

# Knowledge Graph for Network Management

Mingzhe Xing

[xingmz@zgclab.edu.cn](mailto:xingmz@zgclab.edu.cn)

Zhongguancun Lab

# Network Data Challenge

- **Data Overload:** Analysis is hampered by volume, complex correlations, and loss of critical content.
- **Vendor-Specific Silos:** YANG models from different vendors, IETF, and OpenConfig describe the same concept (e.g., an interface) with varied structures and syntax, creating data silos.
- **Heterogeneous Data:** Difficulty in correlating vast amounts of diverse operational data (configurations, logs, processes, etc.).

# Net KG Drafts @NMOP

- Knowledge Graph Framework for Network Operations (draft-mackey-nmop-kg-for-netops-03)
- Knowledge Graphs for YANG-based Network Management (draft-marcas-nmop-knowledge-graph-yang-05)
- Knowledge Graph for Network Traffic Monitoring and Analysis (draft-pang-nmop-kg-for-traffic-monitoring-analysis-02)
- Knowledge Graphs for Enhanced Cross-Operator Incident Management and Network Design (draft-tailhardat-nmop-incident-management-noria-04)
- .....

# How to use Net KG?

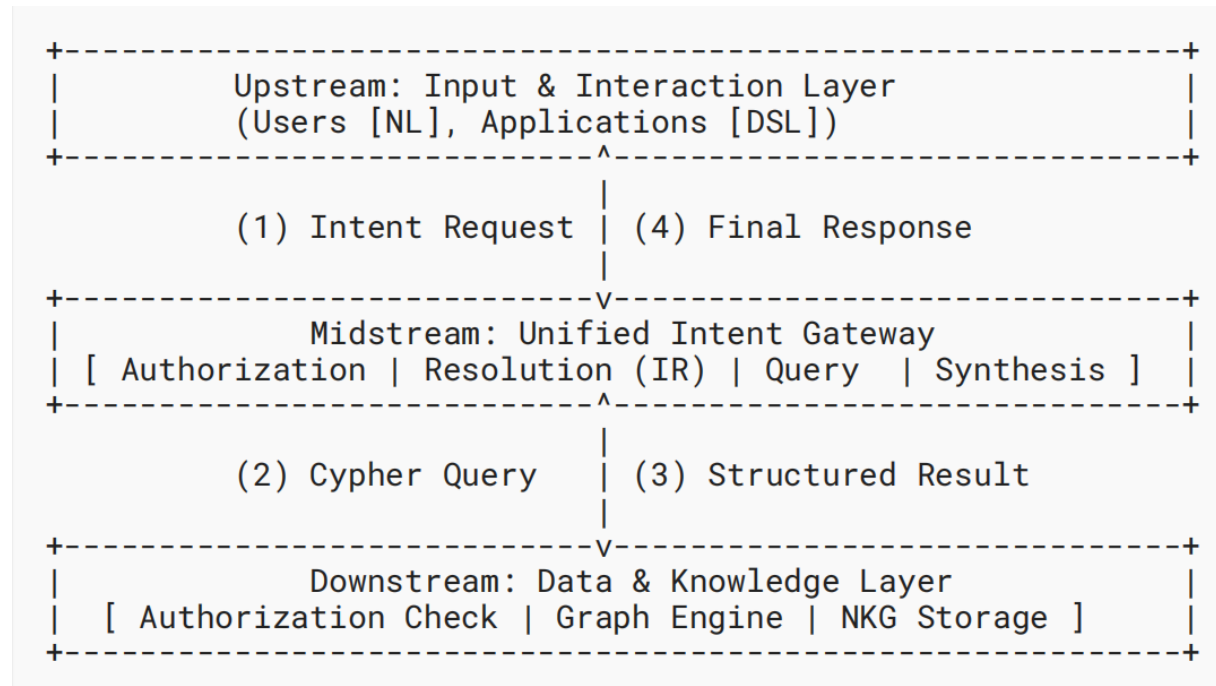
- Engineers primarily interact with the NetKG using **structured query languages (e.g., Cypher, SPARQL)**, which presents challenges:
  - **High Learning Barrier:** Users must understand the underlying graph schema, relationship definitions, and query syntax.
  - **Schema Evolution:** As graph modeling evolves, labels, relationships, and properties are frequently renamed, split, or merged, complicating query maintenance.
  - **Intent Abstraction & Semantic Validation:** The gap between high-level operational intent and low-level queries hinders automation and security governance.

# Requirements

- An intelligent interface capable of converting **human-intent natural language (NL)** or application-driven **domain-specific language (DSL)** instructions into queries and operations for the NetKG.
  - **Unify Heterogeneous Inputs:** Integrate and process instructions from both users (Natural Language) and applications (Domain-Specific Language).
  - **Ensure Security & Compliance:** Enforce strict identity authentication, authorization, and operational compliance for all actions.
  - **Accurate Intent-to-Query Conversion:** Precisely translate ambiguous or high-level intents into exact, executable graph queries.
  - **Synthesize Query Results:** Transform structured graph data outputs into comprehensible, user-friendly formats.

# Framework

- Three-layer architecture:
  - Input and Interaction Layer
  - Unified Intent Gateway
  - Data and Knowledge Layer

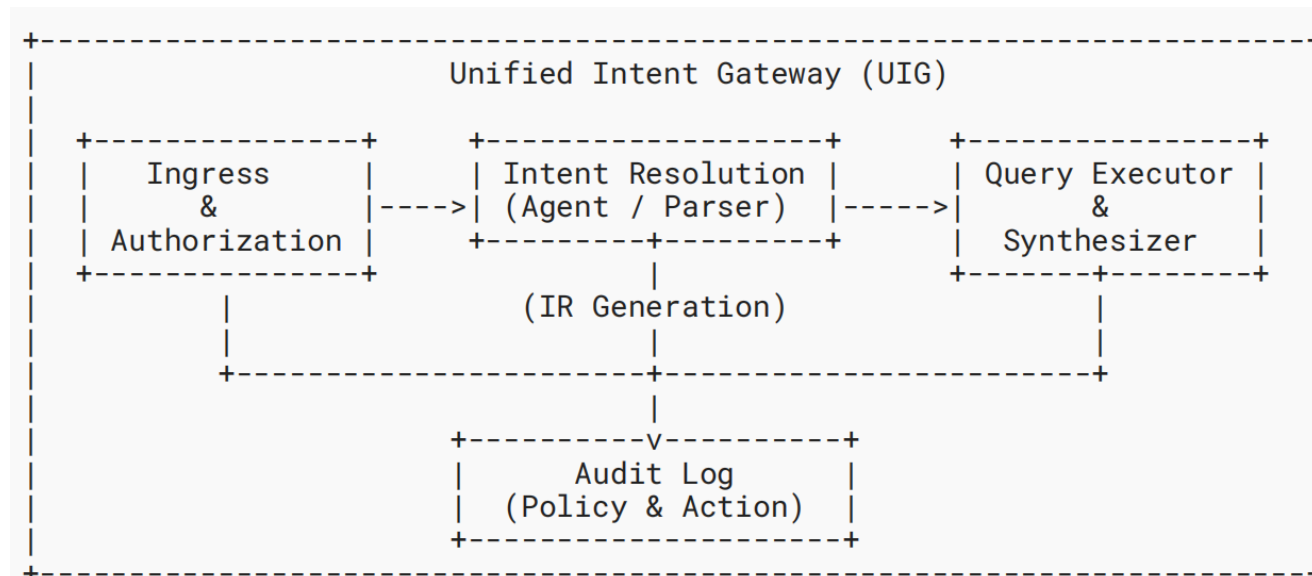


# Input and Interaction Layer

- Two Types of Input
  - **Natural Language (NL):** e.g., *"List nodes with abnormal traffic in the past five minutes."*
  - **Application/DSL:** e.g., `SELECT_TRAFFIC(srcIP=10.0.0.1, window=5m)`
- Interface Responsibilities
  - **Provide Unified APIs:** Offer standard interfaces such as REST, gRPC, and WebSocket.
  - **Transmit Enriched Requests:** Forward requests that include identity (e.g., user/administrator) and contextual metadata.
  - **Deliver Structured Results:** Return query outcomes (e.g., in JSON) paired with natural language explanations.

# Unified Intent Gateway

- Request Processing Pipeline
  - Receives incoming requests.
  - Performs authentication and authorization.
  - Parses intent and normalizes requests into an Intermediate Representation (IR).
  - Generates executable graph query/operation statements (e.g., in Cypher).



# Data and Knowledge Layer

- NetKG & Graph Database Layer
  - Executes the generated graph queries/operations.
  - Returns query results in a structured format (e.g., head, relation, tail triples).
- Security Enforcement:
  - Performs identity authentication for the querying entity.
  - Applies fine-grained access control on nodes and edges.

# Use Cases

- **Use Case 1: HTTP Flood Attack Investigation (NL)**

- **Scenario:** A service is experiencing abnormal response latency. Operations personnel suspect an HTTP Flood attack.
- **Input (NL):** "Please check if the target service is under an HTTP Flood attack."

```
MATCH (src:IP)-[r:SENDS_HTTP]->(dst:Server)
WHERE r.request_rate > $threshold AND dst.port = 80 AND r.window = "5m"
RETURN src, r.request_rate, dst
ORDER BY r.request_rate DESC
```

# Use Cases

- **Use Case 1: Dynamic NetKG Management (DSL)**

- **Scenario:** A network administrator dynamically maintains the NetKG by creating, updating, or deleting graph nodes.

- **Input (DSL):**

- UPSERT\_NODE(type="Device", key="name", value="FW1", props={interface:"eth0", status:"active"})
- DELETE\_NODE(type="Device", key="name", value="FW1")

```
</> cypher
```

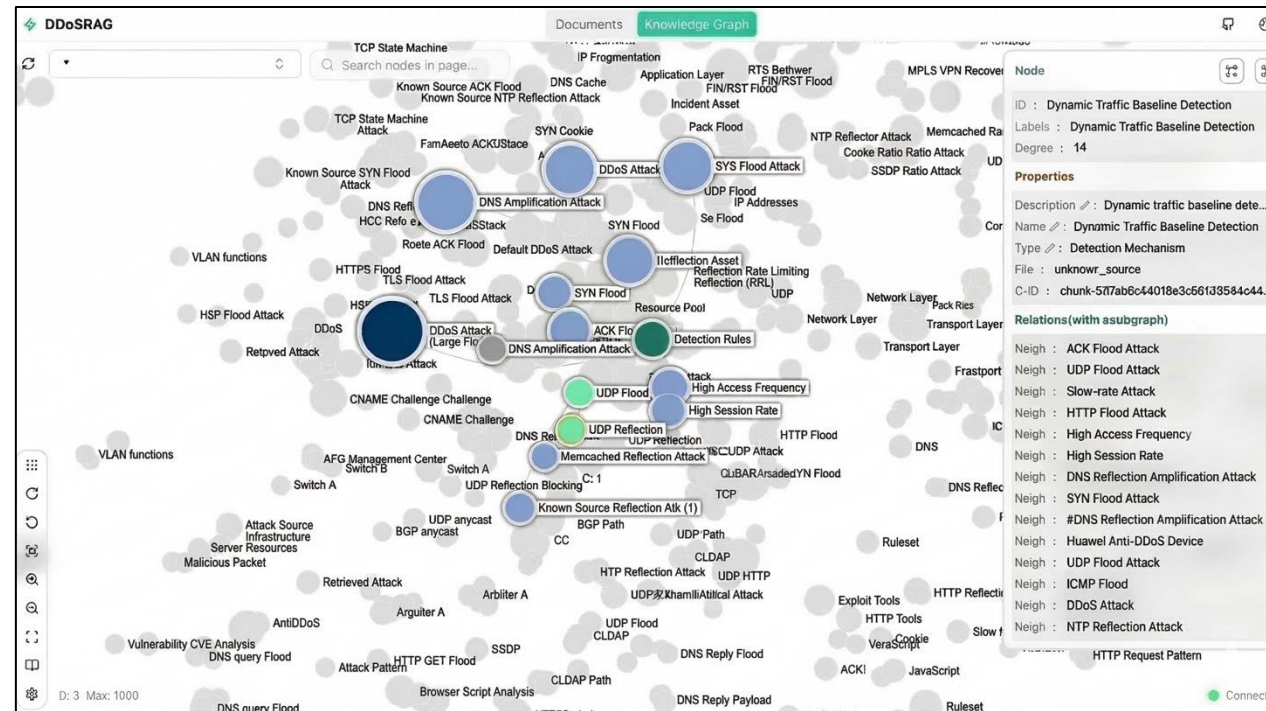
```
MERGE (n:Device {name:"FW1"})  
SET n.interface="eth0",  
    n.status="active";
```

```
</> cypher
```

```
MATCH (n:Device {name:"FW1"})  
DELETE n;
```

# Hackathon Project

## Net KG



## Retrieve Interface

How to defend against SYN Flood attacks?

- Use SYN Cookie technology to avoid maintaining half-open connection queues on the server.
- Implement rate limiting on high-risk source IPs to control the flow of SYN packets.
- Deploy specialized hardware defense systems, such as Huawei Anti-DDoS devices, for high-performance traffic cleaning.
- Apply TCP source challenge authentication to verify the legitimacy of the source IP address.
- Utilize protocol filtering with predefined rules to strictly control the reception of SYN packets.