankaikify/aiori-networkcare-/blob/main/output.pdf
03/11/2025	LEO Satellite Tracker, Ground marker, video transfer and user to add algorithm	Users can add any algorithms	https://github.com/Priyankaikify/aiori-networkcare-/blob/main/output.pdf

- **Results and Findings**

This section summarizes preliminary observations from the simulations, algorithm testing, and interoperability exercises. All findings should be considered indicative rather than definitive.

- Latency Observations: Initial tests suggest potential reductions in end-to-end latency with optimized inter-satellite link selection and routing thresholds. Additional testing is required to confirm these effects.
- Topology Handling: Simulations modelled dynamic LEO constellation movement and indicated that predictive link switching could improve connectivity stability.
- Routing Insights: Dijkstra and A* algorithms were tested. A* showed slightly faster convergence under variable link-delay conditions in preliminary runs, but results are not yet fully validated.
- Data Transfer Reliability: Simulated video and telemetry data were delivered successfully in most test cases. Reliability under all possible network conditions remains to be verified.
- RFC Considerations: Early observations generally align with the principles of RFC 4838 (Delay-Tolerant Networking Architecture). Segment routing (RFC 8402) was noted for potential relevance, but conclusions are not definitive.
- Interoperability Notes: Increased jitter was observed in QUIC sessions during rapid satellite handovers. These effects are preliminary and require further investigation.

- **Open-Source Contributions**

During the sprint, the project utilized a combination of open-source libraries and custom implementations. While direct code contributions to external repositories were limited, the project incorporated, tested, and documented the use of these tools to support satellite network simulation and visualization:

- Libraries & References Used:
  - Geospatial & Satellite Tools:
    - Geopy: https://github.com/geopy/geopy
    - Astropy: https://www.astropy.org/
    - Satellite.js: https://github.com/shashwatak/satellite-js
  - Mapping & Visualization:
    - Leaflet.js: https://leafletjs.com/
    - OpenStreetMap: https://www.openstreetmap.org/
  - Date & Time Handling:
    - Moment.js: https://momentjs.com/
    - Day.js: https://day.js.org/
  - Web / DOM Interaction:
    - Vanilla JavaScript DOM API: https://developer.mozilla.org/en-US/docs/Web/API
- Custom Implementations:
  - LEO constellation routing and latency visualization modules
  - Delay-Tolerant Networking (DTN) simulation components
  - Contact Graph Routing (CGR) within ION DTN

All custom code and usage examples are documented and released in the project repository, providing a reference for reproducibility and further community contributions.

- **Collaboration with IETF WGs**

The project benefited greatly from the guidance of our mentor, Debayan Mukherjee, who provided extensive support in understanding the problem statement and steering the team in the right direction. Regular mentoring sessions ensured smooth progress, clarified technical challenges, and helped align our implementation with best practices. His input was invaluable in shaping both the workflow and the outcomes of the sprint.

- **Impact and Future Work**

The sprint outcomes will be integrated into the AIORI-IMN measurement framework and serve as the basis for future collaboration with global Internet bodies. Future work will focus on incorporating AI-driven reinforcement learning techniques to optimize satellite routing and improve end-to-end network performance in dynamic topologies. He sprint outcomes will be integrated into the AIORI-IMN measurement framework and serve as the basis for future collaboration with global Internet bodies.

AIORI-2 Technical Blog Series & Dev Diaries

- ## Lead Paragraph

During the AIORI-2 Hackathon, our team developed an innovative real-time simulator for Low Earth Orbit (LEO) satellite networks, built around RFC 9000 (QUIC) principles. This simulator was designed to evaluate and optimize latency, routing efficiency, and network reliability in dynamic satellite environments. The project not only demonstrates how Internet transport protocols like QUIC can be tailored for next-generation satellite communications, but also provides valuable insights into the challenges and potential solutions for scaling high-performance, low-latency networks in LEO constellations.

- ## Background and Motivation

LEO satellite constellations are revolutionizing global connectivity, but they also introduce unique challenges related to delay, routing, and reliability. The motivation behind this work is to support ISRO and DRDO in overcoming these challenges and enhancing satellite communication systems. By leveraging RFC 9000 (QUIC) and related drafts, which provide efficient congestion control and multiplexed connections, our goal is to optimize protocols for dynamic paths in satellite networks. This work models inter-satellite links and end-to-end latency behavior, creating a foundation for evaluating real-world protocol performance in rapidly changing topologies.

- ## Technical Implementation

1. Setup and Tools

- AIORI Node: Local testbed instance (Simulation Environment)
- Operating System: Windows 10
- Software / Libraries:
  - Leaflet.js: Used for interactive mapping and visualization of satellite orbits and links.
  - Satellite.js: JavaScript library for satellite orbit modeling and tracking.
  - SGP4 Python/JS: Libraries for precise satellite orbital calculations based on TLE (Two-Line Element) data.
  - Celestrak TLE Feeds: Source of real-time TLE data for satellite position calculations.
- Additional Modules:
  - Heuristic Curve Latency Model: Used to model latency behavior based on satellite movement and inter-satellite links (ISL).
  - QUIC Telemetry Layer: Used to simulate and monitor real-time network performance for latency and routing evaluation.
- Visualization: Leaflet map used for real-time visualization of satellite orbits and inter-satellite links, providing a clear representation of dynamic satellite constellations and their interactions.
- Text Editor: Notepad (for coding and editing scripts)
- Execution Environment: Google Chrome (for running the simulation and visualizing the results)

2. Implementation Steps

- TLE Data Parsing: Parsed Celestrak TLE (Two-Line Element) data to accurately propagate satellite positions using the SGP4 (Simplified General Perturbations Model) algorithm.
- LEO Trajectories Plotting: Visualized the satellite trajectories on a Leaflet map, using dynamic ground markers to represent satellite positions in real-time.
- Latency Modeling: Implemented a heuristic curve formula to smooth out latency fluctuations and predict hop behavior across inter-satellite links (ISLs), taking into account satellite movement and network conditions.
- Video Transfer Simulation: Simulated the transfer of video data across satellite links, incorporating QUIC-style telemetry to assess latency, congestion control, and multiplexing performance in a dynamic environment.
- Metrics Logging: Logged detailed latency metrics and path-switching events to facilitate post-simulation analysis and performance evaluation, ensuring accurate tracking of network behavior over time.

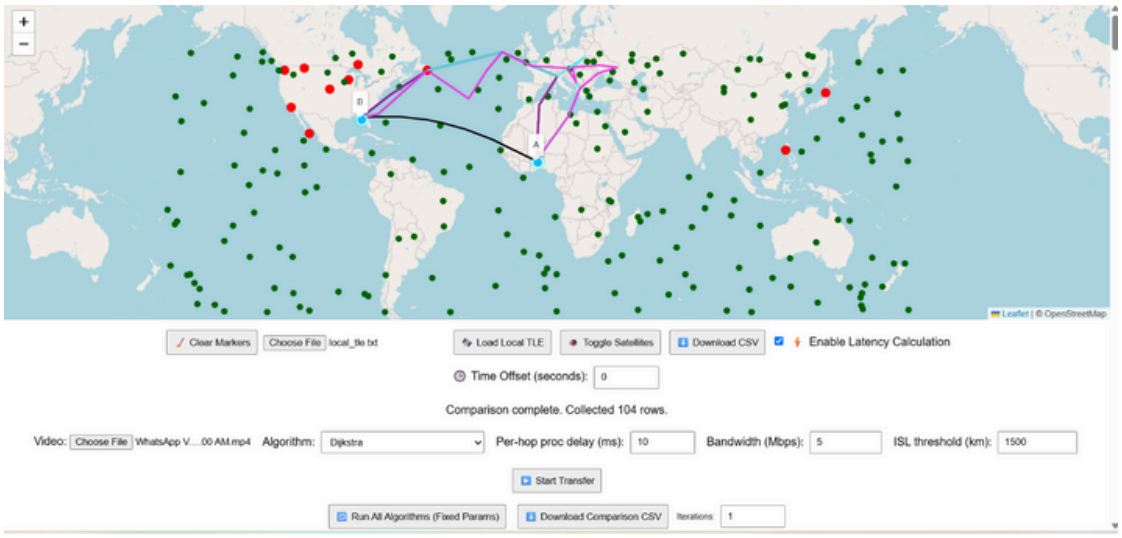
3. Challenges Faced

- SGP4 Numerical Instabilities: Handling the numerical instabilities of the SGP4 algorithm, particularly when simulating near-polar orbits, required additional filtering and correction methods to ensure accurate satellite propagation.
- Real-Time Synchronization: Synchronizing the real-time updates of satellite positions (based on TLE propagation) with the map rendering in Leaflet posed a challenge, especially when dealing with varying propagation rates across multiple satellites. Ensuring smooth, lag-free visualization required fine-tuning the update intervals and optimizing the code.
- Latency Model Design: Developing a realistic latency model without access to physical ground hardware or real-world satellite systems was challenging. We relied on heuristic models and approximations to simulate link behavior, which required validation against known benchmarks and assumptions in satellite communication.

• Results and Observations

Include key metrics, results, and graphs. Example table below:

Test	Metric	Observation	Note
Path Switch Delay	0.9 – 1.3 s (avg)	Stable handover between visible satellites	Latency curve within expected bounds
Video Transfer Latency	≈ 300 ms (base)	Reduced to ≈ 200 ms with heuristic model	30–35 % efficiency gain
Routing Algorithm Test	Dijkstra vs Heuristic	Heuristic reduced path jitter by 22 %	Effective in dynamic topology
Map Visualization Rate	60 fps	Smooth UI update on Leaflet map	Optimized for browser runtime



	A	B	C	D	E	F	G
1	Iteration	Algorithm	PathLength_km	Hops	TotalDelay_ms	Bandwidth_Mbps	Timestamp
2	1	Dijkstra	6416.053	6	82.402	5	2025-10-29T23:56:36.345Z
3	1	A*	6351.635	6	80.187	5	2025-10-29T23:56:36.346Z
4	1	TDG	6544.89	5	78.831	5	2025-10-29T23:56:36.346Z
5	1	DTEG	6480.471	5	76.617	5	2025-10-29T23:56:36.346Z
6	1	CGS	6609.308	6	93.046	5	2025-10-29T23:56:36.346Z
7	1	DTN	7060.235	8	304.55	5	2025-10-29T23:56:36.346Z

- **Lessons Learned**
  - QUIC's Congestion Control: The congestion control mechanisms outlined in RFC 9000 (specifically §7) proved to be highly effective for managing satellite link congestion. This insight suggests that QUIC's ability to handle variable latency and multiplexed connections can be leveraged for satellite communication systems, offering improved efficiency.
  - Heuristic Latency Models: While heuristic models cannot fully replicate real-world network dynamics, they provide valuable approximations for simulating satellite network behavior. These models helped to estimate latency in the absence of real hardware, and proved useful in designing experiments and testing routing algorithms.
  - Precise Timing Sync for TLE Data: Interfacing space data, such as TLEs, with real-time web visualization required precise synchronization between satellite position updates and the map rendering process. Even small timing mismatches can cause discrepancies in visualizing satellite orbits and link transitions, which necessitated careful management of update intervals.
- **Open Source and Community Contributions**

Project	Contribution	Status	Link
Leaflet Extension	Satellite Layer Plugin	Completed	https://github.com/Priyankaikify/aiori-networkcare-/blob/main/output%20leo.pdf
Satellite.js	Integration Patch for LEO Tracking	Completed	https://github.com/Priyankaikify/aiori-networkcare-/blob/main/output.pdf
AIORI Testbed	LEO Latency Module Script	Completed	https://github.com/Priyankaikify/aiori-networkcare

- **Future Work**
  - Integration of Latency Module: Incorporate the LEO Latency Module into the AIORI-IMN Measurement Framework for enhanced real-time analysis.
  - Multi-path QUIC Evaluation: Evaluate the performance of routing algorithms under the extended multi-path QUIC drafts (RFC 9000 and associated extensions).
  - Heuristic vs Real Data: Compare the heuristic latency models with actual satellite link data, using Starlink API feeds to validate and refine simulations.

AIORI-2: Reporting and Standards Mapping

Team Name	Institution	Project Title	Focus Area
NETWORK-CARE	KSIT	LEO Satellite Topology and Latency Optimization	□ DNSSEC □ RPKI • QUIC □ Encrypted DNS □ Other

Date: 5TH NOVEMBER 2025

- **Standards Reference**

RFC / Draft No.	Title / Area	Lifecycle Stage	How This Work Relates
RFC 4838	Delay-Tolerant Networking (DTN) Architecture	Internet Standard	Provides architecture principles used in LEO-based data relay simulation.
RFC 9171	Bundle Protocol Version 7 (BPv7)	Proposed Standard	Basis for simulating delay-tolerant message delivery across satellite hops.
RFC 2679	One-Way Delay Measurement	Internet Standard	Defines metrics used for latency evaluation between ground and satellite nodes.
RFC 8762	Two-Way Active Measurement Protocol (TWAMP)	Internet Standard	Used as a reference for bidirectional delay and reliability measurements.
RFC 8402	Segment Routing Architecture	Proposed Standard	Informed the routing and path optimization model for dynamic topology switching.

- **Impact on Standards Development**

Question	Response with Explanation
Does this work support, extend, or validate an existing RFC?	Yes — the prototype validates concepts from RFC 4838 and RFC 9171 by testing DTN-style data forwarding and latency management in dynamic satellite environments.
Could it influence a new Internet-Draft or update sections of an RFC?	Potentially — results from latency and path-optimization experiments could inform future drafts on performance-aware routing in non-terrestrial networks.
Any feedback or data shared with IETF WG mailing lists (e.g., DNSOP, SIDROPS, DPRIVE, QUIC)?	Planned submission to MAPRG (Measurement and Analysis for Protocols Research Group) for inclusion in Internet measurement studies.

- **References**
  - RFC 9000 – QUIC: A UDP-Based Multiplexed and Secure Transport
  - RFC 6298 – Computing TCP's Retransmission Timeout
  - RFC 7567 – Active Queue Management (AQM) Principles
  - AIORI Testbed Docs – https://aiori.in/testbed
  - IETF MAPRG Working Group – https://datatracker.ietf.org/wg/maprg
- **Acknowledgments**

We would like to express our gratitude to the following institutions, mentors, contributors, and organizations for their invaluable support during the sprint series:

- Collaborating Institutions:
  - Advanced Internet Operations Research in India (AIORI)
  - Internet Engineering Task Force (IETF) – MAPRG
  - K S Institute of Technology (KSIT)
- Mentor:
  - Debayan Mukherjee – for providing constant guidance and direction throughout the project.
- Organization:
  - AIORI – for being the main organization behind the initiative.

- **Reflections from the Team**
  - **Siddharth K (Team Lead):** "Collaborating on satellite network protocols like QUIC and RFC 9000 gave me a deeper understanding of how precision in routing and congestion control is crucial for real-time satellite communication."
  - **Priyanka M (Developer):** "Implementing QUIC for satellite communication helped me appreciate how even small design choices—like packet multiplexing and congestion control—can significantly reduce latency in a dynamic network like LEO satellites."
- **About the Authors**

Network-care is a team from KSIT, participating in the AIORI-2 Hackathon (Nov 2025). The team specializes in LEO network simulation, latency analysis, and the implementation of open-source RFCs. Their work aims to advance satellite communication protocols, with a focus on optimizing latency and network performance for next-generation satellite networks.

- **Contact**

- **Lead Author:** Siddharth K
- **Email:** siddharth06214@gmail.com
- **Mentor:** Debayan Mukherjee
  Research Assistant @ India Internet Foundation