Network Working Group
Internet-Draft
Intended status: Experimental
Expires: 11 November 2026

S. Koga

10 May 2026

Wire-format Alerting for Risk Notification
draft-koga-warn-00

Abstract

This document describes WARN, a transport-agnostic compact binary wire protocol used for emergency alerts under constrained hardware and lossy networks. It is designed to reach farther and quicker than OASIS CAP, which tries to deliver full human-readable information at the cost of complexity. It primarily uses fixed fields, with support for TLVs when more expressiveness is required. It also has a mandatory signature to prevent fake alerts propagating through a mesh.

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 11 November 2026.

Copyright Notice

Table of Contents

1. Introduction

The current emergency alert infrastructure, [CAP] is flexible and widely deployed, but the structure inherently bears complexity which may fail under extreme conditions such as lossy networks and overloaded devices (i.e. potential outcomes of emergency situations).

This document defines a protocol aimed to be maximally resilient, distributed, and lightweight to mitigate those points.

2. Motivation

2.1. Issues of Existing Solutions

* Lack of efficient internationalization
  - The size of messages multiply with internationalization.
* Unbounded
  - On systems that lack proper amounts of available memory, CAP may overwhelm the chip, given that its length is not hard-capped.
* Centralized
  - Even though the networking around the alert origin may be robust, bottlenecks may exist at any point in between the central origin and client.

  - Physical interruptions of the network may cause connections to fault, and may require an expensive route change.

  - The amount of connections the origin can juggle simultaneously limits the amount of clients that can receive CAP XML alerts.

* Complex

  - XML is hierarchical, and you need to parse the whole tree to fetch information safely.

  - CAP is too flexible and tolerant of duplicate fields in arbitrary order.

  - Pushing multimedia through emergency pipelines increases the risk of corruption and/or incomplete messages.

* Insecure

  - No legitimate enforcement on signing.

* Demanding

  - Memory requirement is theoretically unbound due to CAP being unbound.

  - Media interpretation requires mature, complex libraries.

  - Realistically requires a whole OS to be running to make use of CAP.

2.2. Design Goals

* Minimal alert-plane packet

  - Single UDP datagram

  - Signed, verifiable, immutable

* Best-effort propagation

  - Time-bounded spread

  - Geographic advisory bounds

* Two-plane architecture

  - Alert-plane: eager, proactive, stateless

- Info-plane: lazy, optional (not specified here)

* Compatibility
  - Be easily convertible from existing formats (e.g., CAP)
  - Allow future extensions without breaking existing implementations
* Strong authenticity
  - Alerts are valid only if signed by an Alert Origin key
  - Relay "trust" does not imply alert validity
* no_std / zero-copy friendly
  - Fixed layouts
  - Explicit bounds
  - No required allocation
* MTU safety
  - Recommended UDP payload <= 1200 bytes
* Lightweight
  - Be able to run on minimal hardware

3. Definitions and Conventions

3.1. Normative 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.

3.2. Versioning

WARN uses a two-level versioning scheme to distinguish wire incompatibility from backward-compatible extensions.

* Major version (version_major)

- A change in major version indicates wire-incompatible changes.
- If version_major is greater than the highest version supported by the implementation, the packet MUST NOT be fully interpreted.
- Implementations MAY parse only the common prefix for the purposes of safe rejection, logging, or routing, but MUST treat the packet as unsupported.
- No compatibility guarantees are provided across major versions.
- If version_major is zero, the packet is considered invalid.

* Minor version (version_minor)
  - A change in minor version indicates a backward-compatible extension within the same major version.
  - Minor version updates MUST NOT change the meaning of any field defined in the current major version.
  - Minor version updates MAY:
    - define new flag bits (previously RESERVED),
    - define new TLV types,
    - define new hazard_minor values,
    - add new semantics that can be safely ignored by older implementations.
  - However, keeping up to date is strongly recommended.
  - Receivers MUST ignore unknown flag bits and unknown TLV types.

3.3. Time

* UNIX seconds (UTC), u64.
* Method of syncing time is dependent on hardware and medium.

3.4. Geographic Encoding

Latitude and longitude are signed i32 in 100-nanodegree units (1e-7 degrees).

Ranges:

* latitude: -900_000_000 ... +900_000_000

* longitude: -1_800_000_000 ... +1_800_000_000

3.5. Distance

Distance is encoded as 10-meter units:

* Stored value: radius_10m (u16)

* Real meters: affected_radius_m = radius_10m x 10

* Used for propagation decisions (see Section 10.2)

* A value of 0 indicates "unknown" or "see polygon TLV"

3.6. Authority Model

* ALERT validity requires:

  1. A valid Ed25519 signature, AND

  2. The signing key being present in the local Origin Registry (registry format defined in Section 18).

* On-wire packets DO NOT embed public keys in v1.0.

* The packet identifies the signing key by origin_key_id.

4. Common Prefix (All Packets)

Total size: 8 bytes

Offset	Size	Field	Type	Description
0x00	4	magic	u8[4]	ASCII "WARN"
0x04	1	version_major	u8	v1.0 -> 1
0x05	1	version_minor	u8	v1.0 -> 0
0x06	2	flags	u16	See Section 5
0x08	...	payload	-	ALERT or non-ALERT fields

Table 1

Notes:

* The signed region is [0x00, packet_len - 64) for signed packets.

* The signature is always the last 64 bytes of signed packets.

5. Flags

Bit numbering: Bit 0 is the most significant bit (MSB).

Bit	Name	Meaning
0	ALERT	This is an ALERT packet
1	URGENT	Forward with priority
2	UPDATE	Revision of existing event
3	CANCEL	Cancels an existing event
4	TEST	Test alert
5-15	RESERVED	Unused in v1; MUST be ignored by receivers

Table 2

6. ALERT Packet

6.1. Fixed ALERT Fields

Offset	Size	Field	Type
0x08	8	timestamp_s	u64
0x10	4	event_id	u32
0x14	2	seq	u16
0x16	2	ttl_s	u16
0x18	1	hazard_major	u8
0x19	1	hazard_minor	u8
0x1A	1	urgency	u8
0x1B	1	severity	u8
0x1C	1	certainty	u8
0x1D	1	response	u8
0x1E	8	onset_s	u64
0x26	8	expiry_s	u64
0x2E	8	effective_time_s	u64
0x36	4	epicenter_lat	i32
0x3A	4	epicenter_lon	i32
0x3E	2	radius_10m	u16
0x40	N	signed_tlv	bytes

Table 3

Field descriptions:

* timestamp_s: The time this alert was issued

* event_id: The root ID that this event has. Subsequent updates or queries to a database will utilize this specific key.

* ttl_s: The baseline amount of time relays SHOULD propagate for.

* hazard_major, hazard_minor, urgency, certainty, response: Specified in Section 7.

* onset_s: When the alert becomes active

* expiry_s: When the alert expires

* effective_time_s: When the event actually occurred or will occur

* epicenter_lat, epicenter_lon: Epicenter coordinates in 100-nanodegree units (see Section 3.4).

* radius_10m: Affected radius used for propagation decisions (see Section 10.2)

Deriving signed_tlv bounds: The signed TLV block has no explicit length field. Bounds are derived from the transport-provided packet length:

* signed_tlv starts at offset 0x40

* signed_tlv ends at packet_len - 68 (68 = 4 origin_key_id + 64 signature)

* Receivers MUST validate: packet_len >= 0x40 + 68 (i.e., packet_len >= 132)

6.2. Signature Block

Immediately follows signed_tlv:

Field	Size	Description
origin_key_id	4	Identifies the signing key in the Origin Registry (Section 18)
signature	64	Ed25519 signature

Table 4

* Algorithm: Ed25519 [RFC8032] (required)

* Signed region: [0x00, packet_len - 64)(covers Common Prefix + ALERT fields + signed_tlv + origin_key_id).

* Receivers MUST resolve origin_key_id via the Origin Registry and reject if not present.

* Receivers MUST verify the signature before acting on any ALERT field.

7. Value Tables

These list the possible values for fields in WARN ALERT packets. Most fields are designed to reflect CAP. For advanced meanings of these values, refer to the OASIS [CAP] specs Section 3.2.2. Refer to Section 9 for the tables.

NOTE: additional hazard_minor values are to be determined. Should be able to convert from all preexisting CAP messages which have been produced using this table.

8. Event Identity and Updates

* event_id identifies a physical event.

* seq is monotonic per event, starting from 0.

* seq MUST NOT overflow. In the case seq reaches 65535, the origin MUST re-issue an alert with a REPLACES TLV, and that alert SHOULD be URGENT.

Receiver rules:

* seq < highest_seen -> drop

* seq == highest_seen -> drop duplicate

* seq > highest_seen -> accept update, advance highest_seen

Deduplication state per event:

(origin_key_id, event_id) -> highest_seq: u16

8.1. CANCEL Semantics

A packet with the CANCEL flag set cancels the event identified by event_id.

* CANCEL packets MUST carry a seq strictly greater than the highest previously accepted seq for that event.

* Upon accepting a CANCEL, receivers MUST immediately expire the event and cease acting on it.

* The CANCEL's deduplication entry MUST persist in the replay cache for at least ttl_s seconds measured from the CANCEL packet's own timestamp_s. This prevents late-arriving retransmissions of the original alert from slipping through after the CANCEL entry expires.

9. Signed TLV Format

TLV layout:

* type: u8

* len: u8

* val: u8[len]

Rules:

* Unknown TLVs MUST be ignored.

* TLVs MUST NOT be required for core safety behavior.

TLVs:

* 0x00 UNUSED

* 0x01 HAZARD_NAME (UTF-8)

* 0x02 POLYGON ((i32, i32)[])

  - POLYGON MUST contain no less than 3 points and no more than 8 points.

  - POLYGON points MUST be closed, and MUST be ordered in a counterclockwise fashion, abiding to [RFC7946].

  - POLYGON points MUST be the latitude and longitude of the point in 100-nanodegree units (see Section 3.4).

* 0x03 REPLACES (u32[])

- This is for when an alert origin issues an alert which may replace another for a variety of reasons, such as prevention of seq overflow, merging of two alerts, etc. An alert replacing another SHOULD be marked as URGENT.

10. Forwarding Semantics (ALERT)

10.1. Time-Based TTL

The ttl_s field represents how many seconds the packet is permitted to spread.

Conceptual rule:

```
age_s = now_s - timestamp_s
if age_s > ttl_s -> SHOULD NOT forward
```

10.2. Geographic Bounding

* Relays SHOULD drop packets outside radius_10m x 10 meters.
  - If a relay does not know its own location, it MUST propagate.

10.3. Forwarding Strategy

* Stateless fan-out
* Medium-dependent congestion control

10.4. Forwarding Exceptions

Under any of the conditions specified below, alerts MAY be propagated regardless of ttl and geographic bounding.

* When a relay has a record of that specific event ID in its cache and it receives an alert with higher seq.
* When a relay is unsure of its own time, or if it may have a skewed clock for any reason.
* When a transport medium is known to have a slow transport speed.

11. Non-ALERT Packets

11.1. Fixed Headers

Offset	Size	Field	Type
0x08	2	kind	u16
0x0A	...	general payload	-

Table 5

The kind field identifies the packet type. All non-ALERT packets share this header immediately after the Common Prefix.

Receivers MUST silently drop any packet whose kind is unknown or whose kind belongs to a different transport’s range.

11.2. WARN Reserved Ranges

The table of reservations for WARN 1.0 is as follows.

Range/Value	Category
0x0000	RESERVED(invalid)
0x0001-0x00FF	WARN
0x0100-0x01FF	IPWARN
0x0200-0xFEFF	Future use
0xFF00-0xFFFE	Private use
0xFFFF	RESERVED(invalid)

Table 6

12. Non-ALERT Reserved Packet Kinds

All ADVISORY packets MUST be signed by the master origin key. Signing is determined by kind, not by a flag.

Receivers MUST:

* Verify the Ed25519 signature over the signed region [0x00, packet_len - 64) if they exist.

* Drop the packet if verification fails.

* Drop the packet if kind is unknown.

* Drop the packet if new_registry_version is less than or equal to its own registry’s version.

12.1. Advisory Signature Block

Immediately follows the kind-specific payload:

Field	Size	Description
signature	64	Ed25519 signature over [0x00, packet_len - 64)

Table 7

12.2. Advisory Kind Assignments

Kind	Name
0x0001	ADVISORY_NEW
0x0002	ADVISORY_REVOKE
0x0003	ADVISORY_RETIRE
0x0004	ADVISORY_UPDATE
0x0005	ADVISORY_REGISTRY_REFRESH

Table 8

12.3. ADVISORY_NEW (0x0001)

Registers a new alert origin. Receivers MUST add the entry to their local registry and update their stored registry version to new_registry_version. In the case nodes receive a valid ADVISORY_NEW packet that collides with the current registry, nodes MUST drop that packet and refuse update, and SHOULD do a full resync of its local origin registry.

Offset	Size	Field	Type
0x0A	8	new_registry_version	u64
0x12	4	origin_key_id	u32
0x16	32	pubkey_ed25519	u8[32]

Table 9

Minimum packet size: 8 (prefix) + 2 (kind) + 44 (payload) + 64 (signature) = 118 bytes

12.4. ADVISORY_REVOKE (0x0002)

Emergency revocation of a compromised or rogue alert origin. Receivers MUST immediately remove the identified origin from their local registry and reject any further ALERTs signed by it, regardless of signature validity.

This packet SHOULD have URGENT set.

Offset	Size	Field	Type
0x0A	8	new_registry_version	u64
0x12	4	origin_key_id	u32

Table 10

Minimum packet size: 8 + 2 + 12 + 64 = 86 bytes

12.5. ADVISORY_RETIRE (0x0003)

Planned decommission of an alert origin. Receivers MUST remove the identified origin from their local registry and update their stored registry version.

Unlike ADVISORY_REVOKE, retirement is planned and does not imply compromise. URGENT SHOULD NOT be set.

Offset	Size	Field	Type
0x0A	8	new_registry_version	u64
0x12	4	origin_key_id	u32

Table 11

Minimum packet size: 86 bytes

12.6. ADVISORY_UPDATE (0x0004)

Notifies nodes of a scheduled WARN protocol update. Implementations MAY ignore it. It carries no enforcement.

Offset	Size	Field	Type
0x0A	1	version_major	u8
0x0B	1	version_minor	u8
0x0C	8	scheduled_update_s	u64

Table 12

Minimum packet size: 8 + 2 + 10 + 64 = 84 bytes

12.7. ADVISORY_REGISTRY_REFRESH (0x0005)

Signals that the registry has been updated and nodes SHOULD re-sync via the info-plane. Carries the authoritative current registry version so receivers can determine whether they are behind.

Offset	Size	Field	Type
0x0A	8	current_registry_version	u64

Table 13

Receivers that find their local registry version behind current_registry_version SHOULD fetch the full registry from the info-plane.

Minimum packet size: 8 + 2 + 8 + 64 = 82 bytes

13. Seeding Model

* Alert Origin: signs and issues ALERT packets.

* Data Relay: receives alerts from Alert Origins and performs first-hop seeding into the mesh.

Goals:

* Multiple independent entry points into the mesh

* Delivery over precision

All alert-plane propagation from Alert Origins MUST pass through at least one Data Relay before reaching leaf clients. General WARN coordination packets MAY originate from any node. Transport-specific seeding rules (relay discovery, registration, forwarding topology) are defined in per-transport specifications. See WARNIP.

14. Freshness and Replay Windows

REQUIRED cache length:

* Replay cache duration >= ttl_s

15. Transport Constraints

Transport-specific constraints such as NAT traversal, port binding, and client keepalive are defined in per-transport specifications.

16. Compliance Targets

16.1. Relay MUST

* Bounds-check packets
* Resolve origin_key_id via Origin Registry
* Verify signature
* Enforce time-based TTL
* Enforce replay protection
* Forward immutable packets

16.2. Client MUST

* Bounds-check packets
* Resolve origin_key_id via Origin Registry
* Verify signature
* Enforce freshness

17. Reference Sizes (ALERT, no TLV)

* Common prefix: 8 bytes
* ALERT fixed fields: 56 bytes
* Signature block: 68 bytes
* Total: 132 bytes

18. Origin Registry Format

This section defines a simple local file or in-memory region that maps origin_key_id -> public key.

This file is NOT transmitted on the alert-plane. How it is distributed/updated is out of scope.

18.1. Top-Level Structure

* registry_version: u64, used to manage deltas and versions. The registry version MUST monotonically increase. This MUST match the newest version that the node has obtained, either via a sync or ADVISORY.

18.2. Origin Entry

Each entry MUST contain:

* origin_key_id: integer (must fit u32)

* pubkey: Raw Ed25519 public key (32 bytes)

18.3. Required Receiver Behavior (Authorization)

Receivers MUST:

* Reject ALERTs if origin_key_id does not exist in the registry.

* Verify Ed25519 signature using the registry pubkey.

* Remove origins immediately upon receiving a valid ADVISORY_REVOKE or ADVISORY_RETIRE signed by the master origin.

19. Mesh Isolation

Although WARN alerts are made to be usable in any region, nations SHOULD isolate their mesh against potential attacks by neighboring nations via a false alert. In such cases, nations SHOULD compile in their own keys for their nation, and isolate their trust system.

20. Transportation and Auxiliary Infrastructure

WARN only declares the common protocol which all devices using WARN must be able to parse. Therefore info-plane schemas, key distribution, and propagation will be medium-dependent. There are other specifications that are dependent on the medium, such as IP. Other mediums may distribute WARN messages natively with medium-specific framing. The auxiliary data may change, but the WARN packet itself will be preserved.

21. Security Considerations

The security and validity of WARN effectively relies on a single master key, which should be kept airtight, preferably using an HSM or equivalent security measure. If the master key is compromised, then a large-scale firmware update would be necessary for resetting keys.

Alert origins should keep their private keys secure, but in the case of a compromise, the master key should be able to revoke the compromised key fairly quickly. Malicious alert relays will not be able to issue alerts or affect another node unless they have a origin private key, and complying implementations will be able to drop

invalid packets either forged by a node or replayed from a previous time. A malicious fleet of relays can "dilute" the mesh depending on the medium. On mediums such as IP, if a fleet of relays register but drop packets on a real emergency alert, they may degrade the resilience of the network. A malicious fleet of relays will be able to effectively do a DoS attack on a relay to overwhelm its ability to process and validate the costly Ed25519 signature.

22. IANA Considerations

IANA will create a new registry group called "WARN." This group includes the "WARN Hazard Codes" "WARN Assigned Ranges" "WARN Advisory Codes" "WARN Response" "WARN Urgency" "WARN Severity" "WARN Certainty" registries described below.

22.1. WARN Hazard Codes

IANA will create the following registry:

Registry Name: WARN Hazard Codes

Registration Procedure: RFC Required

NOTE: The complete values of this table are TBD.

hazard_major	hazard_minor	Meaning
0	0	RESERVED (invalid)
1	0	Geophysical Unknown
1	1	Earthquake
1	2	Landslide
1	3	Tsunami
2	0	Meteorological Unknown
2	1	Storm
2	2	Flood
3	0	Safety Unknown
4	0	Security Unknown

4	1	Terrorism
4	2	Military Activity
5	0	Rescue Unknown
6	0	Fire Unknown
6	1	Wildfire
6	2	City Fire
6	3	Prescribed Fire
7	0	Health Unknown
8	0	Environmental Unknown
8	1	Air pollution
9	0	Transport Unknown
0x0A	0	Infra Unknown
0x0B	0	CBRNE Unknown
0xFF	0	Other

Table 14

22.2. WARN Response

IANA will create the following registry:

Registry Name: WARN Response

Registration Procedure: RFC Required

Value	Meaning
0	RESERVED(invalid)
1	All Clear
2	Assess
3	Avoid
4	Evacuate
5	Execute
6	Monitor
7	Prepare
8	Shelter
9	None

Table 15

22.3. WARN Urgency

IANA will create the following registry:

Registry Name: WARN Urgency

Registration Procedure: RFC Required

Value	Meaning
0	RESERVED(invalid)
1	Expected
2	Future
3	Immediate
4	Past
5	Unknown

Table 16

22.4. WARN Severity

IANA will create the following registry:

Registry Name: WARN Severity

Registration Procedure: RFC Required

Value	Meaning
0	RESERVED(invalid)
1	Minor
2	Moderate
3	Severe
4	Extreme
5	Unknown

Table 17

22.5. WARN Certainty

IANA will create the following registry:

Registry Name: WARN Certainty

Registration Procedure: RFC Required

Value	Meaning
0	RESERVED(invalid)
1	Unlikely
2	Likely
3	Possible
4	Observed
5	Unknown

Table 18

22.6. WARN Reserved Ranges

IANA will create the following registry:

Registry Name: WARN Reserved Ranges

Registration Procedure: RFC Required

Range/Value	Category
0x0000	RESERVED(invalid)
0x0001-0x00FF	WARN
0x0100-0x01FF	IPWARN
0x0200-0xFEFF	Future use
0xFF00-0xFFFE	Private use
0xFFFF	RESERVED(invalid)

Table 19

22.7. WARN Advisory Codes

IANA will create the following registry:

Registry Name: WARN Advisory Codes

Registration Procedure: RFC Required

Kind	Name
0x0001	ADVISORY_NEW
0x0002	ADVISORY_REVOKE
0x0003	ADVISORY_RETIRE
0x0004	ADVISORY_UPDATE
0x0005	ADVISORY_REGISTRY_REFRESH

Table 20

23. References

23.1. Normative References

[RFC8032] Josefsson, S. and I. Liusvaara, "Edwards-Curve Digital Signature Algorithm (EdDSA)", RFC 8032, DOI 10.17487/RFC8032, January 2017, <https://www.rfc-editor.org/info/rfc8032>.

[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>.

23.2. Informative References

[CAP] Westfall, J., Jones, E., and OASIS, "Common Alerting Protocol Version 1.2", 2010, <https://docs.oasis-open.org/emergency/cap/v1.2/CAP-v1.2-os.html>.

[RFC7946] Butler, H., Daly, M., Doyle, A., Gillies, S., Hagen, S., and T. Schaub, "The GeoJSON Format", RFC 7946, DOI 10.17487/RFC7946, August 2016, <https://www.rfc-editor.org/info/rfc7946>.

Author’s Address

Shunta Koga
Email: kogashunta@gmail.com