nmrg
Internet-Draft
Intended status: Informational
Expires: 1 December 2026

K. Xu
Tsinghua University & Zhongguancun Laboratory
L. Zhu
Zhongguancun Laboratory
X. Gao
Y. Zhang
Tsinghua University

30 May 2026

Intelligent Data Plane (IDP): Framework and Protocol Considerations
draft-xu-nmrg-idp-framework-00

Abstract

This document describes a framework and protocol considerations for an Intelligent Data Plane (IDP). An IDP enables data-plane nodes to perform bounded, policy-constrained, and observable packet, flow, state, telemetry, or service processing based on standardized signals and locally executable decision functions.

The document defines terminology, a reference architecture, signal and result models, decision function abstractions, and communication patterns for IDP systems. It focuses on the interfaces and data formats through which IDP nodes expose signals, features, and results to AI for Network Management (AI-NM) systems, and through which they receive models, policies, and configuration from control and management entities in a self-driving and self-managing network (SD/MN) context.

This document does not define a specific AI or ML algorithm, nor does it require data-plane nodes to perform model training. Instead, it focuses on the interoperable communication patterns and data representations required to integrate intelligent data-plane processing into AI-NM and self-managing network architectures.

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/.

Copyright Notice

Table of Contents

1. Introduction 3
1.1. Relationship to Existing RFCs 4
1.2. AI Network Management and Self-Managing Networks Context 6
2. Requirements Language 6
3. Terminology 6
4. Scope 7
4.1. In Scope 7
4.2. Out of Scope 8
5. Relationship to AI Network Management and Self-Managing Networks 8
5.1. The AI-NM and SD/MN Closed Loop 8
5.2. IDP as the Data-Plane Interface for AI Network Management 9
5.3. Three IDP Interfaces in the AI-NM and SD/MN Context . . . 9
6. Reference Architecture 11
7. IDP Signals, Features, and Results 13
7.1. Signal Categories 13
7.2. Feature Representation 13
7.3. Result Model 14
7.4. Signal Freshness and Confidence 15
8. Decision Function Abstraction 15
8.1. Generalized Processing Primitives 15
8.2. Hardware Mapping Examples 16
8.3. Model Representation and Versioning 16
9. IDP Communication Patterns 17
9.1. Northbound Communication (IDP Node to AI-NM and Management Systems) 17

9.2. East-West Communication (IDP to IDP) 18
9.3. Internal Communication (Forwarding Engine to Accelerator) 18
10. Collaborative and Escalated Processing 18
10.1. Hybrid Local and Escalated Processing 19
10.2. Collaboration with External Analytics Platforms 19
11. Data-Plane Processing Procedure 20
12. State Management 20
13. Metadata and Protocol Mapping Considerations 21
14. Requirements 22
14.1. Signal and Feature Requirements 22
14.2. Decision Function Requirements 23
14.3. Communication and Collaboration Requirements 23
14.4. State Management Requirements 24
14.5. Security and Privacy Requirements 24
15. Manageability Considerations 25
16. OAM Considerations 25
17. Performance Considerations 25
18. Security Considerations 26
18.1. Signal Spoofing and Poisoning 26
18.2. Model and Decision-Function Integrity 26
18.3. Resource Exhaustion 26
18.4. Evasion and Adversarial Traffic 26
18.5. Policy Bypass 26
18.6. Metadata Integrity 27
18.7. Safe Mode 27
19. Privacy Considerations 27
20. IANA Considerations 27
21. References 27
21.1. Normative References 27
21.2. Informative References 28
Acknowledgments 29
Authors' Addresses 29

1. Introduction

Network infrastructure is evolving from a model centered primarily on connectivity and forwarding toward a model that also includes in-network sensing, feature extraction, state maintenance, telemetry preprocessing, conditional triggering and classification, and coordination with controllers, management systems, and compute platforms.

Programmable data planes, network telemetry, in-situ OAM, network service chaining, computing-aware networking, and switch-attached accelerators have enabled forwarding devices to perform increasingly sophisticated processing while packets traverse the network. Recent research systems have demonstrated multiple implementation

directions, including tree-based data-plane inference [NetBeacon], neural network traffic analysis in programmable switches [BoS], deep learning execution on match-action pipelines [Pegasus], and FPGA-enhanced programmable switches for in-network DNN inference [FENIX]. These systems demonstrate that intelligent data-plane processing is becoming technically feasible, while also revealing common issues around state management, resource limits, safety, fallback, capability exposure, metadata exchange, and operational control.

1.1. Relationship to Existing RFCs

This document builds upon existing IETF and IRTF work. The relationship to relevant RFCs is summarized below.

RFC	Contribution	IDP Extension
[RFC7426]	SDN layers and architecture terminology (data, control, management planes)	IDP defines internal data-plane components for intelligent processing and their interfaces to control and management planes
[RFC9232]	Network telemetry framework with measurement, export, and analysis modules	IDP defines signal and result models for telemetry preprocessing, online feature extraction, and local decision execution at the data plane
[RFC9197]	In-situ OAM (IOAM) data fields for in-band telemetry collection	IDP signals and results map onto IOAM, extending its semantics to carry feature values, inference results, and confidence indicators
[RFC8300]	Network Service Header (NSH) for service function chaining metadata	IDP metadata can be carried as NSH context; IDP extends NSH with new metadata types for feature and result exchange
[RFC8402] [RFC8986]	Segment Routing architecture and SRv6 network programming	IDP decision functions can be triggered per-SR-hop; IDP processing can be encoded as SRv6 network program segments
[RFC7011]	IPFIX protocol for flow information export	IDP results, telemetry summaries, and decision traces can be exported via IPFIX
[RFC9637]	In-network computing use cases for the IRTF coinrg	IDP provides a framework for the intelligent data-plane subset of in-network computing use cases

Table 1

1.2. AI Network Management and Self-Managing Networks Context

The IRTF Network Management Research Group (NMRG) has identified three core research areas: Intent-Based Networking (IBN), AI for Network Management (AI-NM), and Self-Driving and Self-Managing Networks (SD/MN). This document positions the Intelligent Data Plane as a foundational enabler for both AI-NM and SD/MN:

* *For AI-NM*: IDP nodes provide standardized signals, features, and telemetry that serve as inputs to AI management and analytics systems. Without a standardized interface at the data plane, AI-NM systems rely on ad-hoc data collection mechanisms that limit interoperability.

* *For SD/MN*: IDP nodes provide bounded local decision execution, conditional triggering, and configurable fallback behavior that constitute the "act" and "enforce" components of the autonomous network control loop. This enables closed-loop operations at the data-plane level under management-plane policy.

This document does not standardize AI-NM or SD/MN architectures themselves; it focuses on the data-plane interfaces, data formats, and communication patterns that make these architectures feasible.

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. Terminology

This document uses terminology from [RFC7426] for SDN architectural concepts, [RFC9232] for network telemetry concepts, and [RFC9637] for in-network computing use cases. It also uses the following terms.

Intelligent Data Plane (IDP): A data plane that supports bounded, policy-constrained, and observable processing of packets, flows, states, telemetry, or service-related signals in order to assist forwarding, classification, monitoring, mitigation, steering, or coordination.

IDP Node: A forwarding node that implements IDP capabilities. Examples include programmable switches, FPGA-enhanced switches, SmartNIC-based forwarding nodes, switch-attached accelerators, and other data-plane devices.

IDP Domain: An administrative domain in which IDP capabilities, policies, metadata, and decision behavior are configured, authorized, and managed under a common trust and policy model.

IDP Signal: An input used by an IDP decision function or processing function. Signals can include packet header fields, metadata, flow statistics, queue state, link state, buffer state, telemetry, service state, compute state, model confidence, or operator-provided policy parameters.

Feature: A derived value extracted from one or more IDP signals for use by a decision function, for export to a management or analytics system, or for collaborative processing between IDP nodes.

Decision Function: A bounded function executed by an IDP node or an attached processing module that maps signals and policy to an output result. A decision function can use rules, thresholds, lookup tables, model inference, or hybrid methods.

Result: The output of a decision function, which may include a classification label, anomaly indication, confidence value, action hint, telemetry summary, or decision trace identifier.

Online Sensing: The ability of an IDP node to acquire and identify packet, flow, queue, buffer, link, event, or resource information while traffic is being forwarded.

Safe Mode: A fallback operating mode in which an IDP node continues basic forwarding or a configured conservative action when an intelligent processing function, model, signal source, metadata channel, accelerator, or external system is unavailable or untrusted.

Decision Trace: An observable record that describes the inputs, policy, decision function identifier, model version, confidence value, output result, and selected action associated with a decision, subject to security and privacy policy.

4. Scope

4.1. In Scope

This document covers:

* terminology for IDP systems in the context of AI-NM and SD/MN;

* a reference architecture for IDP nodes and their interfaces to AI-NM and management systems;

* signal and result models, including feature representation and confidence encoding;

* decision function abstractions that generalize across hardware platforms;

* communication patterns for northbound, east-west, and internal IDP interactions;

* requirements for IDP signaling, data representation, and protocol mappings.

4.2. Out of Scope

This document does not:

* define a specific AI, ML, neural network, decision tree, or inference algorithm;

* require a forwarding node to train models;

* define a new packet header, TLV, YANG model, or IANA registry;

* require inspection or decryption of encrypted payloads;

* define implementation details for ASICs, NPUs, FPGAs, P4, eBPF, SmartNICs, or vendor SDKs;

* define AI-NM or SD/MN system architectures.

5. Relationship to AI Network Management and Self-Managing Networks

This section describes how the Intelligent Data Plane framework relates to AI for Network Management (AI-NM) and self-managing networks (SD/MN), as defined by the IRTF NMRG.

5.1. The AI-NM and SD/MN Closed Loop

An AI-enabled self-managing network operates as a closed control loop:

1. *Observe:* Collect telemetry, signals, and state from the network.

2. *Analyze:* Apply AI or ML models to detect patterns, anomalies, or optimization opportunities.

3. *Decide:* Determine the appropriate action or policy adjustment.

4. *Act:* Execute the decision through network configuration or data-plane control.

IDP contributes to the *Observe* and *Act* phases by providing standardized interfaces at the data plane.

5.2. IDP as the Data-Plane Interface for AI Network Management

In the Observe phase, an AI-NM system requires data from the network. Current deployments rely on device-specific telemetry, ad-hoc log collection, or external probe systems. IDP defines a standardized approach:

* *Online Sensing:* IDP nodes acquire packet, flow, queue, and device-state signals during forwarding.

* *Feature Extraction:* IDP nodes derive structured features from raw signals, producing data that AI models can consume directly.

* *Result Export:* IDP nodes export inference results, telemetry summaries, and decision traces to AI-NM analyzers.

In the Act phase, an SD/MN system requires the ability to enforce decisions at the data plane:

* *Conditional Triggering:* IDP nodes execute local actions when configured conditions are met.

* *Local Decision Functions:* IDP nodes perform bounded inference under management-plane policy.

* *Safe Fallback:* IDP nodes revert to basic forwarding when intelligent functions are unavailable.

5.3. Three IDP Interfaces in the AI-NM and SD/MN Context

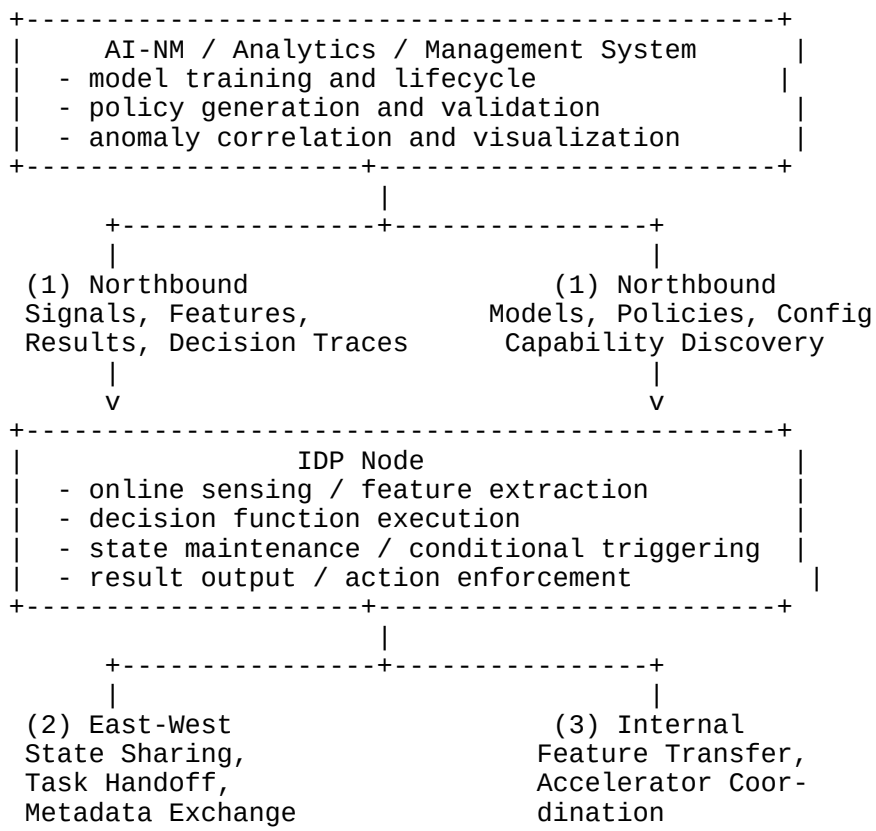

Figure 1: IDP Interfaces in the AI-NM and SD/MN Context

Three interface categories are identified:

1. *Northbound Interface (IDP with AI-NM and Management Systems):* Used for exporting signals, features, and results upward, and for receiving models, policies, and configuration downward.

2. *East-West Interface (IDP to IDP):* Used for sharing state, handing off tasks, and exchanging metadata between peer IDP nodes within the same IDP domain.

3. *Internal Interface (Forwarding Engine to Accelerator):* Used for feature vector transfer, model parameter loading, and resource coordination between the primary forwarding pipeline and an attached accelerator.

6. Reference Architecture

The logical architecture of an IDP system is shown below.

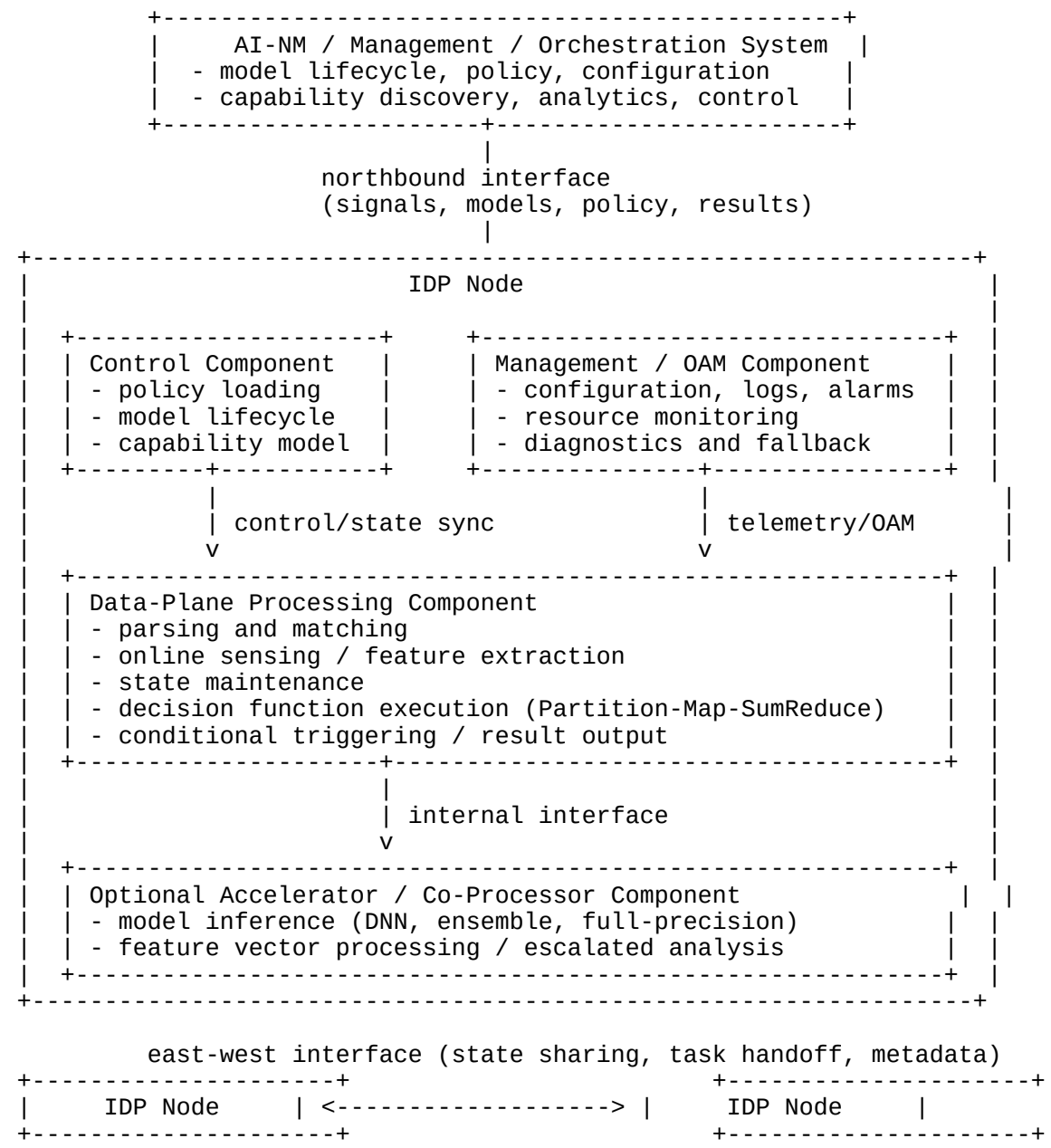

Figure 2: IDP Reference Architecture

The architecture is logical. An implementation can combine components or distribute them across hardware and software elements. For example, a switch-only implementation can execute all decision functions in a match-action pipeline. A hybrid implementation can perform feature extraction in a switch ASIC and model inference in an attached FPGA or other accelerator.

7. IDP Signals, Features, and Results

This section describes the data model for IDP signals, features, and results. These are the data units exchanged across the northbound, east-west, and internal interfaces defined in Section 5.

7.1. Signal Categories

IDP signals can include:

* *Packet signals:* header fields, packet length, labels, tunnel identifiers, ingress port, egress port, timestamp, metadata, or classification labels.

* *Flow signals:* packet count, byte count, inter-packet delay, flow duration, windowed counters, flowlet information, or flow state.

* *Queue and buffer signals:* queue depth, drop counters, ECN marks, buffer occupancy, scheduling state, or congestion indication.

* *Link and path signals:* link utilization, delay, loss, failure indication, path identifier, or path quality.

* *Device signals:* CPU, memory, table occupancy, accelerator load, chip resource pressure, temperature, or error counters.

* *Service and compute signals:* service instance health, compute load, accelerator availability, job state, or service capacity.

* *Model and decision signals:* model identifier, model version, confidence value, ambiguity indication, escalation state, or result age.

7.2. Feature Representation

A feature is a derived value computed from one or more signals for use by a decision function or for export. An IDP feature representation SHOULD include:

* *Feature identifier:* a unique identifier within the scope of the applicable IDP policy.

* *Feature type:* indicating the data type (e.g., uint8, uint32, float16, boolean, enumerated, counter, rate).

* *Feature value:* the extracted value, encoded according to the feature type.

* *Optional metadata:* feature name, derivation description, extraction timestamp, freshness interval.

Feature extraction MUST NOT require decryption of encrypted payloads. Feature types SHOULD support the range and precision required by the configured decision function while respecting the resource limits of the IDP node.

7.3. Result Model

An IDP result is the output of a decision function. A result SHOULD include:

* *Result identifier:* a unique identifier within the scope of the applicable policy.

* *Decision function identifier:* identifies the function that produced the result.

* *Model version:* if the decision function uses a versioned model.

* *Outcome:* classification label, anomaly indication, or action hint.

* *Confidence value:* an optional indicator of result reliability, expressed as a value in a defined range (e.g., 0-100).

* *Timestamp or age:* indicates when the result was produced.

* *Decision trace identifier:* optional reference to a detailed decision trace.

* *Scope:* indicates the packet, flow, session, or other entity to which the result applies.

A result used for forwarding or filtering MUST be interpreted under the applicable IDP policy. Results SHOULD be treated as hints unless policy explicitly defines them as authoritative.

7.4. Signal Freshness and Confidence

An IDP signal can become stale. A signal used for IDP processing SHOULD have an associated freshness rule, such as a timestamp, sequence number, age, timeout, or validity interval. An IDP node MUST NOT use a stale signal for a decision that affects forwarding, filtering, or service steering unless local policy explicitly permits such use.

An IDP signal MAY have an associated confidence value or quality indicator. Confidence can be derived from measurement accuracy, sampling rate, model confidence, signal source trust, or freshness. If confidence is used to influence traffic treatment, the IDP node SHOULD expose the confidence semantics to the AI-NM or management system.

8. Decision Function Abstraction

An IDP decision function maps signals and policy to an output result. This section describes a generalized abstraction for IDP decision functions, derived from published research systems but applicable across heterogeneous data-plane hardware.

8.1. Generalized Processing Primitives

The processing performed by an IDP decision function can be modeled as a composition of three primitive operations:

Partition: Divides the input space (signals, features, state) into manageable segments. Partitioning can be based on flow keys, time windows, packet classes, or policy-specified criteria. The result is a set of independent processing contexts that can be assigned to available compute resources.

Map: Applies a processing function to each segment independently. The mapping function can use rules, thresholds, model inference, or other methods (e.g., lookup tables, match-action entries, comparisons against learned values).

SumReduce: Aggregates results from multiple mapping operations into a consolidated output. Aggregation can include voting, averaging, confidence-weighted fusion, max-pooling, or policy-specified combination rules.

This Partition-Map-SumReduce model is hardware-agnostic. The same logical decision function can be realized on different hardware platforms through different mappings of the three primitives.

8.2. Hardware Mapping Examples

Hardware Platform	Partition	Map	SumReduce
P4/Tofino	Pipeline stage allocation; flow hashing	Match-action tables; register operations	Cross-stage register aggregation; control-plane readout
FPGA (Xilinx ZU19EG)	Parallel pipeline partitions	Custom logic blocks; DSP slices	BRAM/URAM accumulation trees
SmartNIC/ DPU (NVIDIA BlueField)	ARM core or HW thread assignment	ML inference via ONNX Runtime	Shared memory or inter-core communication
eBPF/XDP	CPU core affinity; flow hashing	BPF map lookups; helper functions	BPF map value aggregation
NPU (Broadcom NetGNT)	Hardware thread scheduling	Micro-engine instructions	Distributed memory aggregation

Table 2

The table above is illustrative. Specific product capabilities vary by generation and vendor.

8.3. Model Representation and Versioning

An IDP node that supports model-based decision functions MUST provide a way to identify the active model, its version, its resource requirements, and its allowed output actions. A model identifier SHOULD include:

* *Model name or identifier:* unique within the IDP domain.

* *Version number:* monotonically increasing or semantically versioned.

* *Model type:* decision tree, random forest, neural network (MLP, RNN, CNN), support vector machine, or other type.

* *Primitive mapping:* how the model maps onto Partition, Map, and SumReduce primitives on the target hardware.

* *Input specification:* the expected feature vector format, size, and data types.

* *Output specification:* the result format, action space, and confidence semantics.

* *Resource profile:* estimated table entries, memory, processing stages, and latency impact.

A decision function SHOULD be bounded in execution time, memory use, and data-plane side effects.

9. IDP Communication Patterns

This section describes the communication patterns across the three interface categories defined in Section 5.

9.1. Northbound Communication (IDP Node to AI-NM and Management Systems)

Northbound communication includes the following patterns:

Capability Advertisement (IDP to AI-NM): An IDP node exposes its capabilities to the AI-NM or management system, including supported parsing, feature extraction types, state scopes and limits, decision function types, action types, result output methods, model versioning support, resource limits, and fallback modes.

Policy and Model Distribution (AI-NM to IDP): The AI-NM or management system installs or updates IDP policies, decision function parameters, models, thresholds, and action maps on the IDP node. The IDP node SHOULD validate resource requirements before activation and MUST reject a task that cannot be safely installed.

Signal and Result Export (IDP to AI-NM): The IDP node exports features, inference results, telemetry summaries, event reports, and decision traces to the AI-NM or analytics system. Export SHOULD be configurable by policy and MAY be filtered, sampled, or aggregated to respect bandwidth and resource limits.

Fallback and Lifecycle Control (AI-NM to IDP): The AI-NM or management system can disable specific IDP functions, enter safe mode, trigger model rollback, or initiate diagnostics. The IDP node MUST report fallback events and state changes to the management system.

9.2. East-West Communication (IDP to IDP)

East-west communication includes the following patterns:

State Sharing: An IDP node MAY share flow state, model context, or decision context with a peer IDP node to support flow affinity, state consistency, or multi-hop processing.

Task Handoff: When a flow moves from one IDP node to another (e.g., due to routing changes or mobility), the first node MAY transfer relevant state, feature context, or partial results to the next node.

Metadata Exchange: IDP nodes MAY exchange result metadata, confidence values, or telemetry summaries for collaborative processing or multi-node aggregation.

East-west metadata MUST be scoped to an authorized IDP domain. A node that does not understand the metadata MUST have well-defined behavior, such as ignoring, forwarding unchanged, stripping, or dropping according to the protocol mapping.

9.3. Internal Communication (Forwarding Engine to Accelerator)

When an IDP node includes an optional accelerator or co-processor:

Feature Vector Transfer: The forwarding engine transfers extracted feature vectors to the accelerator for model inference or escalated analysis. The transfer MUST define format, size, rate limits, and timeout behavior.

Result Return: The accelerator returns inference results, confidence values, or classification labels to the forwarding engine for action execution. The return path MUST define timeout, failure, and fallback behavior.

Resource Coordination: The forwarding engine and accelerator coordinate on resource allocation, load balancing, and model loading. The IDP node MUST define behavior for accelerator overload, timeout, failure, restart, version mismatch, and result loss.

10. Collaborative and Escalated Processing

10.1. Hybrid Local and Escalated Processing

Some IDP systems perform most processing locally and escalate a subset of traffic to an attached accelerator or external analysis module. Escalation can be triggered by:

* low confidence in the local decision function result;
* ambiguous results near a classification boundary;
* resource shortage (e.g., state exhaustion, table overflow);
* policy-specified traffic classes requiring full-precision analysis;
* model update or version mismatch.

An IDP node that supports escalation MUST provide:

* an escalation policy specifying triggers and targets;
* a maximum escalation rate or budget;
* backpressure or rate limiting for escalated traffic;
* timeout behavior for escalation responses;
* result correlation with the original flow or packet context;
* fallback behavior when escalation is unavailable;
* counters for escalated packets and flows.

Escalation MUST NOT cause unbounded delay or resource exhaustion for unrelated traffic.

10.2. Collaboration with External Analytics Platforms

IDP nodes MAY collaborate with external AI-NM, telemetry, security, or analytics platforms. Such collaboration includes:

* exporting feature vectors for model retraining or refinement;
* exporting decision traces for audit, analysis, or explainability;
* receiving updated model parameters or thresholds based on global analysis;

* participating in federated or distributed analytics workflows.

All external collaboration MUST be subject to authorization, domain boundary, privacy, and policy constraints.

11. Data-Plane Processing Procedure

For each eligible packet or flow, an IDP node follows a bounded procedure such as the following:

1. Identify the applicable IDP policy.
2. Validate traffic eligibility and policy scope.
3. Parse required headers and metadata.
4. Retrieve or create the required state context.
5. Extract configured features using Online Sensing and Feature Extraction.
6. Check signal freshness, confidence, and resource limits.
7. Execute the configured Decision Function using Partition-Map-SumReduce primitives.
8. Compute result, confidence, and action.
9. Apply flow affinity or state consistency rules if required.
10. Execute the authorized action.
11. Export result, telemetry, or decision trace if configured.
12. Enter fallback behavior (Safe Mode) if required state, signals, resources, accelerators, or decision functions are unavailable.

The procedure above is logical. Implementations can realize it using tables, registers, counters, accelerators, software agents, or other mechanisms.

12. State Management

Stateful processing is central to many IDP functions. State resources in forwarding devices are limited. An IDP node MUST provide predictable state management behavior.

An IDP node SHOULD support:

* creation, update, lookup, aging, deletion, and query of state;
* per-flow or per-session state where required;
* state timeout and inactivity expiration;
* collision detection when hash-based indexing is used;
* safe eviction rules;
* fallback behavior when state cannot be allocated;
* counters for state allocation failures, collisions, overwrites, and evictions.

If state collision or exhaustion occurs, the IDP node MUST NOT corrupt unrelated forwarding state. The node SHOULD fall back to a configured safe behavior, such as per-packet processing, default forwarding, conservative classification, or controlled escalation.

13. Metadata and Protocol Mapping Considerations

This document does not define a new metadata encapsulation. Future protocol mapping documents can define how IDP signals, features, results, and metadata are carried using existing or new mechanisms.

Possible mapping targets include:

* *IOAM data fields or related OAM mechanisms [RFC9197]:* for in-band signal collection and feature export.
* *Service function metadata such as NSH context metadata [RFC8300]:* for carrying classification results and action hints along a service path.
* *Segment Routing or SRv6 policy-associated metadata [RFC8402][RFC8986]:* for per-hop IDP context and result propagation.
* *IPFIX or telemetry export records [RFC7011]:* for out-of-band export of features, results, and decision traces.
* *YANG-based configuration and operational state models:* for northbound capability advertisement, policy configuration, and model lifecycle management.

* *Controller-to-device programming protocols (e.g., P4Runtime, OpenConfig, NETCONF/YANG):* for model deployment and policy synchronization.

A protocol mapping that claims consistency with this framework MUST define:

* the metadata format and encoding;

* scope and lifetime of IDP metadata;

* processing behavior at capable and incapable nodes;

* MTU and fragmentation considerations;

* integrity and confidentiality protection;

* domain boundary behavior;

* privacy treatment;

* IANA considerations if any;

* failure and fallback behavior.

14. Requirements

14.1. Signal and Feature Requirements

REQ-IDP-01:
An IDP node MUST support a fallback behavior that does not depend on the availability of an intelligent decision function.

REQ-IDP-02:
An IDP node MUST support feature extraction based on packet headers, metadata, and flow-level state without requiring payload decryption.

REQ-IDP-03:
An IDP node SHOULD support queue, buffer, link, and device-state feature extraction when relevant to the configured policy.

REQ-IDP-04:
Feature representation SHOULD include a feature identifier, type, value, and optional freshness metadata for interoperability across IDP nodes and AI-NM systems.

REQ-IDP-05:

A signal used for IDP processing SHOULD have an associated freshness rule. A stale signal MUST NOT be used for forwarding-affecting decisions unless policy explicitly permits it.

14.2. Decision Function Requirements

REQ-IDP-06:
A decision function MUST be identifiable by a stable identifier and version.

REQ-IDP-07:
A decision function SHOULD be bounded in execution time, memory use, and data-plane side effects.

REQ-IDP-08:
An IDP node SHOULD support result confidence when the decision function can produce confidence. The confidence semantics SHOULD be exposed to the AI-NM or management system.

REQ-IDP-09:
An IDP node MUST apply only actions authorized by the applicable policy.

REQ-IDP-10:
An IDP node SHOULD support dampening, hysteresis, or similar mechanisms when decisions can cause traffic oscillation.

14.3. Communication and Collaboration Requirements

REQ-IDP-11:
An IDP node MUST be able to describe its IDP capabilities to a management or AI-NM system, including supported parsing, feature types, state limits, decision function types, and resource constraints.

REQ-IDP-12:
An IDP node that supports collaboration with external systems MUST authenticate and authorize those systems.

REQ-IDP-13:
An IDP node that exports metadata to peer nodes MUST scope that metadata to an authorized IDP domain.

REQ-IDP-14:
An IDP node that supports escalation MUST support rate limiting or backpressure for escalated traffic.

REQ-IDP-15:

An IDP node SHOULD support correlation between escalated results and the original traffic context.

14.4. State Management Requirements

REQ-IDP-16:
An IDP node that supports stateful processing MUST support state aging.

REQ-IDP-17:
An IDP node that uses hash-based state indexing MUST provide behavior for collision detection or collision mitigation.

REQ-IDP-18:
State exhaustion MUST NOT cause corruption of unrelated flows or forwarding behavior.

REQ-IDP-19:
An IDP node SHOULD provide counters for state allocation failure, state collision, state eviction, and fallback events.

14.5. Security and Privacy Requirements

REQ-IDP-20:
IDP control and management interfaces MUST provide authentication, authorization, and integrity protection.

REQ-IDP-21:
IDP metadata that can affect forwarding or filtering MUST be integrity protected within its trust domain.

REQ-IDP-22:
An IDP node MUST provide protection against excessive metadata export, escalation, or event generation.

REQ-IDP-23:
An IDP node MUST support policy controls to limit export of sensitive features, results, and decision traces.

REQ-IDP-24:
IDP metadata that is meaningful only inside an IDP domain MUST be removed, hidden, or protected at domain boundaries unless explicitly permitted by policy.

15. Manageability Considerations

An IDP implementation SHOULD provide management functions for capability discovery, policy configuration, feature extraction configuration, state size and timeout configuration, model or decision-function lifecycle, threshold and confidence configuration, metadata export configuration, resource monitoring, logs and alarms, fallback and safe-mode control, version consistency checks, and configuration rollback.

Management systems SHOULD be able to determine whether an IDP task can be installed without exceeding resource limits. An IDP node SHOULD reject or defer activation of a task that cannot be safely installed.

16. OAM Considerations

IDP functions can affect forwarding, classification, steering, telemetry, and security operations. Therefore, OAM support is important.

An IDP node SHOULD support OAM visibility for whether an IDP task is active, whether a packet or flow matched an IDP policy, whether a decision function was executed, whether a result was produced, whether fallback was used, whether metadata was exported, whether escalation occurred, and decision latency and resource pressure.

OAM tools SHOULD avoid exposing sensitive features or results unless authorized by policy.

17. Performance Considerations

IDP functions can consume table entries, memory, registers, counters, action bandwidth, accelerator bandwidth, export bandwidth, and processing stages. Implementations SHOULD provide explicit resource accounting.

Operators SHOULD evaluate maximum supported throughput, additional processing latency, maximum number of concurrent flows or states, table and memory occupancy, accelerator throughput and queueing, metadata overhead, telemetry export rate, behavior under traffic bursts, and behavior under adversarial or malformed traffic.

Accuracy metrics are application-specific and are not standardized by this document. Operators SHOULD evaluate accuracy, false positive rate, false negative rate, and confidence behavior for each deployment.

18. Security Considerations

IDP introduces new security considerations because data-plane signals and decision results can influence forwarding, filtering, steering, telemetry, or mitigation.

18.1. Signal Spoofing and Poisoning

An attacker can attempt to spoof, manipulate, or poison signals used by IDP functions. Examples include crafted traffic patterns, misleading metadata, forged service state, or manipulated telemetry. IDP nodes and controllers MUST validate the source and integrity of signals that can affect forwarding or filtering. Signals from untrusted sources SHOULD be treated as low confidence or ignored.

18.2. Model and Decision-Function Integrity

If a model or decision function is installed on an IDP node, its origin and integrity MUST be verified. Implementations SHOULD support versioning, rollback, and consistency checks. Unauthorized model updates can result in traffic misclassification, policy bypass, or denial of service.

18.3. Resource Exhaustion

IDP functions can be attacked by generating many flows, many state entries, excessive metadata, high escalation rates, or excessive event reports. IDP nodes MUST support resource limits and SHOULD support rate limiting, quotas, and fallback.

18.4. Evasion and Adversarial Traffic

Attackers can craft traffic to evade classification, reduce confidence, escalate to slower processing paths, or exploit pre-analysis windows. Operators SHOULD use conservative actions for low-confidence results and SHOULD monitor abnormal changes in confidence, fallback, and escalation counters.

18.5. Policy Bypass

Data-plane decisions MUST remain constrained by policy. An IDP node MUST NOT allow a decision function to perform an action that is not explicitly authorized. Policy consistency SHOULD be verifiable by the management system.

18.6. Metadata Integrity

IDP metadata can influence downstream nodes. Metadata that affects forwarding, filtering, steering, or security actions MUST be protected against tampering within the IDP domain.

18.7. Safe Mode

An IDP node MUST have a safe mode. Safe mode SHOULD preserve basic forwarding behavior or apply a configured conservative policy when signals, models, accelerators, or collaboration channels fail.

19. Privacy Considerations

IDP systems can process packet metadata, flow behavior, timing patterns, application indicators, service identifiers, tenant context, or user-related traffic patterns. Such information can be sensitive even when payloads are encrypted.

An IDP deployment SHOULD apply data minimization. It SHOULD export only the features and results necessary for the configured purpose. It SHOULD avoid exporting raw packet bytes unless explicitly authorized. It SHOULD limit retention of decision traces and feature records. It SHOULD remove or protect IDP metadata at administrative boundaries.

If IDP results can be used to infer user behavior, application use, tenant identity, or sensitive service information, access to those results MUST be controlled.

20. IANA Considerations

This document makes no request of IANA.

Future documents that define IDP metadata, protocol extensions, TLVs, YANG modules, or registries MUST include their own IANA considerations.

21. References

21.1. Normative References

[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/rfc/rfc2119>.

[RFC7426] Haleplidis, E., Ed., Pentikousis, K., Ed., Denazis, S., Hadi Salim, J., Meyer, D., and O. Koufopavlou, "Software-Defined Networking (SDN): Layers and Architecture Terminology", RFC 7426, DOI 10.17487/RFC7426, January 2015, <https://www.rfc-editor.org/rfc/rfc7426>.

[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/rfc/rfc8174>.

[RFC9197] Brockners, F., Ed., Bhandari, S., Ed., and T. Mizrahi, Ed., "Data Fields for In Situ Operations, Administration, and Maintenance (IOAM)", RFC 9197, DOI 10.17487/RFC9197, May 2022, <https://www.rfc-editor.org/rfc/rfc9197>.

[RFC9232] Song, H., Qin, F., Martinez-Julia, P., Ciavaglia, L., and A. Wang, "Network Telemetry Framework", RFC 9232, DOI 10.17487/RFC9232, May 2022, <https://www.rfc-editor.org/rfc/rfc9232>.

21.2. Informative References

[BoS] Yan, J., Xu, H., Liu, Z., Li, Q., Xu, K., Xu, M., and J. Wu, "Brain-on-Switch: Towards Advanced Intelligent Network Data Plane via NN-Driven Traffic Analysis at Line-Speed", Proceedings of the 21st USENIX Symposium on Networked Systems Design and Implementation , 2024.

[FENIX] Gao, X., Li, T., Zhang, Y., Wang, Z., Zeng, X., Yao, S., and K. Xu, "FENIX: Enabling In-Network DNN Inference with FPGA-Enhanced Programmable Switches", Proceedings of the USENIX Symposium on Networked Systems Design and Implementation , 2026.

[NetBeacon] Zhou, G., Liu, Z., Fu, C., Li, Q., and K. Xu, "An Efficient Design of Intelligent Network Data Plane", Proceedings of the 32nd USENIX Security Symposium , 2023.

[Pegasus] Zhang, Y., Yao, S., Feng, Y., Chen, K., Li, T., Liu, Z., Zhao, Y., Zhang, L., Gao, X., Xiong, F., Li, Q., and K. Xu, "Pegasus: A Universal Framework for Scalable Deep Learning Inference on the Dataplane", Proceedings of ACM SIGCOMM , 2025.

[RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information", STD 77, RFC 7011, DOI 10.17487/RFC7011, September 2013, <https://www.rfc-editor.org/rfc/rfc7011>.

[RFC8300] Quinn, P., Ed., Elzur, U., Ed., and C. Pignataro, Ed., "Network Service Header (NSH)", RFC 8300, DOI 10.17487/RFC8300, January 2018, <https://www.rfc-editor.org/rfc/rfc8300>.

[RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <https://www.rfc-editor.org/rfc/rfc8402>.

[RFC8986] Filsfils, C., Ed., Camarillo, P., Ed., Leddy, J., Voyer, D., Matsushima, S., and Z. Li, "Segment Routing over IPv6 (SRv6) Network Programming", RFC 8986, DOI 10.17487/RFC8986, February 2021, <https://www.rfc-editor.org/rfc/rfc8986>.

[RFC9637] Welzl, M., Gjessing, S., Carrozzo, G., and A. Sathiaseelan, "Use Cases for In-Network Computing", RFC 9637 , 2024.

Acknowledgments

The authors would like to thank the IETF and IRTF community for their valuable feedback and insights during the development of this proposal.

Authors' Addresses

Ke Xu
Tsinghua University & Zhongguancun Laboratory
Beijing
China
Email: xuke@tsinghua.edu.cn

Liang Zhu
Zhongguancun Laboratory
Beijing
China
Email: zhuliang-x@hotmail.com

Xiangyu Gao
Tsinghua University
Beijing
China
Email: gao-xy24@mails.tsinghua.edu.cn

Yinchao Zhang
Tsinghua University
Beijing
China
Email: afireswallow@gmail.com