Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: 3 December 2026

C. Pearson
Independent Submission

1 June 2026

Lowest Common Denominator Protocol (LCDP)
draft-pearson-lcdp-02

Abstract

The Lowest Common Denominator Protocol (LCDP) is a message-oriented, peer-to-peer wire format consisting of UTF-8 encoded JSON arrays of externally tagged objects transported over datagrams. LCDP provides perpetual compatibility by extension rather than versioning: unknown message types and fields are ignored. This document describes the wire format, core message types for peer discovery and anti-spoofing, and the design rationale. Security, reliability, and congestion control are delegated to optional messages or higher layers.

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 3 December 2026.

Copyright Notice

Table of Contents

1. Introduction..... 2
2. Terminology..... 2
3. Wire Format..... 2
4. Core Message Types..... 3
5. Design Considerations..... 4
6. Security Considerations..... 4
7. IANA Considerations..... 5
8. Normative References..... 5
9. Informative References..... 5
Acknowledgments..... 6
Author’s Address..... 6

1. Introduction

Many peer-to-peer protocols begin by establishing a connection-oriented session over UDP, then layer streams, negotiation, and mandatory cryptography on top. LCDP takes an alternative approach: define a minimal datagram message format and allow applications to build additional properties only as needed.

The complete wire format is a UTF-8 encoded JSON array. Each element is a single-key object where the key identifies the message type. Receivers MUST ignore objects with unknown keys and MUST ignore unknown fields within known objects.

This document is Informational and is an Independent Submission. It does not define a Standards Track protocol and makes no claim to replace existing protocols such as CoAP [RFC7252].

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.

3. Wire Format

1. Transport: UDP [RFC768] is RECOMMENDED. Other datagram transports MAY be used. Port 24254 is used by convention.

2. Encoding: Each datagram payload MUST be valid UTF-8 [RFC8259] encoding a single JSON array. Array elements MUST be objects with exactly one key.

3. Extensibility: Receivers MUST ignore objects with unknown keys and MUST ignore unknown fields within known objects. New message types MAY be defined at any time. New fields MAY be added to existing message types. The semantics of existing fields MUST NOT change.

4. MTU Considerations: Senders SHOULD limit datagrams to 5888 bytes to avoid excessive IP fragmentation, and MAY limit datagrams to 1200 bytes.

4. Core Message Types

4.1. Anti-Spoofing

To mitigate reflection and amplification attacks, implementations MUST implement a source address verificitaion mechanism.

Implementations SHOULD verify source addresses by exchanging:

```
[{"PleaseAlwaysReturnThisMessage":["cookie","string"]}]
[{"AlwaysReturned":["cookie","string"]}]
```

Per [RFC8085] Section 6, receivers MUST NOT reply with responses larger than twice the request size to unverified sources.

4.2. Optional Peer Discovery

```
[{"PleaseSendPeers":{}}]
[{"Peers":{"peers":["198.51.100.1:24254"]}}]
[{"WhereAreThey":{"ed25519h":"hex"}}]
```

The Peers message contains an array of address:port strings. The address format is not constrained by this specification.

4.3. Optional Cryptography

```
[{"MyPublicKey":{"ed25519h":"hex"}}]
[{"SignedMessage":{"ed25519":"hex","signature":"base64",
"payload_json":"string"}}]
[{"EncryptedMessages":{"base64":"string","noise_params":"string"}}]
```

Authentication and encryption are provided on a per-message basis and are OPTIONAL. This differs from transport-layer security models.

5. Design Considerations

LCDP separates mechanism from policy:

1. Statelessness: No connection state is required. This permits communication with large numbers of peers with minimal per-peer state.

2. Extensibility: Compatibility is achieved by ignoring unknown data, not by version negotiation. Incompatible changes require new message types.

3. Transport Agnosticism: While UDP is RECOMMENDED, the format may be carried over other datagram services.

4. Debuggability: The use of UTF-8 JSON permits inspection with standard tools such as tcpdump -A.

6. Security Considerations

LCDP does not provide security properties in the base protocol. Security is delegated to message types defined by applications.

1. Amplification: Implementations MUST implement rate limiting as described in [RFC8085] Section 6. The cookie mechanism described in Section 4.1 is RECOMMENDED.

2. Congestion Control: UDP provides no congestion control. Per [RFC8085] Section 3.1 applications that send more than one packet per RTT to a destination MUST implement congestion control. Low-rate gossip applications MAY operate without it.

3. Privacy: Peer discovery messages reveal IP address and port information. Deployments MUST consider whether this exposure is acceptable for their threat model.

4. Parser Security: Implementations MUST use memory-safe JSON parsers and MUST reject messages with excessive nesting depth or size.

7. IANA Considerations

This document has no IANA actions.

8. 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/info/rfc2119>.

[RFC768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, DOI 10.17487/RFC768, August 1980, <https://www.rfc-editor.org/info/rfc768>.

[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>.

[RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <https://www.rfc-editor.org/info/rfc8259>.

[RFC8085] Eggert, L., Fairhurst, G., and G. Shepherd, "UDP Usage Guidelines", BCP 145, RFC 8085, DOI 10.17487/RFC8085, March 2017, <https://www.rfc-editor.org/info/rfc8085>.

9. Informative References

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <https://www.rfc-editor.org/info/rfc7252>.

[RFC9518] Thaler, D., Ed., "Centralization, Decentralization, and Internet Standards", RFC 9518, DOI 10.17487/RFC9518, December 2023, <https://www.rfc-editor.org/info/rfc9518>.

Acknowledgments

The design of LCDP is informed by operational experience with peer-to-peer systems and by the observations in [RFC9518] regarding protocol design trade-offs in 2026.

Author’s Address

Christopher Pearson
Independent
Email: lcdp@azai.net

Additional Resources

https://github.com/kermit4/LCDP