Team Name: **SYN/ACK**

Members:
- **Aman Kumar(Student)**
- **Pragyensh Pritiman Panda (Student)**
- **Dr. Tejaswi Khanna(Professor)**

Problem Statement: **Modular Internet Measurement Network over NATS**

TABLE OF CONTENTS

Blog link

Introduction

- **Theme:** Implementation and Testing of Selected Internet-Drafts / RFCs using AIORI Testbed
- **Focus Areas:** Modular Internet Measurement Architecture
- **Organized by:** Advanced Internet Operations Research in India (AIORI)
- **Collaborating Institutions:** Amity University Greater Noida
- **Date:**11/2025
- **Prepared by:**

Name	Designation	Institution
Aman Kumar	Student	Amity University Greater Noida
Pragyensh Pritiman Panda	Student	Amity University Greater Noida
Dr. Tejaswi Khanna	Professor	Amity University Greater Noida

Contact: ambxr2005@gmail.com(+91 9955672444)

Executive Summary

The **"Modular Internet Measurement Network over NATS"** is a decoupled, fault-tolerant architecture for Internet measurement tasks. It uses NATS.io to separate the management plane (Anchor) from measurement modules (latency-ping, DNS-lookup). This design enables hot-plugging, true fault isolation (crashing modules don't affect the system), and load balancing for horizontal scaling.

Overview

The "Modular Internet Measurement Network over NATS" is a cutting-edge, decoupled, and fault-tolerant architecture developed during the AIORI implementation sprint by Amity University. Its core objective was to modernize Internet measurement systems by addressing the resilience and scalability limitations inherent in traditional monolithic designs.
The solution centers on using NATS.io as a high-performance, asynchronous message bus to fundamentally separate the Anchor service (the management plane) from autonomous measurement modules (the data plane), such as latency-ping and DNS-lookup.
Key Technical Achievements:

- **Resilience:** Demonstrated true fault isolation, ensuring module crashes do not affect the core Anchor service or other running modules.
- **Scalability:** Achieved horizontal scaling with automatic load balancing handled efficiently via NATS queue groups.
- **Flexibility:** Implemented hot-plug capability, allowing new modules to be dynamically discovered and integrated within 2-3 seconds without requiring a system restart.
- **Management:** Built a functional management UI/CLI for real-time job submission and result streaming.

The project followed a structured agile methodology across four sprints, validating its design against standards like **RFC 2330** (IPPM Framework). The work serves as a reference implementation for cloud-native measurement architectures and yielded significant open-source contributions, including the core framework and standardized message schemas. Future plans include integrating this framework with AIORI's main testbed and formalizing the NATS-based architectural patterns as a potential Internet-Draft for the IETF IPPM Working Group.

RFC-Open Source Contribution Report

- **Categorization :**

The project established the Modular Internet Measurement Network over NATS, a decoupled, fault-tolerant architecture separating the management plane (Anchor) from measurement modules via NATS.io. This design enables hot-plugging, true fault isolation, and horizontal scaling. Key deliverables included the core Anchor service, independent measurement modules (latency-ping, DNS-lookup), and a management UI/CLI.

- **Scope and Focus Areas :**

Focus Area	Relevant RFCs / Drafts	Open-Source Reference	AIORI Module Used
Message-Based Architecture	RFC 2330 (Framework)	NATS Protocol NATS.io	Docker, Go/Node.js AIORI Transport Module
Internet Measurement Protocols	RFC 4656 (OWAMP)	RFC 5357 (TWAMP)	RFC 2681 (Delay) Custom Ping/DNS modules AIORI Network Testbed
Security & Authentication	RFC 7519 (JWT)	RFC 8446 (TLS 1.3) NATS JWT auth	TLS libraries AIORI Security Framework
Data Formats & APIs	RFC 8259 (JSON)	RFC 8949 (CBOR)	WebSockets AIORI Data Processing

The project scope covered a wide array of Internet measurement standards and modern architecture patterns, ensuring technical depth and interoperability.

- **Sprint Methodology Workflow :**

The project followed a structured agile workflow, leveraging the AIORI testbed infrastructure and open-source tools across distinct phases.

  - **RFC / Draft Selection & Architecture Design:** Analyzed RFC 2330, RFC 4656, and RFC 5357 specifications. Designed NATS-based message schema, subject hierarchy, and contract specifications for decoupled communication.
  - **Sprint Preparation & Environment Setup:** Established a NATS.io cluster with JetStream persistence. Configured a Docker-based multi-container development environment and prepared AIORI testbed nodes.
  - **Implementation Phase -** Core Infrastructure: Built the Anchor Service with dynamic module discovery and health monitoring. Implemented JWT authentication and TLS encryption for the NATS security layer.
  - **Implementation Phase - Measurement Modules:** Developed Latency-Ping and DNS-Lookup modules. Incorporated graceful shutdown, retry policies, and circuit breaker patterns for module lifecycle management.
  - **Implementation Phase - Management Plane:** Created the Web UI (real-time dashboard via WebSockets), CLI interface for job submission, and REST API endpoints for system control.
  - Interoperability Testing & Validation: Conducted rigorous tests verifying hot-plugging, fault injection (module stability during failure), horizontal scaling (load balancing), and backpressure mechanisms.
  - **Documentation & Contribution:** Created comprehensive API documentation, deployment guides for Docker/Kubernetes, and published performance benchmarks

- **Activities and Implementation Timeline**

Date	Activity	Description	Output / Repository
24/09/2024	**Sprint 1: Anchor Core**	Implemented Anchor service with module discovery, health tracking, and NATS integration	https://github.com/synack2025/AIORI-2-Hackathon (since, it's a private directory , only accessible using the github id which has been added as the collaborator)
01/10/2024	**Sprint 2: Measurement Modules**	Developed latency-ping and DNS-lookup modules with independent NATS clients and job processing	
15/10/2024	**Sprint 3: Management Plane**	Built Web UI/CLI for job submission, real-time result streaming via WebSockets, and module management	
24/10/2024	**Sprint 4: Validation & Scaling**	**Tested hot-plugging, fault isolation, load balancing, and backpressure mechanisms**	

- **Open-Source Contributions :**

The project generated a wealth of open-source components and documentation intended for community use and adoption:

- **Core Repository:** modular-internet-measurement - Complete NATS-based measurement framework, including the Anchor service with dynamic discovery, reference modules (latency-ping, DNS-lookup, HTTP-probe), and the Management UI/dashboard. Docker/Kubernetes deployment configurations are included.
- **Developer Tools & Schemas:** nats-measurement-schemas - Standardized message contracts, versioned JSON schemas, and Protocol Buffer definitions.
- **Documentation & Guides:** Architecture Guidelines (NATS-based decoupled patterns), a Module Development Cookbook, and Production Deployment Runbooks.
- **Community Impact:** Released a Module Development Kit (templates, testing utilities) and Integration Adapters (Prometheus exporter, Grafana dashboards) to build a contributing ecosystem.

Collaboration with IETF WGs and Project Impact :

The architectural insights generated are relevant for immediate feedback to global standardization efforts:

- **IPPM WG (IP Performance Metrics):** The modular architecture provides practical implementation insights for Internet measurement frameworks (RFC 2330), demonstrating how decoupled designs using NATS enhance system resilience and scalability over traditional coordination methods.
- **Potential for NETMOD WG:** The patterns for dynamic module discovery and management can inform model-driven management approaches for distributed measurement systems.
- **Immediate Impact:** Provides a validated reference implementation for modular, resilient Internet measurement architectures and a readily deployable solution for distributed network monitoring within the AIORI testbed.
- **Future Work Directions:** Integrate the framework with the main AIORI-IMN testbed; develop advanced measurement modules (e.g., QUIC performance, BGP monitoring); and Formalize the NATS subject schema and message formats as a potential Internet-Draft for the IETF community.

Results and Findings :

Key Technical Insights:

- Successfully demonstrated hot-plugging and achieved true fault isolation—module crashes did not affect the system's stability.
- Validated horizontal scaling as multiple module instances were automatically load-balanced via NATS queue groups.
- Implemented effective backpressure by having the Anchor throttle job submission to saturated modules.
- Performance outcomes showed module discovery completed within 2-3 seconds, and job processing latency was under 50ms.

Future Work Directions :

- Integrate with AIORI-IMN measurement framework as scalable measurement nodes
- Develop additional measurement modules (QUIC performance, BGP monitoring, RPKI validation)
- Enhance security with more granular NATS permissions and authentication
- Explore federation across multiple NATS clusters for geographic distribution
- Formalize the NATS subject schema and message formats as potential Internet-Draft

Technical Blog Series & Dev Diaries

- ## Technical Implementation :

1. Setup and Tools:

- Development Environment:
- Primary Platform: [Your OS - Ubuntu/Windows/macOS]
- Container Environment: Docker with Docker Compose
- Programming Stack: [Your languages - Go/Node.js/Python]

- **Core Infrastructure:**
  - Message Broker: NATS.io server with JetStream persistence
  - Service Discovery: File system watcher with NATS announcements
  - Container Orchestration: Docker Compose for multi-service management

- **Measurement & Analysis Tools:**
  - Module Framework: Custom NATS client wrappers with health monitoring
  - Job Management: Versioned JSON schemas for measurement jobs
  - Real-time Monitoring: WebSocket dashboard for live result streaming

- **Key Software Components:**
  - Anchor Service: Module discovery and health management
  - Measurement Modules: Latency-ping, DNS-lookup, HTTP-probe
  - Management Interface: Web UI and CLI for system control
  - Security Layer: JWT authentication and TLS encryption

2. Implementation Steps:

- **NATS-based Architecture Design**
  - Designed subject hierarchy for job distribution and result collection
  - Implemented versioned JSON schemas for job requests and responses
  - Established module discovery protocol using mmn.modules.announce subject
  - Created health monitoring system with heartbeat mechanisms

- **Anchor Service Development**
  - Built dynamic module loader with file system watcher integration
  - Implemented health tracking with circuit-breaker patterns
  - Developed job scheduling with backpressure awareness
  - Added graceful shutdown handling for clean module termination

- **Measurement Module Implementation**
  - Created latency-ping module implementing ICMP-based measurements
  - Developed DNS-lookup module with configurable record type support
  - Built module template with standardized NATS client integration
  - Implemented retry policies and error handling for network fluctuations

- **Management Plane Integration**
  - Developed Web UI with real-time WebSocket result streaming
  - Built CLI tool for job submission and module management
  - Implemented REST APIs for system monitoring and control
  - Created visualization components for measurement results

- ## Challenges Faced :

  - Module Discovery Timing: Initially faced race conditions where modules would announce before Anchor was ready, requiring heartbeat-based discovery with retry mechanisms.
  - Message Schema Versioning: Early schema changes broke compatibility, leading to implementation of versioned subjects and backward-compatible parsing.
  - WebSocket Connection Management: Real-time UI updates required careful connection pooling and reconnection logic for reliable data streaming.
  - Health Monitoring Accuracy: Distinguishing between temporary network issues and actual module failures needed sophisticated heartbeat patterns with configurable timeouts.
  - Load Balancing Coordination: Ensuring fair job distribution across multiple module instances required careful queue group configuration and workload monitoring.

- **Results and Observations :**

Test	Metric	Observation	Note
Hot plugging	2-3 seconds	Module discovery and registration successful	Automatic without Anchor restart
Fault Isolation	0% impact	Other modules continue working normally	Crashed module doesn't affect system
Load Balancing	Even distribution	Work shared across multiple instances	NATS queue groups automatically balance
Backpressure	Graceful degradation	Anchor throttles job submission	Prevents module overload
Measurement Latency	< 50ms	Local measurement performance	Varies by measurement type

- **Lessons Learned :**
  - **Decoupled Architecture Power:** Discovered that a message-based, decoupled design (using NATS) provides true fault isolation and enables hot-plugging, proving that loose coupling is superior to monolithic systems for resilience and scalability.
  - **NATS.io in Practice:** Learned that NATS.io subjects and queue groups offer a remarkably simple yet powerful way to implement load balancing and pub-sub patterns, but they require careful subject hierarchy design from the start.
  - **State Management Complexity:** Realized that managing the state and health of dynamic, independent modules is more challenging than in a monolithic system, necessitating robust heartbeat and discovery mechanisms.
  - **Tool Integration:** Found that integrating the NATS ecosystem with our measurement modules and management UI required careful configuration and a clear understanding of connection lifecycles and error handling in a distributed context.
  - **Development Workflow Value:** Experienced how a microservices-like approach allowed our team to work on the Anchor, modules, and UI in parallel, significantly speeding up development and testing.

- ## Reflections from the Team :
- **Pragyensh Pritiman Panda (Research & Presentation Lead):**
    - "This hackathon was a deep dive into turning complex architectural concepts into a working reality. My focus was on ensuring our solution wasn't just technically sound but also understandable and presentable. Designing the UI and crafting the narrative around our NATS-based architecture taught me how crucial clear communication is in bridging the gap between advanced engineering and its real-world impact. Seeing our modular vision come to life on the dashboard was incredibly rewarding."
- **Aman (Backend & Systems Architect):**
    - "Building the core backbone of the system—connecting the Anchor to NATS and ensuring the modules could communicate seamlessly—was the ultimate challenge. It was like being a conductor ensuring every instrument in an orchestra plays in perfect sync, even if one suddenly stops. Debugging the initial connection issues and finally achieving true hot-plugging and fault isolation was a moment of pure triumph. This experience solidified my understanding of distributed systems in a way no textbook ever could."
- **Dr. Tejaswi Khanna (Mentor):**
    - "Guiding Syn/Ack has been a remarkable experience. Pragyensh's focus on the user-facing aspects and Aman's deep dive into the backend created a perfect balance of form and function. Their ability to divide the problem domain, tackle complex issues in NATS integration and module decoupling, and then synthesize it all into a coherent system was impressive. Their journey is a testament to how hands-on building—coupled with strong research—can create robust and innovative solutions."

- ## Future Work :
- **AIORI Framework Integration & Data Democratization**
    - Showcase real-time Docker container metrics and system health on a public-facing landing page, transforming raw data into actionable insights for the broader community.
    - Develop standardized APIs to seamlessly integrate our modular measurement node into the AIORI-IMN framework, enabling distributed, large-scale Internet measurement.
    - Create advanced, interactive dashboard visualizations that not only display data but also tell the story of network performance and system behavior.
- **Standards Development & Ecosystem Collaboration**
    - Publish our NATS-based architectural patterns and interoperability results as an Internet-Draft, contributing a modern, cloud-native approach to measurement frameworks.
    - Combine and correlate our measurement data with datasets from IETF and other research bodies to identify larger Internet trends and patterns.
    - Develop Best Current Practices documents for building resilient, decoupled measurement infrastructure, based on our proven implementation.
- **Protocol Enhancements & Architectural Revolution**
    - Extend our module ecosystem to include QUIC performance measurement, RPKI validation, and BGP monitoring, making the platform a one-stop-shop for Internet health checks.
    - Pioneer the use of our NATS-based, hot-pluggable architecture as a new paradigm for building adaptive, future-proof Internet measurement tools.
    - Explore AI/ML-driven auto-scaling and predictive module deployment to create a truly self-healing, intelligent measurement network.

- **AIORI-2: Reporting and Standards Mapping**
  - **Standards Reference**

RFC / Draft No.	Title / Area	Lifecycle Stage	How This Work Relates
RFC 2330	IP Performance Metrics Framework	Proposed Standard	Implements and extends the architectural framework by providing a practical NATS-based modular implementation for scalable and fault-tolerant Internet measurement systems.
RFC 4656	One-way Active Measurement Protocol (OWAMP)	Proposed Standard	Provides the foundational principles for active measurements that our latency-ping module implements within the decoupled architecture.
NATS Protocol	Cloud Native Messaging System	Open Standard	Serves as the core messaging backbone enabling hot-plugging, fault isolation, and load balancing across measurement modules.

  - **Impact on Standards Development**

Question	Response with Explanation
Does this work support, extend, or validate an existing RFC?	**Yes - Extends RFC 2330:** Our implementation provides a practical, cloud-native architectural pattern for building modular measurement systems, demonstrating how RFC 2330's framework principles can be implemented with modern message-based decoupling.
Could it influence a new Internet-Draft or update sections of an RFC?	**Yes - Could influence measurement architecture BCP:** Our NATS-based approach for hot-pluggable modules and fault isolation could be documented as a Best Current Practice for building resilient Internet measurement infrastructure.
Any feedback or data shared with IETF WG mailing lists?	**Planned for IPPM WG:** Will share architectural insights and implementation experience with building decoupled measurement systems using cloud-native patterns, including performance data on hot-plugging and fault isolation.
Planned next step	1. Open-source the modular measurement framework 2. Submit architectural patterns to IPPM WG 3. Draft BCP text for NATS-based measurement systems 4. Extend module ecosystem with QUIC and RPKI measurements

- ## About the Authors :

  - **Syn/Ack** is composed of two dedicated developers, each bringing complementary expertise and a shared commitment to innovative networking solutions and modular system architecture. We share a common vision—to create systems that are not only efficient and scalable but also resilient and adaptive to dynamic internet conditions.

  - **Pragyensh Pritiman Panda (Research & Presentation Lead)** is an undergraduate student at Amity University with a strong academic record (9.2 CGPA) and diverse technical skills in Full Stack Development, AI/ML tools, and Public Speaking. With experience building modern web applications using Next.js, React, and Flutter, and a proven track record in competitive hackathons including being selected for the AIORI-2 Physical Hackathon Finale, he brings strong research, UI/UX design, and communication skills to the team. His leadership roles as Head Boy and event host have honed his ability to coordinate teams and present complex technical concepts effectively.

  - **Aman (Backend & Systems Architect)** is a full-stack developer with deep expertise in cloud computing and a strong enthusiasm for data analytics. His backend development skills and understanding of distributed systems were crucial in architecting the NATS-based modular framework, ensuring robust connections between the Anchor service and measurement modules. His cloud computing knowledge helped design the containerized deployment strategy, while his data analytics interest drove the implementation of meaningful metrics collection and visualization.

  - Crucial to our progress was the guidance of our mentor, **Dr. Tejaswi Khanna**, whose deep insight into internet infrastructure ensured our project strategy remained rigorous and industry-relevant. His mentorship created an environment of exploration and constructive feedback, helping us navigate complex architectural decisions and transform challenges into valuable learning opportunities.

  - Together, we formed a nimble, collaborative unit that combined frontend excellence with backend robustness, eager to tackle complex distributed systems problems and committed to building solutions with real-world impact.

We thrive on the perfect handshake between research and execution—hence the name Syn/Ack.

- ## Acknowledgement :

We extend our deepest gratitude to the AIORI organizing committee, our mentors, the NATS.io community for their elegant messaging system, and the IETF community (especially authors of RFC 2330) for providing the foundational standards. Our sincere thanks also go to the open-source ecosystem and our fellow participants for making this sprint transformative.

- ## References:

  - RFC 2330 - IP Performance Metrics (IPPM) Framework
  - RFC 4656 - A One-way Active Measurement Protocol (OWAMP)
  - RFC 5357 - A Two-Way Active Measurement Protocol (TWAMP)
  - RFC 8762 - Simple Two-way Active Measurement Protocol (STAMP)
  - RFC 2681 - A Round-trip Delay Metric for IPPM
  - RFC 7679 - A One-Way Delay Metric for IPPM
  - RFC 7680 - A One-Way Loss Metric for IPPM
  - RFC 5905 - Network Time Protocol Version 4 (NTPv4)

- RFC 8259 - The JavaScript Object Notation (JSON) Data Interchange Format
- RFC 8949 - Concise Binary Object Representation (CBOR)
- RFC 8446 - The Transport Layer Security (TLS) Protocol Version 1.3
- RFC 7519 - JSON Web Token (JWT)
- NATS.io Documentation: https://docs.nats.io/
- AIORI Testbed Documentation: [aiori.in/testbed]
- IETF IPPM Working Group: https://datatracker.ietf.org/wg/ippm/

Contact :
member 1-> Email: princepragyensh@gmail.com ; **member 2->** Email: ambxr2005@gmail.com

Mentor: Dr Tejaswi Khanna