Operations and Management Area
Internet-Draft
Intended status: Standards Track
Expires: 29 November 2026

L. Melegassi
Catellix

28 May 2026

Exporting MVPS Coherence Events over Standard Telemetry Channels (syslog, IPFIX, YANG-Push)

draft-melegassi-opsawg-mvps-telemetry-export-00

Abstract

Multi-Vantage Path Snapshot (MVPS) deployments raise coherence events -- ALARM, BYZANTINE EVENT, and phase transitions such as MPLS_CAMOUFLAGE_SUSPECTED -- that operators must ingest into existing monitoring systems (Network Management Systems, SIEMs, time-series collectors). This document specifies a single canonical MVPS event object and three NORMATIVE, lossless mappings of that object onto standard, vendor-neutral telemetry channels: structured syslog (RFC 5424), IP Flow Information Export (IPFIX, RFC 7011), and YANG-Push subscribed notifications (RFC 8639, RFC 8641) carried over NETCONF or RESTCONF. A consumer MAY ingest MVPS events from any one channel without loss of the fields required to reproduce the alarm decision, and the same stable event identifier is carried on every channel so that deduplication across channels is consistent.

This document specifies CHANNELS, not products. No monitoring product is named normatively; a non-normative companion guide demonstrates ingestion into one open-source system. Every mapping property is validated by scripts/validate_telemetry_export.py (8/8 PASS, exit 0) and recorded in evidence/telemetry_export_receipt.json.

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 29 November 2026.

Copyright Notice

Copyright (c) 2026 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

An MVPS broker (see the MVPS bundle format [I-D.melegassi-ippm-mvps-bundle] and the Coherence-BFD fast path [I-D.melegassi-coherence-bfd]) emits operational events: it raises an ALARM when the multi-vantage Mahalanobis distance D^2 crosses threshold for M consecutive ticks, a BYZANTINE EVENT when max and minimax aggregation diverge, and phase transitions such as MPLS_CAMOUFLAGE_SUSPECTED [I-D.melegassi-ippm-mvps-mpls].

These events are useless if they cannot reach the operator's existing tooling. Operators do not deploy new dashboards per framework; they ingest into the Network Management System (NMS), SIEM, or time-series collector they already run. Those systems consume a small set of standard, vendor-neutral channels.

This document defines ONE canonical MVPS event object (Section 3) and three NORMATIVE mappings onto those standard channels:

A. Structured syslog, RFC 5424 [RFC5424], using a registered STRUCTURED-DATA element. Ubiquitous; every NMS/SIEM parses it.
B. IPFIX, RFC 7011 [RFC7011], using enterprise-specific Information Elements. For collectors already on a flow pipeline.
C. YANG-Push subscribed notifications, RFC 8641 [RFC8641] over the dynamic-subscription mechanism of RFC 8639 [RFC8639], carried by NETCONF [RFC6241] or RESTCONF [RFC8040], using the YANG module in Appendix A. For model-driven telemetry stacks.

The design goal is LOSSLESS, CONSISTENT export: a consumer reading any single channel obtains every field needed to reproduce the alarm decision, and the same stable event identifier (Section 3.3) appears

on all channels so cross-channel deduplication is well defined.

1.1. Relationship to other MVPS documents

This document is the EXPORT surface of the MVPS family. It does not define new detection mathematics; it transports the events already defined elsewhere:

- The event payload fields (D^2, phase, vantage count, Byzantine fraction) are produced by the core detector [I-D.melegassi-ippm-mvps-bundle] and its profiles.

- The MVPS Operational Log Format [I-D.melegassi-opsawg-mvps-logging] is the INTERNAL, append-only, tamper-evident record. THIS document is the EXTERNAL push to third-party monitoring. An exported event SHOULD carry the log seq and prev/record hash of its log entry so an investigator can pivot from the NMS back into the anchored log.

- When integrity of the exported stream matters, the anchor head of the Coherent-Witness Trust checkpoint [I-D.melegassi-santos-ippm-mvps-cwt] and/or the Proof Envelope [I-D.melegassi-ippm-mvps-proof-envelope] MAY be carried as an opaque field; verification of that anchor is out of scope here.

1.2. Scope and Non-Goals

In scope:

- A canonical, hash-stable MVPS event object.
- Three lossless field mappings (syslog, IPFIX, YANG-Push).
- Cross-channel consistency rules (identifier, severity ordering).

Out of scope (NON-GOALS):

- Naming or endorsing any monitoring product. The mappings target open standards; any conformant collector can consume them. A separate, NON-NORMATIVE companion guide shows one open-source ingestion as an illustration only.
- Transport security and authentication of the channels themselves; these are provided by the underlying transports (syslog over TLS [RFC5425]; NETCONF/RESTCONF over SSH/TLS). See Section 9.
- Re-specifying detection, trust, or log-integrity mathematics; those live in the cited drafts and are not weakened or restated here.
- A bidirectional control channel. This document is EXPORT only (telemetry out); it defines no command or actuation path.

2. Terminology

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.

MVPS event object: the channel-independent, canonical representation of a single MVPS operational event (Section 3).

Channel: one of the three standard telemetry encodings (syslog, IPFIX, YANG-Push) onto which the event object is mapped.

Consumer: any NMS, SIEM, or collector that ingests one or more channels.

Canonical form: the JSON Canonicalization Scheme (JCS) [RFC8785] serialization of the event object, used for the stable identifier and for the test vectors in Appendix B.

3. The Canonical MVPS Event Object

An MVPS event is a flat object of typed fields. The canonical form is its JCS [RFC8785] serialization. Every channel mapping in Sections 4-6 is a total function on this object: each REQUIRED field maps to exactly one channel element, and no channel element is introduced that does not correspond to a field defined here. This one-to-one property is the basis of the lossless guarantee (Section 7) and is checked by T-TEL-LOSSLESS-1.

3.1. Fields

Field	Type	Meaning	Req'd?
event_id	string	Stable identifier (Section 3.3)	MUST
event_type	enum	alarm | byzantine | phase | vantage | anchor	MUST
severity	enum	Section 3.2	MUST
timestamp	string	RFC 3339 UTC instant	MUST
bundle_seq	uint64	Coordination-window seq	MUST
phi_d	decimal	Phase distance Phi_D	SHOULD
d2	decimal	Mahalanobis D^2	SHOULD
vantage_count	uint16	N admitted vantages	SHOULD
byzantine_frac	decimal	Estimated f (0..1)	MAY
phase	enum	Operational phase label	SHOULD
path_fingerprint	string	64 hex (bundle FP)	MAY
log_seq	uint64	Log entry seq (logging)	MAY
log_record_hash	string	64 hex (logging)	MAY
anchor_head	string	Opaque CWT/Envelope head	MAY

Table 1: MVPS event fields

A producer MUST emit every field marked MUST; it SHOULD emit SHOULD fields whenever the value exists; it MAY omit MAY fields. A consumer MUST tolerate the absence of any non-MUST field.

The "phase" enumeration includes at least: NOMINAL, DEGRADED, ALARM, BYZANTINE, MPLS_CAMOUFLAGE_SUSPECTED. Additional phase labels MAY be defined by profiles; consumers MUST treat an unrecognized phase as opaque and not as an error.

3.2. Severity

MVPS severity is an enumeration aligned ONE-TO-ONE with the syslog SEVERITY numeric scale of RFC 5424 (Table 2 of [RFC5424]), so that

ordering is preserved across all channels by construction:

MVPS severity	syslog code	typical use
emergency	0	(reserved; not emitted)
alert	1	fleet-wide BYZANTINE
critical	2	BYZANTINE EVENT
error	3	admission / auth failure
warning	4	ALARM raised
notice	5	phase transition
info	6	vantage join/leave
debug	7	diagnostic

The map severity -> syslog code is a strictly decreasing bijection on the eight labels; lower numeric code means higher urgency. Every channel mapping MUST preserve this ordering (T-TEL-SEV-1). Note that the "audit" marker of the MVPS log format is NOT a severity here; an audit event carries its own severity plus an audit=true marker, so ordering remains total.

3.3. The Stable Event Identifier

event_id is a stable, channel-independent identifier computed as the lowercase hex SHA-256 of the JCS serialization of the event object with the "event_id" key itself removed:

```
event_id = SHA-256( JCS( event \ {event_id} ) )   [64 hex chars]
```

Two producers given the same event fields MUST compute the same event_id (determinism), and the identifier MUST be carried unchanged on every channel. Consumers use it as the deduplication key, so an event delivered redundantly on two channels is recognized as one event (T-TEL-IDEMPOTENT-1). Because event_id binds every field, a field change yields a different identifier (T-TEL-CANON-1).

4. Mapping A: Structured syslog (RFC 5424)

Each MVPS event is emitted as one RFC 5424 SYSLOG message.

- PRI: PRIVAL = FACILITY * 8 + SEVERITY. FACILITY defaults to 16 (local0) and is configurable; SEVERITY is the code from Section 3.2.

- HEADER: VERSION = 1; TIMESTAMP = the event timestamp in RFC 3339 with mandatory fractional seconds; HOSTNAME = broker host; APP-NAME = "mvps"; MSGID = event_type.

- STRUCTURED-DATA: a single SD-ELEMENT with SD-ID "mvps@32473", where 32473 is the example Private Enterprise Number reserved for documentation [RFC5612]; a deployment MUST substitute its own IANA-assigned PEN. Each event field of Table 1 becomes one SD-PARAM whose PARAM-NAME is the field name and whose PARAM-VALUE is the field value rendered as a UTF-8 string. The three characters '"', '\', and ']' inside a PARAM-VALUE MUST be escaped with a backslash per RFC 5424 Section 6.3.3.

- MSG: a short human-readable summary; it is informational and MUST NOT be required to recover any field (all fields live in the SD-ELEMENT).

Example (folded for layout; one physical line on the wire):

```
<132>1 2026-05-28T18:00:00.500Z broker01 mvps alarm.raise
[mvps@32473 event_id="b3d1..." event_type="alarm"
 severity="warning" bundle_seq="41" d2="38.7" phi_d="0.91"
```

```
    vantage_count="12" phase="ALARM"] ALARM D2=38.7 over M ticks
```

PRIVAL 132 = 16*8 + 4 (local0, warning). A consumer recovers the event object by reading the SD-PARAMs; the round trip object -> syslog -> object is lossless on all present fields (T-TEL-SYSLOG-1) and the framing is well-formed (T-TEL-SYSLOG-2).

5. Mapping B: IPFIX (RFC 7011)

For collectors on a flow pipeline, each MVPS event is one IPFIX Data Record described by an enterprise-specific Template (RFC 7011 Section 3.2, enterprise bit set, Private Enterprise Number as in Section 4). Information Elements (abstract data types per RFC 7011 Section 6):

Field	IE name (enterprise)	abstractDataType
event_id	mvpsEventId	octetArray[32]
event_type	mvpsEventType	unsigned8 (enum)
severity	mvpsSeverity	unsigned8
timestamp	mvpsEventTime	dateTimeMilliseconds
bundle_seq	mvpsBundleSeq	unsigned64
d2	mvpsD2Milli	unsigned32 (D^2*1000)
phi_d	mvpsPhiDMilli	unsigned32 (Phi_D*1000)
vantage_count	mvpsVantageCount	unsigned16
byzantine_frac	mvpsByzFracMilli	unsigned16 (f*1000)
phase	mvpsPhase	unsigned8 (enum)
path_fingerprint	mvpsPathFingerprint	octetArray[32]

Decimal fields are carried as fixed-point integers scaled by 1000 ("milli" suffix) to avoid floating-point on the wire; a consumer divides by 1000 to recover the value. The scale is exact for the three decimal places that MVPS reports; encode/decode round-trips within type width (T-TEL-IPFIX-1). event_id and path_fingerprint are carried as the raw 32-octet digests, not hex.

Optional fields absent from a given event are omitted from the Template for that record set; a collector keys records by the Template ID it received.

6. Mapping C: YANG-Push (RFC 8639/8641)

For model-driven stacks, MVPS events are delivered as YANG notifications via a dynamic subscription (RFC 8639) using the periodic/on-change push mechanics of RFC 8641, transported over NETCONF [RFC6241] or RESTCONF [RFC8040]. The notification is the "mvps-coherence-event" of the YANG module in Appendix A.

Each event field of Table 1 maps to one leaf:

```
event_id          -> leaf event-id        (string, 64 hex)
event_type        -> leaf event-type      (enumeration)
severity          -> leaf severity        (enumeration)
timestamp         -> leaf event-time       (yang:date-and-time)
bundle_seq        -> leaf bundle-seq      (uint64)
d2                -> leaf d2               (decimal64 fd=3)
phi_d             -> leaf phi-d            (decimal64 fd=3)
vantage_count     -> leaf vantage-count   (uint16)
byzantine_frac    -> leaf byzantine-frac  (decimal64 fd=3)
phase             -> leaf phase            (enumeration)
path_fingerprint  -> leaf path-fingerprint (string, 64 hex)
log_seq           -> leaf log-seq          (uint64)
log_record_hash   -> leaf log-record-hash  (string, 64 hex)
anchor_head       -> leaf anchor-head      (string)
```

The leaf types match the field types of Table 1; a notification instance whose leaves carry the field values validates against the module (T-TEL-YANG-1). REQUIRED fields are "mandatory true"; all others are optional leaves.

7. Cross-Channel Consistency Requirements

A conformant producer MUST satisfy, for every event it emits:

C1 Identity. The event_id (Section 3.3) is byte-identical on every channel the event is emitted on (T-TEL-IDEMPOTENT-1).

C2 Severity ordering. The severity-to-channel encoding preserves the total order of Section 3.2 on every channel (T-TEL-SEV-1).

C3 Lossless coverage. The union of fields recoverable from any one channel includes every REQUIRED field and every SHOULD field the producer populated; no channel may silently drop a populated field (T-TEL-LOSSLESS-1).

C4 Canonical determinism. Given identical event fields, two independent producers compute the same canonical form and hence the same event_id (T-TEL-CANON-1).

A consumer that ingests more than one channel MUST deduplicate by event_id and MUST NOT treat the same event arriving on two channels as two distinct events.

8. Numerical Receipt

All properties of Sections 3-7 are validated CONSTRUCTIVELY by scripts/validate_telemetry_export.py, which builds a synthetic event, maps it onto all three channels, round-trips each mapping, and checks C1-C4. It writes evidence/telemetry_export_receipt.json with the per-check results, the demo event_id, and the platform fingerprint. The validator returns exit 0 iff all eight checks (T-TEL-CANON-1, T-TEL-SYSLOG-1, T-TEL-SYSLOG-2, T-TEL-IPFIX-1, T-TEL-YANG-1, T-TEL-SEV-1, T-TEL-IDEMPOTENT-1, T-TEL-LOSSLESS-1) pass.

9. Security Considerations

This document defines an EXPORT encoding; it does not itself provide confidentiality, integrity, or authentication of the channel. Those are the responsibility of the underlying transport:

- syslog MUST be carried over the TLS transport [RFC5425] when it leaves a trusted segment; cleartext UDP syslog [RFC5426] is acceptable only on an isolated management network.
- NETCONF/RESTCONF carry their own mandatory transport security (SSH/TLS); subscription state is protected by that channel.
- IPFIX SHOULD use the transport protections of RFC 7011 Section 11.

The exported event reveals operational posture (alarm state, Byzantine fraction, vantage count) that an adversary could use for timing or reconnaissance. Producers SHOULD restrict export to authenticated collectors and SHOULD NOT include optional fields (path_fingerprint, anchor_head) on channels crossing untrusted segments unless the transport is encrypted.

Because event_id binds every field by SHA-256, a consumer can detect in-flight modification of a single channel's payload by recomputing the identifier; this is an integrity CHECK, not a replacement for transport authentication, and it does not by itself prove freshness (replay protection is the transport's job).

This export path is strictly read-only: it carries no command, configuration, or actuation, so it adds no remote-control attack surface to an MVPS deployment.

10. IANA Considerations

This document requests, upon adoption:

- Registration of the YANG module "mvps-telemetry" (Appendix A) in the "YANG Module Names" registry, with an assigned namespace URI of the form urn:ietf:params:xml:ns:yang:mvps-telemetry.

- Allocation of the enterprise-specific IPFIX Information Elements of Section 5 under the registrant's Private Enterprise Number; these are enterprise elements and require no IANA IE allocation, but are listed here for interoperability.

The SD-ID "mvps@32473" uses the documentation PEN 32473 [RFC5612]; deployments substitute their own PEN and no IANA action is required for the syslog SD-ID.

This version (-00) defines no other IANA actions.

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, May 2017.

[RFC5424] Gerhards, R., "The Syslog Protocol", RFC 5424, March 2009.

[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, September 2013.

[RFC8639] Voit, E., Clemm, A., Gonzalez Prieto, A., Nilsen-Nygaard, E., and A. Tripathy, "Subscription to YANG Notifications", RFC 8639, September 2019.

[RFC8641] Clemm, A. and E. Voit, "Subscription to YANG Notifications for Datastore Updates", RFC 8641, September 2019.

[RFC8785] Rundgren, A., Jordan, B., and S. Erdtman, "JSON Canonicalization Scheme (JCS)", RFC 8785, June 2020.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, January 2017.

11.2. Informative References

[RFC5425] Miao, F., Ed., Ma, Y., Ed., and J. Salowey, Ed., "Transport Layer Security (TLS) Transport Mapping for Syslog", RFC 5425, March 2009.

[RFC5426] Okmianski, A., "Transmission of Syslog Messages over UDP", RFC 5426, March 2009.

[RFC5612] Eronen, P. and D. Harrington, "Enterprise Number for Documentation Use", RFC 5612, August 2009.

[I-D.melegassi-ippm-mvps-bundle] Melegassi, L., "The MVPS Bundle Format", Work in Progress, Internet-Draft, draft-melegassi-ippm-mvps-bundle-00.

[I-D.melegassi-coherence-bfd] Melegassi, L., "Coherence-BFD", Work in Progress, Internet-Draft, draft-melegassi-coherence-bfd-00.

[I-D.melegassi-ippm-mvps-mpls] Melegassi, L., "MVPS Path Coherence under MPLS Camouflage", Work in Progress, Internet-Draft, draft-melegassi-ippm-mvps-mpls-00.

[I-D.melegassi-opsawg-mvps-logging] Melegassi, L., "The MVPS Operational Log Format", Work in Progress, Internet-Draft, draft-melegassi-opsawg-mvps-logging-00.

[I-D.melegassi-santos-ippm-mvps-cwt] Melegassi, L. and Santos, "Coherent-Witness Trust", Work in Progress, Internet-Draft, draft-melegassi-santos-ippm-mvps-cwt-00.

[I-D.melegassi-ippm-mvps-proof-envelope] Melegassi, L., "The MVPS Proof Envelope", Work in Progress, Internet-Draft, draft-melegassi-ippm-mvps-proof-envelope-00.

Appendix A. YANG Module

This module defines the notification used by Mapping C. It is non-normative pending IANA namespace assignment and is provided so that the YANG-Push mapping is fully specified and machine-checkable.

<CODE BEGINS> file "mvps-telemetry@2026-05-28.yang"

```
module mvps-telemetry {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:mvps-telemetry";
  prefix mvpst;

  import ietf-yang-types { prefix yang; }

  organization "Catellix Research";
  contact "Leonardo Melegassi <melegassi@catellix.com>";
  description
    "Export notification for MVPS coherence events.  Read-only
     telemetry; carries no configuration or actuation.";
  revision 2026-05-28 {
    description "Initial -00 revision.";
    reference "draft-melegassi-opsawg-mvps-telemetry-export-00";
  }
```

```
  typedef hex64 {
    type string {
      pattern '[0-9a-f]{64}';
    }
    description "Lowercase hex SHA-256 digest (64 chars).";
  }

  notification mvps-coherence-event {
    description "One MVPS operational event.";
    leaf event-id        { type hex64;        mandatory true; }
    leaf event-type {
      type enumeration {
        enum alarm; enum byzantine; enum phase;
        enum vantage; enum anchor;
      }
      mandatory true;
    }
    leaf severity {
      type enumeration {
        enum emergency; enum alert; enum critical; enum error;
        enum warning;   enum notice; enum info;    enum debug;
      }
      mandatory true;
    }
    leaf event-time     { type yang:date-and-time; mandatory true; }
    leaf bundle-seq     { type uint64;          mandatory true; }
    leaf d2             { type decimal64 { fraction-digits 3; } }
    leaf phi-d          { type decimal64 { fraction-digits 3; } }
    leaf vantage-count  { type uint16; }
    leaf byzantine-frac { type decimal64 { fraction-digits 3; } }
    leaf phase {
      type enumeration {
        enum NOMINAL; enum DEGRADED; enum ALARM; enum BYZANTINE;
        enum MPLS_CAMOUFLAGE_SUSPECTED;
      }
    }
    leaf path-fingerprint { type hex64; }
    leaf log-seq          { type uint64; }
    leaf log-record-hash  { type hex64; }
    leaf anchor-head      { type string; }
    leaf audit            { type boolean; default false; }
  }
}
<CODE ENDS>
```

Appendix B. Worked Example (all three channels)

Canonical event object (JCS-serialized, pretty-printed here):

```
{
  "anchor_head": "68d3f86d...",
  "bundle_seq": 41,
  "d2": 38.7,
  "event_type": "alarm",
  "phase": "ALARM",
  "phi_d": 0.91,
  "severity": "warning",
  "timestamp": "2026-05-28T18:00:00.500Z",
  "vantage_count": 12
}
```

event_id = SHA-256(JCS(object without event_id)). The exact value for this example is printed by the validator and pinned in evidence/telemetry_export_receipt.json (field "demo_event_id").

syslog (Mapping A): the message of Section 4.

IPFIX (Mapping B): one Data Record with mvpsD2Milli=38700, mvpsPhiDMilli=910, mvpsBundleSeq=41, mvpsVantageCount=12, mvpsSeverity=4, mvpsPhase=ALARM(enum), mvpsEventTime=the instant.

YANG-Push (Mapping C): an <mvps-coherence-event> notification whose leaves carry the same values, d2=38.700, phi-d=0.910.

All three recover the identical event_id, demonstrating C1.

Author’s Address

Leonardo Melegassi
Catellix Research
Email: melegassi@catellix.com