Aura-Translation: Intent Translation for Autonomous Security Management in O-RAN

Aya Al Haj
Dept. of Electrical and Computer Engineering
Concordia University
Montreal, Canada
aya.alhaj@concordia.ca

Hyame Assem Alameddine
Ericsson Research
Ericsson
Montreal, Canada
hyame.a.alameddine@ericsson.com

Chadi Assi
Dept. of Information and System Security
Concordia University
Montreal, Canada
chadi.assi@concordia.ca

***Abstract*—To cope with the increasing complexity of network management in multi-tenant networks, Intent-Based Networking (IBN) has emerged as a new paradigm to enable flexible, agile, and automated network management. Using natural language, tenants can articulate their desired network objectives, termed *"intent"*, to the network without specifying how to achieve them. Such abstraction is coupled with the challenge of translating high-level intents into actionable policies that can drive network configuration. To address this challenge, we propose *Aura-Translation*, a novel framework designed for intent-driven security management in the Open Radio Access Network (O-RAN). Aura-Translation leverages advances in Large Language Models (LLMs) to translate O-RAN-specific security intents into policies defined in a standardized expression model. The latter is populated using a knowledge graph built following O-RAN interface security specifications and is queried based on LLM predictions of user intents. The proposed framework integrates Natural Language Inference (NLI) with a prompt refinement mechanism to validate, enhance, and ensure the reliability of the LLM's output. Using a curated dataset of O-RAN security intents of various complexities, our experimental results demonstrate that Aura-Translation, when integrated with DeepSeek-R1 (7B) achieves an average F1-score of 96.2%, thereby outperforming Llama-3.1 (8B) (95.2%) and Gemma-3 (4B) (33%) at the expense of 4.732 seconds increase in the average processing time when compared to Llama-3.1 (8B) across different intent complexities.**

***Index Terms*—O-RAN, intent-based networking, intent translation, security automation, large language models.**

I. Introduction

The adoption of the Open Radio Access Network (O-RAN) architecture introduces a fundamental shift in the telecommunication industry by promoting RAN disaggregation, openness, and intelligent management [1]. O-RAN supports the disaggregation of traditional RAN functions into modular components interconnected through standardized and open interfaces. This openness allows multi-vendor interoperability and rapid innovation in software-defined and virtualized environments [2]. It introduces a Service Management and Orchestration (SMO) platform that leverages Artificial Intelligence (AI) and Machine Learning (ML) for intelligent resources management and orchestration [3]. This architectural shift paves the way for simplified and automated network management, including security management. In fact, automated security management is key to reducing risks of misconfigurations, faults, and vulnerabilities in such a complex, multi-tenant, and multi-vendor-enabled architecture.

Intent-Based Networking (IBN) is a new emerging paradigm that can be leveraged for automated network and security management. IBN enables users, such as network operators, to express high-level goals — such as their desired network configuration, optimization, healing, and monitoring objectives — in natural language or abstract form, which can then be translated into actionable network policies and configurations [4]. Such high-level goals are termed *"intents"* and define network operational objectives without specifying how they should be implemented. Therefore, IBN enables autonomous, adaptive, and intelligent network management and avoids erroneous manual configurations [4]. In the O-RAN architecture, AI-driven intent translation can be realized within the SMO framework as an RAN Application (rApp), a software application hosted on the Non-Real-Time RAN Intelligent Controller (Non-RT RIC) that enables intelligent, policy-driven control and optimization of the RAN on a non-real-time timescale [5].

Adopting IBN requires solving the intent translation problem, which aims to translate high-level user intents into network policies that can subsequently render low-level network configurations that are pushed into the network [6]. To address this problem, recent works either followed a rule-based approach [7], which fails to generalize to complex intents, or depended solely on Large Language Models (LLMs) while overlooking their risks of hallucination and inconsistency in the absence of validation [8], [9]. Further, except for [7], which explored IBN for performance optimization and applications orchestration in O-RAN, to the best of our knowledge, none of the existing works addressed the problem of intent-driven security management in O-RAN.

To fill this gap, we propose Aura-Translation, an intent-translation framework designed for translating O-RAN security intents into a standard Third Generation Partnership Project (3GPP) compliant intent expression model [10]. Aura-Translation is a use-case and LLM-agnostic framework that can be easily adapted for other O-RAN management use cases

National Cybersecurity Consortium, the Government of Canada, Ericsson Canada, and Concordia University.

ISBN 978-3-903176-82-9

while leveraging any LLM of choice. Our contributions can be summarized as follows:

- We propose *Aura-Translation*, the first intent translation framework for O-RAN security management, compliant with both O-RAN and 3GPP standardization, and deployable within O-RAN SMO architecture.
- We develop a novel LLM-agnostic intent-based translation framework which integrates several LLMs such as DeepSeek-R1 (7B), Llama-3.1 (8B), and Gemma-3 (4B); with a Natural Language Inference (NLI) model and a prompt refinement mechanism to enhance the translation accuracy. The use of NLI ensures the semantic correctness of LLMs' output when representing users' intents in the form of a tuple $\alpha = \langle \text{interface}, \text{security\_control}, \text{conditions} \rangle$, thus, enhancing *Aura-Translation* reliability and performance in the face of possible hallucinations.
- We follow the O-RAN standardization to build an O-RAN knowledge graph that captures O-RAN interface security requirements. We query this graph using the output α of *Aura-Translation*'s LLMs to build a 3GPP-compliant intent model [11] that represents the user intent. The use of the knowledge graph jointly with LLMs enables overcoming the LLM domain-specific training and enables *Aura-Translation* to generalize to other security use cases through the extension of the knowledge graph.
- We build a dataset encompassing intents of various complexities on O-RAN interface security, and make it publicly available.
- Our experimental results demonstrate that *Aura-Translation*, when evaluated across intents of various complexities, achieves an average F1-score of 96.2% when integrated with DeepSeek-R1 (7B), thus outperforming Llama-3.1 (8B) (95.2%) and Gemma-3 (4B) (33%) at the expense of 4.732 seconds increase in the average processing time when compared to Llama-3.1 (8B).

The remainder of this paper is structured as follows. Section II reviews the literature of IBN. Section III provides a background on IBN. Section IV discusses *Aura-Translation*, our proposed framework for security intent translation in O-RAN. Section V illustrates the process of O-RAN security intents dataset generation. Section VI describes the experimental setup and analyzes the results. Finally, Section VII concludes the paper and details future research directions.

II. Literature Review

Many works in the literature addressed the intent translation problem. In [8], the authors customized an LLM for intent extraction and translation in the Fifth Generation (5G) core networks. The authors of [12] developed a Natural Language Processing (NLP) based framework for IBN for a healthcare use case. Their approach converts user intents into structured forms that contain key attributes such as user, goal, action, target, and time frame. Their work focuses mainly on intent extraction and structuring. Lumi is a system designed in [13] that allows operators to express intents in natural language, translates them into network policies via a proposed abstraction layer called Network Intent Language (Nile), and refines them through feedback loops to ensure correctness. Lumi's translation is limited to the extraction, from the user intent, of specific entities that have a matching set of operations defined in Nile, preventing its generalization to different intent types. Authors of [14] proposed a solution for intent translation, policy execution, and deployment. They decomposed intents into a hierarchy of policies and used few-shot learning to retrain an LLM for intent translation. Nonetheless, their work was tested only on a single intent. [15] proposed exploring an LLM-based chatbot for intent acquisition and translation as a doctoral study without any detailed methodology or experimental results.

IBN in O-RAN is still in its infancy, with only limited works in the literature tackling it. For instance, the work in [9] presented AGIR, a system for Service-Level Agreements (SLAs) fulfillment in O-RAN that performs intent translation and conflicting intents resolution and enforcement. The authors leveraged NLP for intent translation and used a rule-based model that maps an intent to an object-action-result format. Such a format is rigid for singular intent but fails to generalize to complex intents. The work in [16] presented an LLM-centric intent life cycle management architecture focused on intent decomposition into RAN and cloud/edge intents and their translation using Code Llama LLM. The latter was adapted using few-shot learning, and its output was validated using human feedback, which limits system automation. Other work, such as [7], leveraged Hierarchical Reinforcement Learning (HRL) to dynamically select and orchestrate eXtended Applications (xApps) in O-RAN to meet the operator's intent for Key Performance Indicators (KPIs). Although this work is designed for O-RAN, it does not tackle intent translation.

While none of the works in the literature on intent translation tackled the security aspect in O-RAN, most of them either used NLP or LLM for intent translation. NLP requires a specific grammar format to correctly capture user intent. LLM-based solutions, however, relied on few-shot learning to acquire domain-specific knowledge and lacked validation techniques against LLM hallucination. In this work, we address this gap by proposing a use case and an LLM-agnostic intent translation framework that augments LLMs with an extensible knowledge base for O-RAN intent-based management and NLI for intent validation.

III. Intent-Based Networking Life Cycle

The concept of intent in network and service management has been addressed by several standardization bodies. 3GPP defines intent-driven management services for mobile networks and details an intent information model. [10]. The TeleManagement Forum (TM Forum) introduces intent as part of its Open Digital Architecture (ODA) and intent ontology, while focusing on business and service layer expressions of customer requirements [17]. The European Technical Standard Institute (ETSI) Zero-touch network and Service Management

(ZSM) group defines intent as mean of automation and specifies its integration within the ZSM framework while touching base on its use in the telecommunication industry. The Internet Engineering Task Force (IETF) in RFC 9315 [18] defines the intent and outlines its lifecycle, including its translation, fulfillment, and assurance.

IBN enables the autonomous operation of systems by decomposing declarative intents into enforceable configurations. The intent is a high-level goal specified by the operator and understandable by both humans and machines. This process is orchestrated through a continuous life cycle (Figure 1) encompassing different functions detailed hereafter [4], [18].

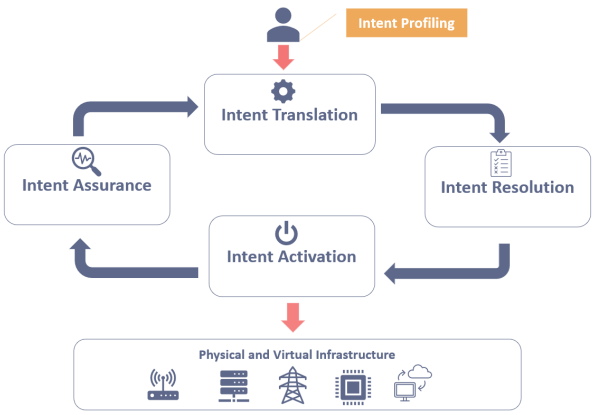

Fig. 1: IBN life cycle [4] [6].

- **Intent Profiling:** The first step in IBN consists of acquiring the user's high-level goal in natural language. It enables the user to express a meaningful desired objective to be implemented in the network [6], [18].
- **Intent Translation:** Intent translation bridges the gap between human and machine-readable intent formats. It translates user intent expressed in natural language into a structured intent, also known as network policy, following one of the known standards (i.e., 3GPP/TM). Policies can then be rendered into network configurations [6].
- **Intent Resolution:** As users may submit intents independently, conflicting configurations may arise. As such, the network configuration, obtained after the intent translation, is analyzed to detect and resolve any conflicts with implemented ones. In case conflict resolution is not feasible, the user or network administrator is alerted [6].
- **Intent Activation:** Intent activation involves effectuating the user's intent through enforcing the rendered network configuration after resolving any conflicts [6], [18].
- **Intent Assurance:** Intent assurance ensures that the network complies with the intent throughout its lifetime. The system collects telemetry and KPIs to monitor the state of the network. In case any deviation from the intended state is detected, the system triggers corrective actions, acting as a self-healing closed loop [6].

The scope of this work is limited to intent profiling and translation, while the remaining intent life-cycle stages, including intent resolution, enforcement, and assurance, are considered as a future direction.

IV. Aura-Translation: An Intent Translation Framework

In this section, we introduce Aura-Translation, a framework for intent-based security management in O-RAN (Figure 2). Aura-Translation combines LLMs with NLI for translating user intent into a structured tuple that can be used to query a knowledge graph constructed following O-RAN security standard specifications. The query results are then used to algorithmically generate 3GGP-compliant policies representing user intents. These policies can later be employed to render network configurations. Aura-Translation encompasses several components detailed in the following, with an illustrative example expressing the framework functionality.

A. User Input Processor

The user input processor is the first component of Aura-Translation. It acquires user intent and leverages an *LLM-based intent extractor* to extract a structured tuple from the intent and an *NLI-based verifier* to validate the LLM's output and ensure its correctness.

1) LLM-Based Intent Extractor: In contrast to conventional NLP or rule-based approaches, which typically require well-formed inputs and repeated user refinements, LLMs guided by domain-specific prompts can effectively handle linguistic variation and incomplete descriptions. For example, when a user describes the interaction between the Non-RT RIC and the Near-RT RIC without explicitly naming the interface, the LLM can correctly infer the A1 interface. The model can also identify inconsistent or infeasible requests based on its understanding of O-RAN concepts. Building on these capabilities, we leverage an LLM to generate a structured intent of a fixed structure $\alpha = \langle \text{interface}, \text{security\_control}, \text{conditions} \rangle$ from the user intent. The *interface* in α represents the targeted O-RAN interface, the *security_control* is the security goal to be enforced (e.g., confidentiality, integrity, authentication), and the *conditions* specify contextual constraints to consider during intent activation. This tuple representation is inspired by the work of [7], and adapted to suit the requirements of O-RAN security management. It encapsulates the essential parameters needed to represent security intents in a compact form that can be easily processed in the following IBN life cycle stages (Section III). The fixed structure of α reduces ambiguity and enables verification of the LLM output. Furthermore, limiting the LLM's role to lightweight information extraction mitigates hallucinations and ensures consistent performance across diverse user inputs [19].

To further guarantee accurate tuple extraction despite errors in typing or imprecise definitions of O-RAN components by users, we perform LLM prompt engineering. Through an iterative design process, we refined multiple prompt templates and tested them on sample user intents to optimize the quality and robustness of the extracted tuples. These prompts incorporate domain knowledge of the O-RAN architecture and interfaces,

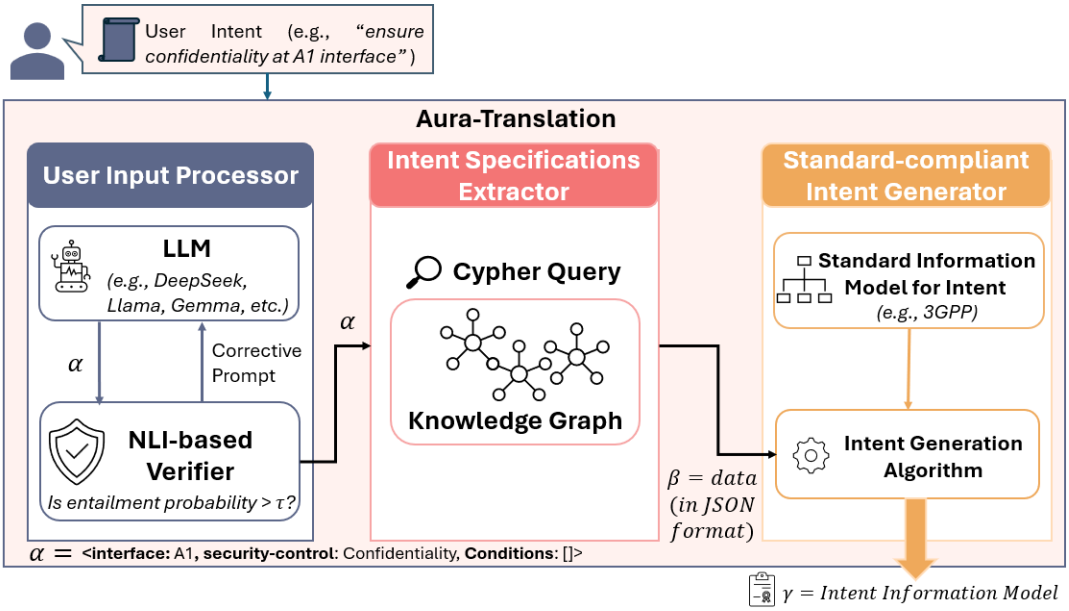

Fig. 2: Aura-Translation architecture.

guiding the LLM to correctly infer requirements from diverse user inputs and produce outputs in a structured JSON [20] format α. These prompts also include instructions for handling and interpreting complex user intents (i.e., intent with multiple objectives or sub-intents).

2) Natural Language Inference (NLI)-based Verifier: LLMs may generate outputs that are inaccurate, hallucinated, or incorrectly structured. Therefore, it is essential to verify that α, extracted by the LLM, accurately captures the intended meaning of the original user intents before they are processed downstream. Existing verification approaches include human-in-the-loop evaluation, which is reliable but not scalable for automated systems [21]. Another approach is the LLM-as-a-judge paradigm, where one model evaluates the output of another; however, this method may reinforce shared biases and lacks explicit logical reasoning of the results [22]. Traditional rule-based or keyword-matching techniques are lightweight but struggle with linguistic variability and complex intents, limiting their robustness in dynamic language scenarios [23].

To address these limitations, Aura-Translation employs a Natural Language Inference (NLI) model, specifically *RoBERTa-large-MNLI* [24], to semantically validate the extracted tuple α. NLI determines whether a "hypothesis" (i.e., a reconstructed intent derived from α) is entailed by, contradicts, or is neutral with respect to a "premise" (i.e., the original user intent) [25]. This enables semantic validation beyond surface-level similarity and ensures that α represents the user intent.

The NLI model outputs a probability distribution over three classes: `ENTAILMENT`, `NEUTRAL`, and `CONTRADICTION`, modeling the relationship between the hypothesis and the premise. The model outputs an `entailment probability`, also known as the `hallucination/faithfulness score`, a scalar value between 0 and 1 that quantifies the relation between hypothesis and premise [25]. Aura-Translation reduces the possibility of classifying the entailment as a contradiction by considering multiple hypothesis formulations and selecting the one with the highest entailment probability.

A decision threshold τ is applied to classify the extracted tuple. Hypotheses with an entailment probability $\geq \tau$ are accepted and forwarded to the knowledge graph extractor, while those below τ are labeled as `CONTRADICTION`. When a contradiction is detected, a refined prompt—containing the original user intent, the extracted tuple α, and its entailment probability—is issued to the LLM to generate a corrected tuple. If the refined output satisfies the threshold, it proceeds downstream; otherwise, intent translation is deemed unsuccessful.

Overall, the NLI-based verifier acts as a lightweight semantic validation layer that filters incorrect or inconsistent LLM outputs before policy generation [25]. Since all downstream stages rely on static knowledge graph queries and standardized intent models, the LLM-based tuple extraction remains the only non-deterministic component. The integration of NLI therefore improves robustness, limits error propagation, and enhances trustworthiness without introducing significant computational overhead.

B. Intent Specifications Extractor

After capturing the user requirements through intent, there is a need to determine how such intent should be enforced and activated in the network. This requires a knowledge graph that can be queried based on the main elements of the user intent, represented by the tuple $\alpha = \langle \text{interface}, \text{security\_control}, \text{conditions} \rangle$; to retrieve the technical details (e.g., protocols and their configurations, etc.) that can serve in formulating how the intent can be activated.

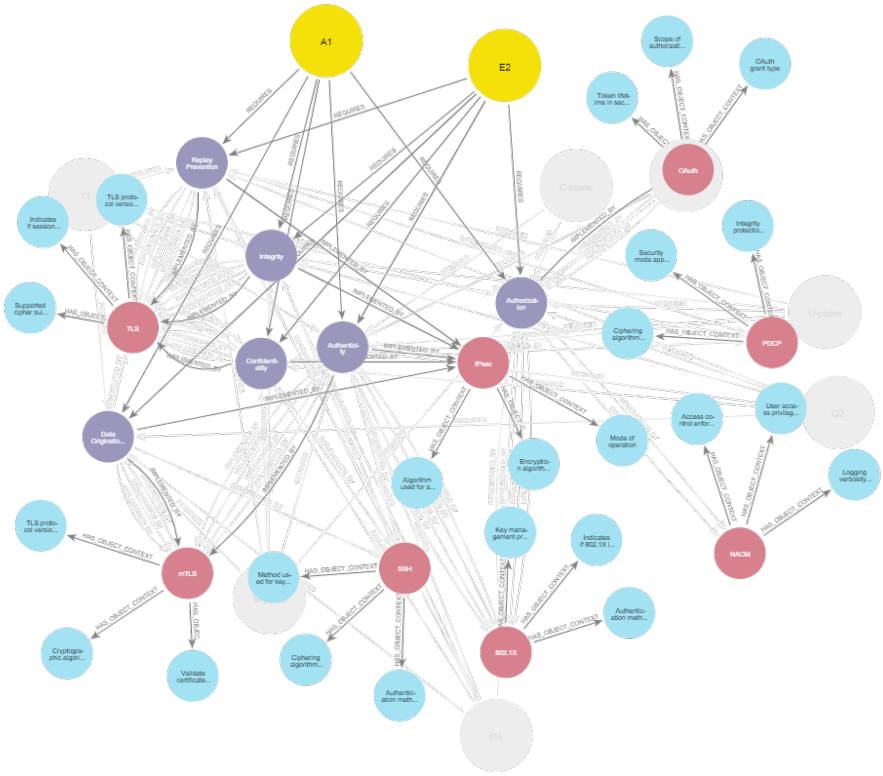

Fig. 3: Partial view of the O-RAN knowledge graph. Some parts are blurred to improve visual clarity.

As such, the intent specifications extractor accounts for a knowledge graph built based on O-RAN Alliance (i.e., the leading body of O-RAN) specifications[1] [11]. A partial view of this graph is illustrated in Figure 3, while the set of rich relationships it models is detailed in Figure 4. In this work, we limit the scope of the knowledge graph to O-RAN interface security controls specified in [11]. However, the proposed model is extensible and can incorporate additional security goals and other O-RAN network requirements by extending the graph with appropriate nodes and relationships that capture the relevant data.

- `Interface` → **`REQUIRES`** → `SecurityControl`
- `SecurityControl` → **`IMPLEMENTED_BY`** → `Protocol`
- `Protocol` → **`HAS_OBJECT_CONTEXT`** → `ObjectContext`
- `Protocol` → **`HAS_EXPECTATION_TARGET`** → `ExpectationTarget`
- `SecurityControl` → **`RECOMMENDED_FOR`** → `UseCase`

Fig. 4: Relationships represented in the knowledge graph.

We use a cypher query to extract details from the knowledge graph with placeholders for the interface and security controls obtained from α, extracted from the user intent, and generated by the user input processor (Section IV-A). The query result includes the protocol assigned by the O-RAN Alliance to achieve the security control, along with its implementation details. The protocol details account for configuration constraints (e.g., key length, encryption mode, or authentication type) and operational metrics (e.g., latency thresholds or session integrity requirements) required for its enforcement in the network.

[1]The code to reproduce the graph is available in our GitHub repository [26].

C. Standard-compliant Intent Generator

To generate policies that can be translated into network configurations as part of the intent translation, we resort to standardized information models for intents such as those defined by 3GPP [10] and TM forum [17]. We use the data retrieved from the knowledge graph (Section IV-B) to populate such a standardized information model in YAML format [27] using an algorithmic approach that maintains the predefined information model with all valid values and constraints as specified in the standard. This ensures the generation of deterministic, valid, and standard-compliant intent, which we also refer to as policies, representing the user intent. In this work, we opt for the 3GPP information model for intent [10] as it is adopted by the O-RAN ALLIANCE for some IBN use cases [28]. However, our proposed solution can be adapted to any other standardized intent model (e.g., TM forum intent model [17]) that defines a comparable abstraction for expressing intents.

D. Illustrative Example of Aura-Translation Functionality

To demonstrate the functionality of Aura-Translation, we present an example showing the full intent translation pipeline.

a) User input processing through LLM: The user begins by expressing a high-level intent in natural language:

"Ensure confidentiality at the E2 interface."

The *user input processor* extracts a tuple (i.e., in case of single intent) or a list of tuples (i.e., in case of complex intents) α from the user intent using the LLM, and represents them in a JSON format shown in Figure 5.

```
[{
    "interface"  "E2",
    "security_control"  "Confidentiality",
    "conditions"  []
}]
```

Fig. 5: Intent in JSON format.

b) Intent verification via NLI: α is then semantically validated using the NLI-based verifier. For that, the NLI-based verifier, configured with a set of hypothesis templates, constructs several hypothesis statements by populating those templates with the information obtained from α, such as:

- *"The intent requires confidentiality at the E2 interface."*
- *"Ensure confidentiality is applied at E2."*
- *"E2 must have confidentiality."*
- *"Apply confidentiality at E2."*

The NLI model evaluates each hypothesis against the premise, original user intent, and returns an entailment probability. For this example, the hypothesis that achieves the highest entailment probability **ENTAILMENT: PASS (0.98)**, is *"Ensure confidentiality is applied at E2."*. This confirms that the extracted tuple α is valid and accurately represents the user's intent.

c) Knowledge graph querying: After verifying the validity of structured intent α, the latter is fed to the *intent specifications extractor* cypher query template to retrieve the data from the knowledge graph. The obtained result in Figure 6 determines how to enforce confidentiality at the E2 interface.

- **Protocol:** IPsec
- **Object Contexts:**
  - EncryptionAlgorithm: AES-GCM-256, AES-GCM-128, ChaCha20-Poly1305
  - AuthenticationAlgorithm: HMAC-SHA-256, HMAC-SHA-512
  - TunnelMode: transport, tunnel
- **Expectation Targets:**
  - EncryptionLatency: IS_LESS_THAN 5 ms
  - SecureThroughput: IS_GREATER_THAN 10 Gbps

Fig. 6: E2 confidentiality enforcement configuration.

This result specifies the parameters corresponding to the IPsec configuration recommended by the O-RAN ALLIANCE for enforcing confidentiality at the E2 interface.

d) 3GPP-compliant intent generation: Using the information retrieved from the knowledge graph, the *standard-compliant intent generator* populates the standardized 3GPP information model for intent. It produces a YAML-based representation that conforms to the schema defined in 3GPP, which represents the policy that can be later used to generate the network configuration for intent resolution and activation. An excerpt of the generated intent is shown in Figure 7.

```
Intent:
  userLabel: InterfaceSecurityControl
  intentExpectation:
  - expectationId:
      69976ca4-4b2e-4171-a2ca-ca643ca20a46
    expectationVerb: Ensure
    expectationObjects:
    - objectInstance: E2_Interface
      objectContexts:
      - contextAttribute: EncryptionAlgorithm
        contextCondition: IS_ALL_OF
        contextValueRange:
        - AES-GCM-256
        - AES-GCM-128
        - ChaCha20-Poly1305
      - contextAttribute: AuthenticationAlgorithm
        contextCondition: IS_ALL_OF
        contextValueRange:
        - HMAC-SHA-256
        - HMAC-SHA-512
    expectationTargets:
    - targetName: EncryptionLatency
      targetCondition: IS_LESS_THAN
      targetValueRange: '5'
    - targetName: SecureThroughput
      targetCondition: IS_GREATER_THAN
      targetValueRange: '10'
  intentPriority: 10
  observationPeriod: 60
  intentAdminState: ACTIVATED
```

Fig. 7: 3GPP-compliant information model for intent.

It is worth emphasizing that Aura-Translation is use-case agnostic as it can be extended to any use-case by updating its knowledge graph.

E. Translation Errors

In Aura-Translation, errors may arise during the LLM-based tuple extraction stage (Section IV-A), as subsequent policy generation relies on static knowledge graph queries and deterministic, standard-compliant algorithms. Invalid tuples that do not conform to the expected JSON structure (Figure 5) are rejected prior to downstream processing. Another type of error occurs when a syntactically valid tuple contains elements that differ from the user's request; this case is typically detected by the NLI-based verifier. In the unlikely event of incorrect NLI thresholding, such an error may lead to the enforcement of a valid but unintended security policy. Beyond these cases, no additional error sources exist, since data extraction and policy generation are fully controlled.

V. O-RAN Security Intent Dataset

Intent-based management for O-RAN security remains largely underexplored, thus lacking publicly available benchmarks. Therefore, to validate our design choices and the correctness of our proposed approach, we generated a dataset

focused on O-RAN security management and, more specifically, O-RAN interfaces' security.

TABLE I: O-RAN interface security controls [11]

Security Control	Non-Fronthaul					Open-Fronthaul			
	A1	O1	O2	E2	Y1	C-plane	U-plane	S-plane	M-plane
Authenticity	mTLS	mTLS	mTLS	IPsec	mTLS	802.1X	802.1X	802.1X	mTLS/SSH/802.1X
Confidentiality	TLS	TLS	TLS	IPsec	TLS	TLS	PDCP		TLS/SSH
Integrity	TLS	TLS	TLS	IPsec	TLS	TLS	PDCP		TLS/SSH
Authorization	OAuth	NACM	OAuth	OAuth		802.1X	802.1X	802.1X	NACM/802.1X
Data Origination	mTLS	mTLS	mTLS	IPsec	mTLS				TLS/SSH
Replay Prevention	TLS	TLS	TLS	IPsec	TLS	TLS	PDCP		TLS/SSH

Using the information in Table I specified by O-RAN ALLIANCE [11], we generate a dataset[2] encompassing 1800 possible intents that programmatically combine the mentioned interfaces and security controls into natural language expressions. Each expression is designed to model a user's input intent. As user intents can be complex and may include multiple O-RAN interfaces and security controls, we do not limit our intent generation process to single intents (e.g., *"Ensure confidentiality at the A1 interface"*) but also account for the complex intents (e.g., *"Ensure confidentiality at A1 and integrity at E2"*). To generate such a dataset, we consider variations in the linguistic phrasing to simulate realistic user input while considering the following combinations:

1) *SingleIntent:* User inputs accounting for a single security control to be applied at a single interface.
2) *xInterfaces:* User input accounting for a single security control to be applied at multiple interfaces.
3) *xControls:* User inputs accounting for multiple security controls to be applied to a single interface.
4) *xControlsxInterfaces:* User inputs accounting for different security controls to be applied to different interfaces.
5) *I3+:* User inputs accounting for a complex or nested combination of intents. I3+ can include between 3 and 6 objectives (i.e., sub-intents).

We detail in Table II the statistical distribution of the generated intents across the dataset based on the different aforementioned combinations and the number of sub-intents in each generated one. For example, we denote by $I2-xInterfaces$ the subset of user inputs that include 2 sub-intents corresponding to the *xInterfaces* combination. In other words, it represents the user input requesting a single security control for 2 interfaces (e.g., *"Ensure confidentiality at A1 and E2 interfaces"*). Finally, to facilitate the evaluation of the quality of Aura-Translate, we define the expected corresponding structured tuple $\alpha = \langle \text{interface}, \text{security\_control}, \text{conditions} \rangle$ as a ground truth.

TABLE II: Distribution of the generated O-RAN intent dataset.

Tag	Intent Type	Number of Samples
I1-SingleIntent	Single control–single interface	300
I2-xInterfaces	Single control–multiple interfaces	54
I2-xControls	Multiple controls–single interface	27
I2-xControlsxInterfaces	Multiple controls–multiple interfaces	228
I3+	Complex or nested combinations ($3 \leq x \leq 6$)	1191
Total		1800

[2]The dataset is publicly available in our GitHub repository [26].

VI. Experimental Evaluation

A. Experimental Setup

To implement and evaluate Aura-Translation, we use a desktop computer equipped with an Intel Core i9-12900 CPU (16 cores, 24 threads, 2.4 GHz), 64 GB of RAM, and an NVIDIA RTX A4000 GPU (16 GB VRAM), running Windows 10 Enterprise (64-bit). The LLMs are deployed locally via LM Studio and operate with a context window of 32k tokens and 4-bit quantization. The knowledge graph was implemented using Neo4j [29], while the entire system was executed using Visual Studio Code and Python 3.10.0.

B. Experimental Results

We evaluate the efficiency of Aura-Translation using three different LLMs, namely, Gemma-3 (4B), Llama-3.1 (8B), and DeepSeek-R1 (7B) [30]. As all downstream stages— including knowledge graph querying and YAML policy generation—are deterministic and standard-compliant, correctness is enforced by design. Consequently, our evaluation focuses on the LLM output and the importance of its validation through the NLI verifier.

1) LLM Prompt Engineering: As described in Section IV-A1, we perform prompt engineering to identify the prompt that yields the most reliable structured intent extraction across the evaluated LLMs. Each model is evaluated in the O-RAN security intent data set (Section V) using 15 engineered prompts, and the resulting structured intents are automatically compared against ground-truth tuples using a devised script. The prompt that maximizes agreement with the ground truth across all evaluated intents and models is selected and subsequently used to configure Aura-Translation for all experiments[3].

2) NLI Model Threshold Selection and Evaluation: The NLI-based verifier outputs an entailment probability that is compared against a decision threshold τ to determine whether a structured intent α is accepted or requires refinement (Section IV-A2). To evaluate the NLI module, we measure its False-Positive (FP) and False-Negative (FN) rates by comparing the LLM-extracted tuples α against the ground-truth tuples in our dataset. An FP occurs when the NLI accepts an incorrect tuple, while an FN occurs when it rejects a correct one. These metrics are used to evaluate and fine-tune the performance of the NLI verifier. Threshold selection is performed using a subset of 200 structured intents of varying complexity from our dataset (Section V). By evaluating FP and FN rates across different values of τ, we select $\tau = 0.85$, which maximizes the NLI F1-score to 97%.

3) Aura-Translation Performance Without NLI-based Verifier: The performance of Aura-Translation is limited by that of its *user input processor*, which comprises the LLM and the NLI-based verifier (Figure 2). This is because the results of the *intent specifications extractor* and the *standard-compliant intent generator* are deterministic, as explained in Section IV-B and Section IV-C, respectively.

[3]The prompt is available in our GitHub repository [26].

We evaluate and compare Aura-Translation performance across three LLMs; Gemma-3 (4B), Llama-3.1 (8B), and DeepSeek-R1 (7B) without accounting for the NLI-based verifier, which can prompt the LLM again to correct its output. Our results from the dataset (Section V) are depicted in Figure 8 after leveraging the included ground truth. Figure 8 shows that Aura-Translation, when used with DeepSeek-R1 (7B), achieves the highest average F1-score of 94.6% across intents of varying complexity. DeepSeek-R1 (7B) slightly outperforms Llama-3.1 (8B) (average F1-score of 93.8%) and significantly outperforms Gemma-3 (4B) (average F1-score of 9%).

The LLM models' performance varied across intents' complexities without an apparent trend. For instance, we notice that DeepSeek-R1 (7B) and Llama-3.1 (8B) performed slightly better on complex intents with 2 sub-intents (I2), with respective average F1-scores of 95.3% and 94.6% when compared with their performance on I1 and I3+. This suggests that these LLMs have strong reasoning capabilities and effective interpretation regardless of the intent complexity. In contrast, the performance of Gemma-3 (4B) increased approximately fivefold with the increase in the intent complexity, when comparing its F1-score for I1 and I3+. This increase indicates that longer inputs provide semantics in complex intents, which can partially offset Gemma-3 (4B) limited reasoning capacity, although its performance remains significantly below that of larger models (i.e., DeepSeek-R1 (7B), Llama-3.1 (8B)).

4) Aura-Translation Performance With NLI-based Verifier: To highlight the contribution of the NLI-based verifier in enhancing Aura-Translation performance, we evaluate and compare its performance across the three LLMs when used with the NLI-based verifier in Figure 8.

The large models, namely DeepSeek-R1 (7B) and Llama-3.1 (8B), achieve consistently high F1-scores with an insignificant increase after the NLI-based verifier initiated correction through another LLM prompting. Their average F1-scores across intents with different complexities are 96.2% and 95.2%, respectively. Therefore, this indicates that the majority of intents are correctly translated in the first iteration, while approximately 5% require a refinement loop. Importantly, this refinement incurs a processing latency comparable to that of the initial inference, as it involves a single additional forward pass with similar computational complexity. In contrast, Gemma-3 (4B) depicted a significant improvement in its performance with an increase of 24% in its average F1-score (33%) when compared with its performance without an NLI-based verifier. These results were obtained after a single refinement iteration, in which the NLI-based verifier identified an incorrectly constructed intent α and guided the LLM to regenerate corrected outputs using a new prompt that includes the entailment probability as input.

5) Aura-Translation Processing Time: Aura-Translation processing time becomes critical for real-time network management, particularly for security operations that require near-real-time mitigation. As such, we evaluate the average intent processing time of Aura-Translation across varying intent complexities. The evaluation targets the LLM stage as it represents the primary computational bottleneck, since subsequent Aura-Translation components rely on static cypher queries and lightweight algorithms running in order of milliseconds.

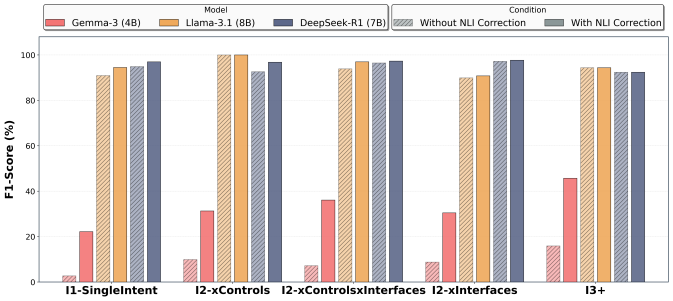

Fig. 8: Aura-Translation performance across different LLMs.

As shown in Figure 9, Aura-Translation processing time increases with the increase of intent complexity across all the evaluated LLMs. Nonetheless, the evaluated LLMs have different processing time ranges. Llama-3.1 (8B) demonstrated exceptional efficiency, maintaining the lowest processing times across all intents' complexities (averaged to 5.362 seconds). DeepSeek-R1 (7B) exhibited moderate processing time (averaged to 10.094 seconds), reflecting its more elaborate reasoning processes that resulted in better performance as previously discussed. Gemma-3 (4B) showed significantly higher processing times (averaged to 87.618 seconds) than Llama-3.1 (8B) and DeepSeek-R1 (7B) despite its smaller parameter size. This discrepancy suggests that reduced model scale does not necessarily translate to lower inference latency, especially when reasoning capacity differs across architectures.

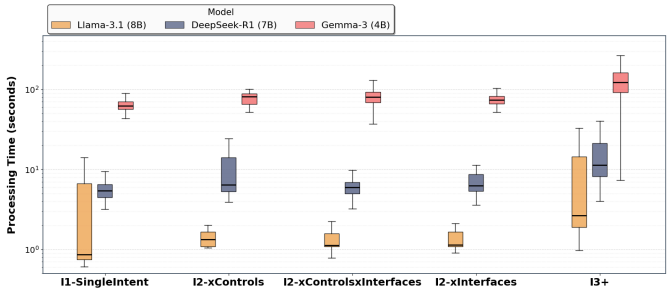

Fig. 9: Aura-Translation processing time across LLMs.

By combining the results in Figure 8 and Figure 9, we can conclude that Llama-3.1 (8B) is the most balanced LLM and recommended one for Aura-Translation as it provides the best trade off between high performance (i.e., comparable to DeepSeek-R1 (7B)) and short processing time (average processing time is 5.362 seconds across all intent complexities). Overall, Aura-Translation scales efficiently with intent complexity, maintaining acceptable end-to-end processing time suitable for deployment in the SMO as an rApp, which operates in the non-real-time domain with latency typically exceeding one second [5].

VII. Conclusion

This paper addressed the challenge of automating security management in O-RAN using IBN. We introduced Aura-Translation, a framework that translates high-level security intents in O-RAN into a 3GPP-compliant information model for intent. The experimental results on a generated O-RAN interfaces security intent dataset revealed its effectiveness in translating simple and complex intents when evaluated with three different LLMs. When used with Llama-3.1 (8B), Aura-Translation balances its performance and processing time. It achieves an average F1-score of 95.2% and an average processing time of 5.362 seconds across intents with varying complexity when integrated with an NLI model for LLM output correctness verification. Therefore, making it suitable for practical deployment in the SMO. The proposed approach provides a foundation for autonomous O-RAN security management, opening new directions toward intent-driven automation and assurance in next-generation open and intelligent networks. As future work, we aim to extend Aura-Translation capabilities to translate more sophisticated O-RAN security intents beyond those used to secure O-RAN interfaces.

Acknowledgment

This work was made possible in part through the support of the National Cybersecurity Consortium, the Government of Canada, Ericsson Canada and Concordia University. The authors would like to thank Dr. Ayse Sayin from Ericsson Research for their invaluable feedback.

References

[1] O-RAN Alliance, "O-ran architecture description," O-RAN Alliance, Tech. Rep. O-RAN.WG1.O-RAN-Architecture-Description-v07.00, 2023. [Online]. Available: https://www.o-ran.org/

[2] M. Polese, L. Bonati, S. D'Oro, S. Basagni, and T. Melodia, "A survey on open radio access networks: Challenges, research directions, and open source approaches," *Sensors*, vol. 24, no. 3, p. 892, 2024.

[3] M. A. Habibi, B. Han, M. Saimler, I. L. Pavón, and H. D. Schotten, "Towards an AI/ML-driven SMO framework in O-RAN: Scenarios, solutions, and challenges," in *2024 IEEE Future Networks World Forum (FNWF)*, Dubai, United Arab Emirates, 2024, pp. 7–14.

[4] ETSI, "Zero-touch network and service management (zsm); requirements based on documented scenarios," https://www.etsi.org/deliver/etsi_gr/ZSM/001_099/011/01.01.01_60/gr_ZSM011v010101p.pdf, ETSI, Tech. Rep. GR ZSM 011 V1.1.1, accessed: 24 Jan. 2025.

[5] ShareTechnote, "rapp in oran architecture," https://www.sharetechnote.com/html/OpenRAN/OR_rAPP.html, 2024, accessed: 2026-01-27.

[6] A. Leivadeas and M. Falkner, "A survey on intent-based networking," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 1, pp. 625–655, 2023.

[7] M. A. Habib, "Intent-driven intelligent control and orchestration in o-ran via hierarchical reinforcement learning," in *2023 IEEE 20th International Conference on Mobile Ad Hoc and Smart Systems (MASS)*, Toronto, Canada, 2023.

[8] D. M. Manias, A. Chouman, and A. Shami, "Towards intent-based network management: Large language models for intent extraction in 5g core networks," in *2024 20th International Conference on the Design of Reliable Communication Networks (DRCN)*. Montreal, QC, Canada: IEEE, 2024, pp. 1–6.

[9] R. de Oliveira, I. M., M. D., S. V. d. M., M. A. L., and D. M., "An agile conflict-solving framework for intent-based management of service level agreement," in *2023 2nd International Conference on 6G Networking (6GNet)*, Paris, France, 2024.

[10] 3GPP, "Management and Orchestration; Intent driven management services for mobile networks," 3rd Generation Partnership Project (3GPP), Tech. Rep. TS 28.312, 2021.

[11] O.-R. A. S. W. G. (WG11), "O-ran alliance security update 2025," https://www.o-ran.org/blog/o-ran-alliance-security-update-2025, 2025, accessed: 2025-10-08.

[12] Y. Njah, A. Leivadeas, J. Violos, and M. Falkner, "Toward intent-based network automation for smart environments: A healthcare 4.0 use case," *IEEE Access*, vol. 11, pp. 136 565–136 576, 2023.

[13] A. S. Jacobs, R. J. Pfitscher, R. H. Ribeiro, R. A. Ferreira, L. Z. Granville, W. Willinger, and S. G. Rao, "Hey, lumi! using natural language for intent-based network management," in *2021 USENIX Annual Technical Conference (USENIX ATC 21)*. USENIX Association, Jul. 2021, pp. 625–639. [Online]. Available: https://www.usenix.org/conference/atc21/presentation/jacobs

[14] K. Dzeparoska, A. T., and A. L.-G., "Emergence: An intent fulfillment system," *IEEE Communications Magazine*, vol. 62, 2024.

[15] M. Fontana, B. Martini, and F. Sciarrone, "Exploring large language models in intent acquisition and translation," in *2024 IEEE 10th International Conference on Network Softwarization (NetSoft)*. Saint Louis, MO, USA: IEEE, 2024, pp. 231–234.

[16] A. Mekrache, A. K., and C. V., "Intent-based management of next-generation networks: an llm-centric approach," *IEEE Network*, vol. 38, 2024.

[17] TM Forum, "TR292: TM Forum Intent Ontology (TIO)," TM Forum, Tech. Rep., 2023. [Online]. Available: https://www.tmforum.org/resources/technical-report/tr292-tm-forum-intent-ontology-tio-v3-1-0/

[18] A. Clemm, L. Ciavaglia, L. Zambenedetti Granville, and J. Tantsura, "Intent-based networking - concepts and definitions," RFC 9315, Internet Research Task Force (IRTF), October 2022. [Online]. Available: https://datatracker.ietf.org/doc/rfc9315/

[19] Z. Yang, J. Chen, N. Ding, Y. Qin, W. X. Zhao, Z. Liu, M. Sun *et al.*, "Harnessing the power of llms in practice: A survey on chatgpt and beyond," *arXiv preprint arXiv:2304.13712*, 2023. [Online]. Available: https://arxiv.org/abs/2304.13712

[20] T. Bray, "The json data interchange format," https://www.rfc-editor.org/rfc/rfc8259, 2017, iETF RFC 8259.

[21] V. Elangovan, Q. Liu, H. Xu, S. Bodapati, and D. Roth, "Considers: The human evaluation framework for generative large language models," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL 2024)*, 2024. [Online]. Available: https://aclanthology.org/2024.acl-long.63/

[22] W. Shi, X. Ma, Y. Liang, S. Vosoughi *et al.*, "Judging the judges: A systematic study of position bias in llm-as-a-judge," *arXiv preprint arXiv:2406.07791*, 2024. [Online]. Available: https://arxiv.org/abs/2406.07791

[23] J. Liang, W. Zhao, M. Chen, Y. Zhou, and J. Huang, "Figv: Fine-grained constraint generation-verification for reliable large language models," in *International Conference on Learning Representations (ICLR 2025)*, 2025. [Online]. Available: https://openreview.net/forum?id=NAdBbxbnv2

[24] F. AI, "roberta-large-mnli: Roberta large fine-tuned on mnli," https://huggingface.co/FacebookAI/roberta-large-mnli, 2024, accessed: 2025-11-05.

[25] A. Pagnoni, V. Balachandran, and Y. Tsvetkov, "Understanding faithfulness and hallucination in neural text generation: A survey," *Computational Linguistics*, 2022. [Online]. Available: https://arxiv.org/abs/2202.03629

[26] A. Al Haj, "Aura-translation: Intent translation for autonomous security management," https://github.com/AyaAlHaj17/Aura-Translation-Intent-Translation-for-Autonomous-Security-Management, 2026, gitHub repository.

[27] O. Ben-Kiki, C. Evans, and I. döt Net, "Yaml ain't markup language (yaml) version 1.2," https://yaml.org/spec/1.2/spec.html, 2009, yAML Language Specification.

[28] O.-R. ALLIANCE, "Wg1 tr smo-int-r004 v06.00 — smo intent management technical report," https://specifications.o-ran.org/specifications, 2024, version v06.00; accessed: 2025-11-05.

[29] I. Neo4j, "Neo4j — graph database analytics," https://neo4j.com/, 2025, accessed: 2025-11-05.

[30] "Lm studio model catalog," https://lmstudio.ai/models, accessed: 2025-10-10.