

NeoTec
Internet-Draft
Intended status: Informational
Expires: 16 October 2025

L. Dunbar, Ed.
Futurewei
N. Cocker, Ed.
RedHat
C. Xie
China Telecom
14 April 2025

Neotec API Strategy and IETF Work Plan
draft-discussion-neotec-api-strategy-00

Abstract

This document outlines a strategy for standardizing the Neotec (NetOps4Cloud) API through the IETF, modeled after the successful approach used in the GROW Working Group's Peering API draft. The objective is to define a vendor-neutral, cloud-provider-agnostic, and secure interface that enables real-time coordination between cloud service scaling events and dynamic network policy adjustments by network controllers.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 16 October 2025.

Copyright Notice

Copyright (c) 2025 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights

and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction: What the APIs Are For	2
2. Requirements Language	3
3. Why YANG Alone Is Not Enough	3
4. Justification for IETF Standardization	4
5. What Will Be Standardized by IETF	4
6. Sample Workflow	6
7. Use Cases	6
7.1. Assist Cloud Platform in Making Connection Setup and Workload Deployment Decisions	6
7.2. Enable the Network Operation to be Cloud-aware and Better Facilitate Real-time Policy Adjustments	7
8. Security Considerations	8
9. IANA Considerations	8
10. References	8
10.1. Normative References	8
10.2. Informative References	8
Acknowledgements	9
Contributors	9
Authors' Addresses	9

1. Introduction: What the APIs Are For

The Neotec APIs are designed to bridge the operational gap between cloud infrastructure and network controllers. As cloud services (such as video, AI inference, and microservices) scale dynamically across multiple edge or core cloud locations, these APIs allow the network to respond intelligently and in real time. The APIs provide a standardized interface for:

- Communicating cloud service scaling events to the network.
- Exchanging metrics such as available compute, storage, or latency targets.
- Informing routing decisions like UCMP weight adjustments, flow steering, or bandwidth allocation.

These APIs will be used by:

- A cloud-aware shim or middleware deployed alongside telecom-managed cloud orchestration platforms (e.g., Kubernetes-based environments).

- A network controller that ingests the standardized metrics and adjusts network policies.

The goal is to ensure SLA guarantees and efficient routing without exposing internal details of cloud platforms or requiring tight integration between cloud and network domains. This approach is particularly important in environments where Kubernetes is used as a core component for building telecom 5G edge cloud infrastructure.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Why YANG Alone Is Not Enough

While YANG models are a critical part of IETF's toolset and have seen broad adoption for network configuration and telemetry, they are not sufficient on their own to support Neotec's goal of real time coordination between cloud orchestration systems and network controllers.

Modern cloud platforms and orchestrators, (including Kubernetes, OpenStack, and major cloud provider platforms), do not natively consume YANG models. Instead, they use well established HTTP based APIs (often RESTful) with JSON or gRPC payloads to expose and consume resource and service information.

Therefore, while YANG can define the semantic model for what information needs to be exchanged (e.g., service scaling status, policy updates), it must be wrapped in API endpoints and message formats that are compatible with:

- Kubernetes Custom Resource Definitions (CRDs).
- REST APIs in OpenAPI/Swagger format.
- gRPC or protobuf-based APIs for performance-sensitive use cases.

The Neotec API strategy proposes:

- Defining the canonical data models in YANG (standardized at IETF).
- Exposing those models through OpenAPI-defined REST interfaces (hosted on GitHub).

- Providing adapters or shim layers that convert between YANG and native Kubernetes or cloud orchestrator formats.

This ensures IETF-driven standardization of semantics while enabling practical integration with the systems operators are already using.

4. Justification for IETF Standardization

Standardizing the Neotec APIs through the IETF ensures consistent behavior, semantics, and interoperability for managing dynamic interactions between cloud-based services and the underlying network infrastructure. While the IETF has historically focused more on protocols than on APIs in the software engineering sense, this work follows well-established IETF precedent for defining standardized interfaces between systems operated by different administrative entities. Interfaces like BGP, NetConf, and various YANG models are not APIs in the REST sense, but they do define how two systems interact, exchange information, and manage configuration securely and predictably.

Importantly, the IETF has begun to acknowledge and support more API-centric efforts where interoperability and automation are essential to network operations. The Peering API in the GROW working group [BGP-PEERING-API] is a clear example. In these contexts, the "API" is not simply an application development convenience, it is a standards based interface for operational automation across autonomous systems or loosely coupled administrative domains.

The Neotec APIs similarly require this level of standardization. As network operators increasingly deploy applications in distributed telecom clouds and multi vendor edge environments, they need standardized, secure, and verifiable ways to exchange resource and scaling information. Without this, SLAs and real time optimization are impossible to maintain. The IETF's strong foundation in modeling (via YANG), its security practices (OAuth2, RPKI, JWT), and its global credibility make it the ideal home for formalizing the Neotec API strategy.

5. What Will Be Standardized by IETF

API Behavior and Semantics:

- Event-driven coordination between cloud telemetry and network policies.
- Required fields, expected flows, and error handling logic for REST endpoints.

Security and Trust Models:

The security and trust model for Neotec APIs is grounded in Zero Trust principles, ensuring that all access is authenticated, authorized, and continuously verified. Instead of relying on static trust anchors like ASN validation (which is specific to BGP-related APIs), Neotec adopts a more flexible and scalable approach aligned with modern service infrastructure.

Key elements include:

- OAuth2 and OIDC for token based authentication and identity federation.

- JSON Web Tokens (JWTs) for signed claims and scoped access to enable secure, stateless authentication and fine grained access control.

JWTs carry signed claims about the requester's identity, permissions, and token validity, allowing each API request to be independently verified. This supports Zero Trust principles by ensuring that only authorized components can trigger network policy changes - without relying on centralized lookups.

- Support for DPOP or HTTP Message Signatures for request validation, which provides an additional layer of security by binding each API request to the sender's cryptographic key.

- Policy driven access control based on dynamic service and identity attributes.

This aligns with the architectural direction outlined in [Neotec-Zerotrust-Access], which explores Zero Trust access management for distributed edge-to-cloud network operations.

API Interface Definitions:

Example endpoints to be defined:

- POST /cloud-events: Notify network of a cloud scaling event.
- GET /cloud-status: Query current service instances and pressure.
- PUT /ucmp-policy: Update traffic routing policies.
- GET /policy-state: Retrieve current network policy for a service.

What Will Be Hosted on GitHub:

6. Sample Workflow

7. Use Cases

7.1. Assist Cloud Platform in Making Connection Setup and Workload Deployment Decisions

Problem to be solved: Currently, cloud connectivity services mainly rely on overlay solutions and remain loosely coupled with the underlying network capabilities. Without visibility into the underlay network, the cloud manager cannot make optimal workload deployment decisions or dynamically adjust connections to meet service-level objectives (SLOs).

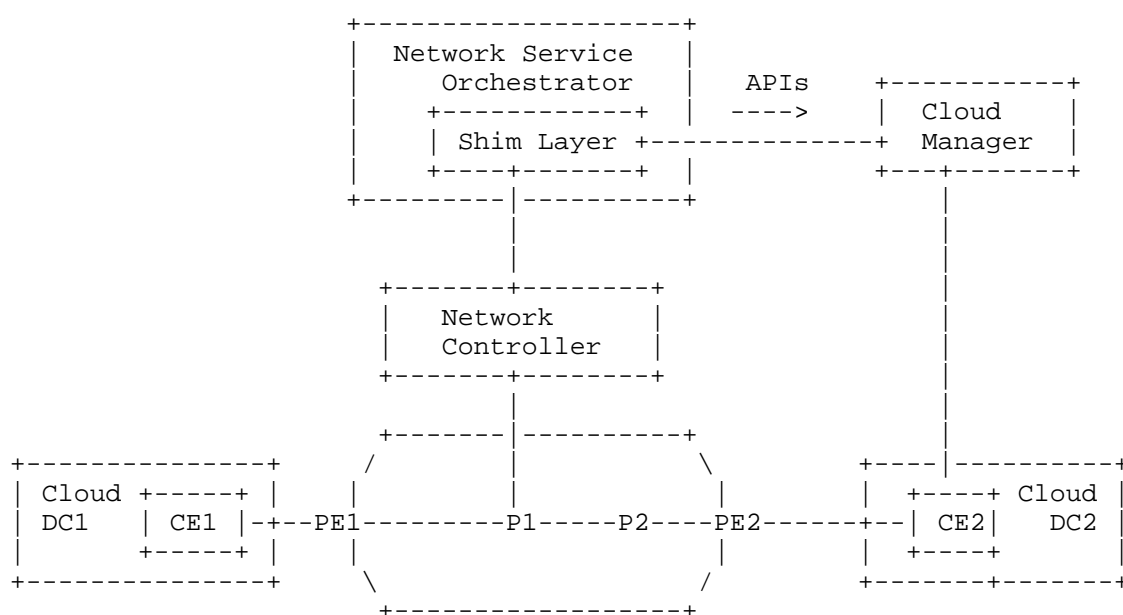


Figure 1. Neotec APIs for Resources Scheduling at the Cloud Side

In this case, the decision and scheduling point is at the cloud side, e.g., Cloud Manager. With the new Neotec APIs, network information, e.g., Attachment Circuit(AC), is exposed to the cloud. This enables the Cloud Manager to make deployment decisions based not only on compute resources, but also on public communication resources. As a result, Cloud Manager can perform placement decisions on the cloud side during scale-up events, update network resources (e.g., bandwidth) for existing paths.

7.2. Enable the Network Operation to be Cloud-aware and Better Facilitate Real-time Policy Adjustments

Problem to be solved: Without standardized interfaces to the cloud manager, Network Service Orchestrator cannot dynamically adjust cloud-aware service policies, or bandwidth to meet SLA requirements when cloud services scale. Neotec API bridges this gap, enabling the network to react to cloud events dynamically.

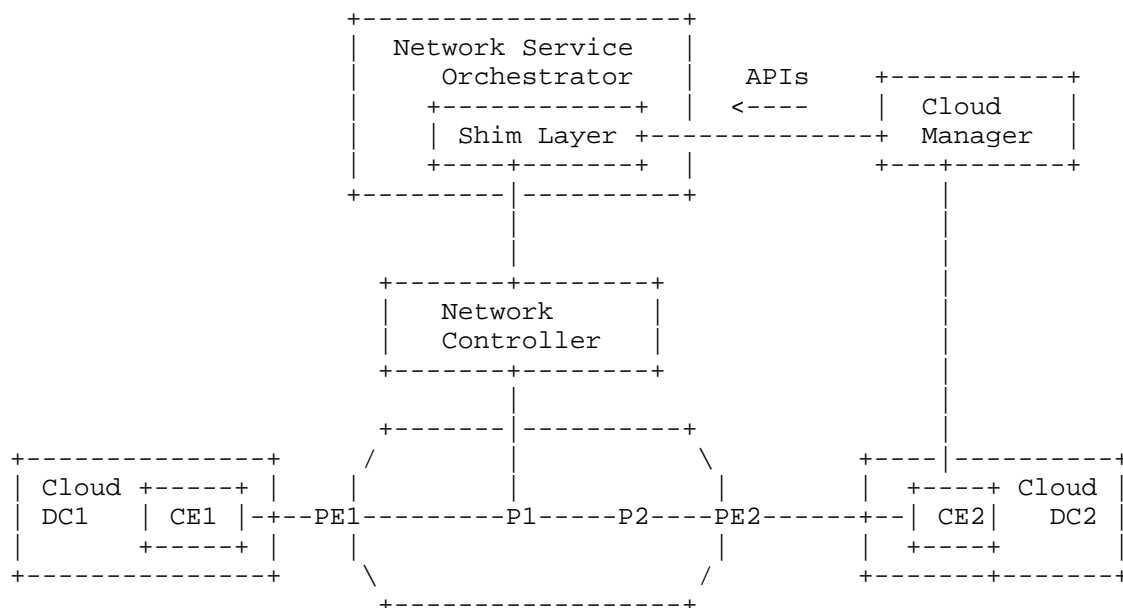


Figure 2. Neotec APIs for Resources Scheduling at the Network Side

In this case, the decision and scheduling point is at the Network Service Orchestrator. With the new APIs, cloud information is exposed to the network service orchestrator, enabling the network operating system to become cloud-aware. Instead of attempting to fully unify cloud models, a minimal set of high-level, cloud-derived metrics, such as service scaling events, bandwidth demands, and latency requirements, is shared with the network layer via standardized APIs. A shim layer translates existing telemetry (from systems like CloudWatch or Azure Monitor) into formats usable by network controllers, facilitating real-time policy adjustments such as dynamic UCMP (Unequal Cost Multi-Path) routing updates. In this case, a joint cloud and network model may be needed (Luis has a individual draft draft-llc-teas-dc-aware-topo-model), based on the cloud and network information collected, the system can automatically schedule cloud and network resource after computation by specific algorithm.

8. Security Considerations

9. IANA Considerations

None

10. References

10.1. Normative References

10.2. Informative References

[BGP-PEERING-API]

C. Aguado, et al, "BGP-PEERING-API", December 2024, <<https://datatracker.ietf.org/doc/html/draft-ietf-grow-peering-api-00>>.

[Neotec-Zerotrust-Access]

L. Dunbar and H. Chihi, "Neotec-Zerotrust-Access", December 2024, <<https://datatracker.ietf.org/doc/draft-dunchihi-neotec-zerotrust-access-dm/>>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

Acknowledgements

The authors would like to thank for following for discussions and providing input to this document: xxx.

Contributors

Authors' Addresses

Linda Dunbar (editor)
Futurewei
United States of America
Email: ldunbar@futurewei.com

Nabeel (editor)
RedHat
United States of America
Email: ncocker@redhat.com

ChongFeng Xie
China Telecom
China
Email: xiechf@chinatelecom.cn