Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: 27 September 2026

A. John

26 March 2026

DNS-Based Content Delivery & Fallback Mechanism
draft-dns-content-delivery-00

Abstract

This document specifies a mechanism for serving content, such as HTML or JSON, directly via DNS TXT records. This feature is intended as a fallback mechanism when a primary service (A/AAAA record) is unreachable, or as a lightweight hosting solution for parked domains to display landing pages without requiring active HTTP servers or individual SSL certificates. Trust is established via DNSSEC, allowing browsers to treat the content as secure.

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 27 September 2026.

Table of Contents

1. Introduction

Managing HTTP servers and HTTPS certificates for a large number of parked domains or "placeholder" sites can be resource-intensive and operationally complex. Additionally, when a web server fails (connection refused or timeout), users are typically presented with a generic browser error page.

This specification defines the "DNS Content" (DNSC) protocol, which allows User Agents (UAs) to retrieve content directly from the DNS system using structured TXT records. This feature is intended as a fallback mechanism when a primary service (A/AAAA record) is unreachable, or as a lightweight hosting solution for parked domains to display landing pages without requiring active HTTP servers or individual SSL certificates.

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.

This document makes use of terminology for record types (TXT), and other technical terms that are specific to the DNS. Since these terms have specific meanings in the DNS, they are not expanded upon first use in this document. For definitions of these and other terms, see [RFC8499].

3. The _dnscontent Resource Record

Content is published using DNS TXT records located at a specific prefix indicating the content type.

3.1. Naming Convention

The TXT record MUST be located at a label constructed as follows:

_<media-subtype>._dnscontent.<domain>

* media-subtype corresponds to the IANA media type subtype (e.g., html for text/html, json for application/json).

* domain corresponds to the FQDN of the service (e.g., example.com).

Examples:

* _html._dnscontent.example.com implies Content-Type: text/html

* _json._dnscontent.api.example.com implies Content-Type: application/json

3.2. Record Syntax (Root Record)

The TXT record content is a semi-colon separated list of key-value pairs. The keys are case-insensitive.

v=DNSC1; [e=<encoding>;] [id=<stream-id>;] [tot=<total-chunks>;] c=<content>

Parameter	Name	Description
v	Version	REQUIRED. Must be DNSC1.
e	Encoding	OPTIONAL. Specifies the compression algorithm applied to the content _before_ Base64 encoding.
id	Stream ID	OPTIONAL (REQUIRED if tot > 1). A unique alphanumeric identifier (e.g., a short hash) for this version of the content. Used to correlate chunks.
tot	Total Chunks	OPTIONAL. Integer indicating the total number of records (chunks) required to reconstruct the content. Default is 1.
c	Content	REQUIRED. The payload data encoded in *Base64*. If chunking is used, this field contains the _first_ chunk (Sequence 0).

Table 1: Parameters

The parameter e (Encoding) can have following the values:

* raw (Default): No compression.

* gzip: GZIP compression [RFC1952].

* br: Brotli compression [RFC7932].

3.3. Chunking Mechanism

DNS messages have size constraints. To support content larger than a single TXT record’s safe limit, the content MAY be split across multiple records.

3.3.1. Chunk Discovery

If tot > 1 in the Root Record, the UA MUST fetch additional records. Subsequent chunks are located at subdomains prefixed with the sequence number (1-based index).

<sequence-number>.<media-subtype>._dnscontent.<domain>

3.3.1.1. Example

If Root is _html._dnscontent.example.com, then:

* Chunk 1 is at _1._html._dnscontent.example.com
* Chunk 2 is at _2._html._dnscontent.example.com
* ...up to tot - 1.

3.3.2. Chunk Record Syntax

v=DNSC1; id=<stream-id>; c=<content>

* *v*: Must be DNSC1.
* *id*: MUST match the id provided in the Root Record. UAs MUST discard chunks with mismatched IDs to prevent mixing versions during updates.
* *c*: The Base64 encoded payload for this segment.

4. Client Behavior

4.1. Trigger Conditions

A UA SHOULD initiate a DNSC lookup under the following conditions:

1. The UA fails to fetch A or AAAA records for a specified domain (NXDOMAIN or NOERROR/NODATA).
2. The UA fails to establish a TCP connection to the IP addresses resolved from A/AAAA records (e.g., Connection Refused, Timed Out).
3. A mechanism (e.g., a specific URI scheme) explicitly requests a DNSC request.

4.2. Processing Logic

1. The UA constructs the query name based on the desired content type (_html for web navigation).
2. Query TXT for _<media-subtype>._dnscontent.<domain>.
3. Parse the TXT record. If v is not DNSC1, ignore.
4. Fetch Chunks (if needed):

   * If tot > 1, loop from i = 1 to tot - 1.

   * Query TXT for _<i>._<media-subtype>._dnscontent.<domain>.

   * Validate id.

   * Concatenate the Base64 strings from Root then _1, _2, etc.

5. Decode:

   * Base64 Decode the assembled string.

   * Decompress based on the e parameter (gzip, br, or none).

6. Display the content with the implied Content-Type.

5. Security Considerations

5.1. Trust & Origin

Since DNSC does not use TLS certificates (X.509), trust is established via DNSSEC.

* If the domain is signed with DNSSEC and the chain of trust is validated by the UA (or the recursive resolver trusted by the UA), the content MUST be treated as a *secure Context*.
  - UAs MAY display a specific indicator (e.g., "Verified by DNSSEC").
* If DNSSEC validation fails or the zone is unsigned, the content MUST be treated as an insecure Context (similar to plain HTTP).
  - Powerful features (Geolocation, Service Workers, etc.) MUST be disabled.

5.2. Mixed Content & Subresources

* Relative Links should be resolved relative to the domain root.
* External Resources:
  - In a secure Context (DNSSEC), resources (JS, CSS, Images) SHOULD be loaded over HTTPS or verified DNSC.
  - In an insecure Context, standard Mixed Content blocking rules apply.

5.3. Denial of Service

To prevent loops or excessive DNS traffic:

* UAs SHOULD enforce a limit on the maximum number of chunks (tot). A recommended limit is 16 chunks.

* UAs SHOULD implement caching for DNSC records respecting the TTL.

Appendix A. Examples

A.1. Simple Landing Page (No Chunking)

```
_html._dnscontent.example.com.  IN TXT  "v=DNSC1; e=raw; c=PGgxPldlbGNvbWU8L2gxPg=="
```

Decodes to <h1>Welcome</h1>

A.2. Compressed & Chunked Content

Root Record:

```
_html._dnscontent.domain.com.   IN TXT  "v=DNSC1; e=br; id=req89; tot=2; c=G873..."
```

Chunk 1:

```
 _1._html._dnscontent.domain.com. IN TXT  "v=DNSC1; id=req89; c=...B729"
```

The UA fetches both, verifies id=req89 matches, concatenates the Base64 payloads, decodes, decompresses using Brotli, and renders.

Author’s Address

April Faye John
Germany
Email: aprl@sakamoto.pl