Network Working Group
Internet-Draft
Intended status: Informational
Expires: 28 June 2026

M. Panke
Independent Researcher

December 2025

Event and Webhook Delivery Semantics
draft-mayankpanke-event-delivery-semantics-00

Abstract

Event- and webhook-based integrations are a common application-layer mechanism for interoperability across Internet services, yet they lack a shared delivery semantics contract. Existing implementations vary widely in retry behavior, acknowledgment signaling, failure classification, idempotency identifiers, and replay handling, resulting in fragile integrations and ambiguous operational expectations.

This document defines a minimal, application-layer delivery semantics profile for event and webhook delivery over existing transports, most commonly HTTP.

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 4 June 2026.

Copyright Notice

Table of Contents

1. Introduction

Event- and webhook-based delivery is widely used to notify external systems of state changes or occurrences across administrative boundaries. These integrations are typically implemented over HTTP and are expected to operate in the presence of partial failure, independent failure domains, and variable network conditions.

Despite their ubiquity, there is no common, interoperable delivery semantics model that defines how delivery attempts, retries, acknowledgments, or failures are signaled and interpreted. This document addresses that gap by defining a minimal, application-layer semantics profile focused on sender-observable behavior.

2. Problem Statement

Event- and webhook-based delivery is a widely deployed application-layer integration pattern used to notify external systems of state changes or occurrences. Such delivery is commonly implemented over existing transports, most frequently HTTP, and is typically expected to operate in the presence of partial failure, network interruption, and independent administrative control of endpoints.

Despite widespread deployment, there is no common, interoperable delivery semantics contract governing how event producers, consumers, and intermediaries signal delivery attempts, retries, acknowledgment, or failure. As a result, independently developed implementations rely on implicit assumptions, vendor-specific conventions, or informal documentation, leading to fragile integrations and inconsistent operational behavior.

2.1. Interoperability Failures Observed in Practice

The absence of a shared delivery semantics model results in recurring issues including ambiguous retry behavior and retry exhaustion signaling, inconsistent acknowledgment semantics, non-standard idempotency and deduplication signaling, unclear classification of delivery failures, and increased exposure to replay and abuse attacks.

These issues arise from missing application-layer semantics rather than deficiencies in underlying transport protocols.

2.2. Limitations of Existing Transport Semantics

Transport protocols such as HTTP provide generic delivery and error signaling mechanisms but intentionally do not define application-specific semantics such as retry interpretation, acknowledgment meaning, or deduplication scope. Correct use of transport semantics alone is therefore insufficient to ensure interoperable event delivery behavior.

2.3. Non-Goals

This document does not attempt to define or guarantee end-to-end delivery or processing correctness; provide exactly-once, at-most-once, or effectively-once guarantees; impose receiver-side storage or execution requirements; redefine or replace existing transport protocols; or standardize event payload formats or business semantics. All guarantees are limited to sender-observable behavior and signaling.

3. Goals

The goals of this specification are to define a common vocabulary for event delivery semantics, standardize observable signaling for delivery attempts and retries, enable interoperable idempotency and deduplication signaling, improve security posture through consistent guidance, and reduce ambiguity without constraining implementation choices.

4. Terminology

The key words "MUST", "MUST NOT", "SHOULD", and "MAY" are to be interpreted as described in RFC 2119.

Delivery Attempt A single effort by a producer or intermediary to deliver an event to a consumer endpoint.

Acknowledgment A signal indicating receipt of a delivery attempt, without implying processing success or durability.

Idempotency Identifier A producer-supplied identifier intended to correlate multiple delivery attempts for the same logical event.

5. Architecture Overview

This specification considers three primary roles: Event Producer, Event Consumer, and an optional Intermediary. The delivery semantics defined in this document apply to interactions between these roles at the application layer.

6. Delivery Semantics Model

This specification defines a sender-observable delivery semantics model for event and webhook delivery. The model applies to each delivery attempt independently and does not infer receiver-side processing behavior beyond what is explicitly signaled.

6.1. Delivery Attempt

A Delivery Attempt represents a single, discrete effort by an Event Producer or Intermediary to deliver an event to a Consumer endpoint. Each Delivery Attempt MUST be treated as best-effort, MUST NOT imply receipt, durability, or processing, and MAY be retried according to sender policy when sender-observed outcomes indicate a non-terminal condition.

6.2. Sender-Observed Outcomes

This specification defines sender-observed outcomes for a Delivery Attempt, including Accepted, Transient Failure, and Terminal Failure. Senders MUST base retry behavior solely on these outcomes and MUST NOT infer receiver-side state beyond what is explicitly signaled.

6.3. Retry Signaling

Retry behavior is sender-driven and policy-dependent. This specification standardizes only retry signaling, not retry guarantees. Consumers MAY provide retry guidance, and senders MAY honor such guidance.

7. Idempotency and Deduplication Signaling

Idempotency and deduplication are addressed through explicit signaling mechanisms only. This specification does not require or assume receiver-side deduplication behavior.

7.1. Idempotency Identifier

An Idempotency Identifier is a producer-supplied identifier intended to correlate multiple Delivery Attempts representing the same logical event. When present, the identifier MUST remain stable across retries and MUST be scoped to the producer-consumer context.

7.2. Deduplication Expectations

Consumers MAY choose to perform deduplication based on the Idempotency Identifier and local policy. Senders MUST tolerate duplicate delivery in all cases. Deduplication remains an optimization, not a correctness requirement.

8. Failure Classification

Failure classification in this specification is minimal and sender-centric. Failures are classified as transport-level failures or application-level rejections and are used solely to inform sender behavior such as retry decisions, alternate delivery paths, or operator intervention.

9. Security Considerations

Event delivery mechanisms are susceptible to spoofing, replay, and abuse. This document profiles existing security mechanisms and provides guidance on authentication, authorization, integrity protection, and replay mitigation. No new cryptographic mechanisms are defined.

10. IANA Considerations

This document makes no request of IANA.

11. Normative References

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

Author’s Address

Mayank Panke
Independent Researcher
Email: panke.mayank@gmail.com