Network Working Group
Internet-Draft
Intended status: Informational
Expires: 13 September 2026

J. Chen
Tsinghua University

12 March 2026

# Defensive Handling of MIME Parsing Ambiguities in Email Delivery
## draft-chen-email-mime-ambiguity-defense-00

## Abstract

Email security gateways and endpoint mail clients frequently rely on different MIME parsers, decoders, and error-recovery behavior. An attacker can exploit those differences so that a security control fails to extract or scan an attachment that a downstream client later exposes to a user. This document describes defensive processing guidance for SMTP receivers, mail gateways, and message stores that handle MIME messages with malformed or ambiguous structure.

This document provides operational guidance for ingress validation, strict decoding floors, ambiguity detection, multi-view extraction, union scanning, logging, and policy handling. It also defines an optional "MIME-Ambiguity-Results" header field for conveying receiver-generated ambiguity assessments to downstream components inside an administrative domain.

## 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 13 September 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

Email attachment defenses often assume that the object scanned by a gateway is the same object that a receiving mail client will later present for download or execution. That assumption is not always true. Divergent handling of malformed or ambiguous MIME can create a gap between the detector-side view and the client-side view of the same message. That gap can be exploited to evade attachment detection.

The problem is operational rather than purely theoretical: deployed products differ in how they resolve duplicate or conflicting header fields, how they parse multipart boundaries, and how they decode

malformed transfer encodings. This document provides defensive guidance intended to ensure that a receiving system scans at least every attachment view that mainstream clients could plausibly expose, or else blocks or quarantines the message.

This document is intentionally scoped to receiver-side defenses. It does not attempt to standardize all client parser behavior, nor does it provide exploit construction guidance.

## 2. Requirements Language and Conventions

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.

"Detector side" means any SMTP receiver, mail gateway, content filter, malware scanner, sandbox, or message store component that parses or scans inbound content before user access. "Client side" means the mail user agent or webmail interface that renders message structure or makes attachments available for download.

"Strict parse" means message parsing and decoding that follows Internet Message Format and MIME specifications, including the baseline decoding semantics required by those specifications. "Compatible parse" means a receiver-controlled parsing path used to approximate tolerated client behavior without inventing new semantics beyond what deployed clients are known to expose.

"Attachment view" means the set of extracted byte sequences that a given parsing path would make available to a user as attachments, downloadable body parts, or equivalent objects.

## 3. Threat Model

The attacker is assumed to be able to send email to the target domain. The attacker need not control the target mail system, the malware detection engine, or the recipient account. The attack succeeds if a message is accepted and delivered without the malicious content being blocked or quarantined, and if a downstream client later exposes an attachment or downloadable object that was not effectively covered by the detector-side scan.

This document addresses attacks that rely on parser disagreement, malformed structure, conflicting MIME metadata, or divergent decoding. It does not attempt to solve malicious content classification in the absence of such ambiguity.

## 4. Defensive Goals

A receiver implementing this document aims to satisfy the following goals:

1. A scanned attachment set MUST be at least as broad as any attachment view that a supported downstream client could plausibly expose.

2. If that condition cannot be met with sufficient confidence, the message MUST be rejected, quarantined, or sanitized according to local policy.

3. Receiver behavior SHOULD favor deterministic and auditable handling over heuristic repair that could create new message semantics.

4. Deployments SHOULD support phased rollout, beginning with logging and policy reporting before enabling blocking behavior.

## 5. Receiver Processing Model

A receiver implementing this specification SHOULD process inbound messages using the following high-level sequence:

1. Ingress structural validation

2. Strict parsing and extraction

3. Compatible parsing and extraction

4. Construction of a union attachment view

5. Scanning of every extracted object in that union

6. Disposition according to local policy

7. Optional emission of receiver-generated ambiguity results

Receivers MAY combine or pipeline these steps internally, but the effective security outcome MUST be equivalent.

### 5.1. Ingress Structural Validation

Before normal delivery, a receiver SHOULD evaluate the message for structural conditions that are highly correlated with parser disagreement. At minimum, implementations SHOULD detect the following classes of conditions:

1. duplicate or conflicting MIME structural header fields, including multiple Content-Type fields with different effective values;

2. control characters, including NUL, in MIME-relevant header field names or values;

3. multipart bodies with absent, empty, or otherwise invalid boundary parameters;

4. use of RFC 2047 encoded-word syntax inside MIME parameter values, such as boundary or filename, where such usage is not permitted;

5. decoding anomalies in transfer encodings that are known to create divergent extraction outcomes; and

6. malformed line folding or abnormal header/body separation that can change message interpretation.

A receiver MUST classify each detected condition as either:

* fatal: the message cannot be trusted to have a single safe interpretation and MUST be rejected or quarantined; or

* ambiguous: the message might still be processable, but additional extraction and union scanning are required before any delivery decision.

Empty multipart boundaries, NUL in MIME-relevant header data, and directly conflicting MIME structural header fields SHOULD be treated as fatal by default.

### 5.2. Strict Parsing and Decoding Floor

A receiver MUST implement at least one strict parsing path grounded in [RFC5322], [RFC2045], [RFC2046], [RFC2047], and [RFC2183].

For transfer encodings, a receiver MUST NOT implement a decoding behavior that is weaker than the minimum semantics already required by the MIME specifications. In particular, if a MIME decoding rule requires tolerant handling of certain non-alphabet characters or whitespace, a receiver MUST NOT stop extraction earlier than that specification permits if doing so would produce a narrower scan view than a conformant client could expose.

### 5.3. Compatible Parsing

A receiver SHOULD implement at least one receiver-controlled compatible parsing path to approximate attachment views that common downstream clients may expose in practice. The purpose of the compatible path is defensive coverage, not message repair for end-user fidelity.

A compatible path MUST be constrained so that it does not invent new attachment semantics unsupported by realistic client behavior. Compatible parsing SHOULD be derived from observed receiver or client interoperability needs, regression testing, or differential parser analysis.

### 5.4. Union Extraction and Scanning

A receiver that performs both strict and compatible parsing MUST form a union attachment view from all extracted objects. Every object in that union MUST be subject to the same malware detection, content policy, archive expansion, and sandboxing controls that would apply to a normal attachment.

If any object in the union is classified as malicious or disallowed, the receiver MUST apply that disposition to the message as a whole, unless local policy instead replaces the object with a safe, auditable sanitization result.

If the union attachment view differs from the strict attachment view, the receiver MUST treat the message as ambiguous. Local policy MAY still permit delivery after successful scanning, but the default action SHOULD be quarantine or other restricted handling.

### 5.5. Resource Limits and Abuse Resistance

Because multi-view parsing and scanning can expand resource consumption, implementations MUST enforce limits on message size, extracted object count, nested multipart depth, recursive archive expansion, decoding output size, and processing time. Messages that exceed such limits MUST fail closed, typically by quarantine or rejection.

## 6. Minimum Anomaly Classes

The following anomaly classes form a minimum common vocabulary for receiver implementations. Implementations MAY define additional local classes.

dup-content-type Duplicate or conflicting Content-Type fields or

parameter interpretations that could change body-part structure.

nul-in-header NUL or comparable control characters in MIME-relevant header field names or values.

empty-boundary Missing or empty multipart boundary values, or equivalent boundary invalidity causing structure disagreement.

invalid-b64-char Base64 decoding anomalies that alter extraction outcome across implementations.

qp-broken-softbreak Quoted-printable soft line break anomalies that can alter recovered bytes or part delimitation.

encoded-word-in-parameter Use of RFC 2047 encoded-word syntax in MIME parameters where not permitted.

Receivers SHOULD log anomaly classes in structured security telemetry even when local policy ultimately delivers the message.

## 7. SMTP Handling and Disposition

If a receiver detects a fatal ambiguity during SMTP transaction processing, it MAY reject the message during or immediately after DATA. Enhanced status codes from the 5.6.x or 5.7.x classes are generally appropriate depending on whether the receiver treats the event as a format violation or a security-policy violation; exact code selection is a local policy matter.

If the receiver accepts the message first and later determines that it is fatally ambiguous or malicious, it MUST prevent unrestricted user access, for example by quarantine, silent administrative hold, or bounded sanitization with operator auditability.

## 8. The MIME-Ambiguity-Results Header Field

This section defines an OPTIONAL receiver-generated header field, MIME-Ambiguity-Results, for use within an administrative domain. The field communicates whether the receiver detected MIME ambiguity and what disposition was applied.

This field is not an originator assertion. It MUST be inserted only by trusted receiving infrastructure. Downstream consumers MUST ignore instances that originate outside the local trust boundary.

### 8.1. Syntax

The syntax in this section is described using ABNF [RFC5234]. The FWS, CFWS, and CRLF rules are imported from [RFC5322]. The authserv-id, token, and value rules are imported from [RFC8601].

```
MIME-Ambiguity-Results = "MIME-Ambiguity-Results:" FWS authserv-id
                         *( CFWS ";" CFWS mar-param ) CRLF

mar-param    = mar-result / mar-policy / mar-anomaly / mar-ext
mar-result   = "result=" ( "pass" / "ambiguous" / "fail" )
mar-policy   = "policy=" ( "accept" / "quarantine" /
                           "reject" / "sanitize" )
mar-anomaly  = "anomaly=" anomaly-code
anomaly-code = "dup-content-type" /
               "nul-in-header" /
               "empty-boundary" /
               "invalid-b64-char" /
               "qp-broken-softbreak" /
               "encoded-word-in-parameter" /
               x-anomaly
x-anomaly    = "x-" 1*(ALPHA / DIGIT / "-")
mar-ext      = token ["=" value]
```

The ALPHA and DIGIT rules are imported from [RFC5234].

### 8.2. Semantics

result indicates the receiver’s overall ambiguity assessment. policy indicates the disposition taken by the receiver. anomaly identifies one or more anomaly classes that contributed to the assessment.

A receiver SHOULD place this field near other receiver-generated assessment fields. A downstream consumer that uses the field for policy decisions MUST rely only on instances inserted by trusted infrastructure inside the same administrative domain.

### 8.3. Example

```
MIME-Ambiguity-Results: mx.example.net; result=ambiguous;
  policy=quarantine; anomaly=dup-content-type;
  anomaly=invalid-b64-char
```

## 9. Operational Deployment Guidance

Deployments SHOULD introduce these checks in stages. A common rollout sequence is:

1. log-only anomaly detection;

2. logging plus internal reporting via MIME-Ambiguity-Results or equivalent telemetry;

3. quarantine for fatal anomalies and union-view disagreement; and

4. selective SMTP rejection for classes shown to be low-noise and high confidence.

Implementers SHOULD maintain a regression corpus of malformed and ambiguous messages and SHOULD use differential testing against supported downstream clients to verify that detector-side coverage remains at least as broad as client-side exposure.

## 10. Security Considerations

This entire document is about security. The central security property is coverage equivalence: the detector-side scan view must not be narrower than the client-side exposure view.

Overly aggressive message repair can itself create security problems. Receivers SHOULD avoid speculative rewriting that changes message structure or attachment semantics in ways not directly justified by local sanitization policy.

The MIME-Ambiguity-Results header field is trustworthy only within a local administrative trust boundary. Attackers can forge the field in received messages; therefore downstream consumers MUST ignore untrusted instances.

Multi-view parsing increases computational cost and therefore creates a denial-of-service risk. Implementations MUST enforce hard resource limits and fail closed when those limits are exceeded.

## 11. Privacy Considerations

Union extraction and scanning can cause more message content to be processed than would be visible under a single parser. Operators SHOULD review retention, access control, and data handling policies for extracted objects and scanner outputs.

Receivers SHOULD avoid placing unnecessary high-entropy content-derived identifiers in MIME-Ambiguity-Results. If deployments need richer forensic linkage, they SHOULD prefer internal telemetry systems over header fields that may later be forwarded outside the original trust boundary.

## 12. IANA Considerations

This document has no IANA actions.

The MIME-Ambiguity-Results header field defined in this document is intended for use within an administrative domain. This document does not request registration of that field in an IANA message header field registry.

## 13. Normative References

[RFC2045] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, DOI 10.17487/RFC2045, November 1996, <https://www.rfc-editor.org/info/rfc2045>.

[RFC2046] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, DOI 10.17487/RFC2046, November 1996, <https://www.rfc-editor.org/info/rfc2046>.

[RFC2047] Moore, K., "MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text", RFC 2047, DOI 10.17487/RFC2047, November 1996, <https://www.rfc-editor.org/info/rfc2047>.

[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>.

[RFC2183] Troost, R., Dorner, S., and K. Moore, Ed., "Communicating Presentation Information in Internet Messages: The Content-Disposition Header Field", RFC 2183, DOI 10.17487/RFC2183, August 1997, <https://www.rfc-editor.org/info/rfc2183>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

[RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322, DOI 10.17487/RFC5322, October 2008, <https://www.rfc-editor.org/info/rfc5322>.

[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>.

[RFC8601] Kucherawy, M., "Message Header Field for Indicating Message Authentication Status", RFC 8601, DOI 10.17487/RFC8601, May 2019, <https://www.rfc-editor.org/info/rfc8601>.

## 14. Informative References

[INBOXINVASION] Zhang, J., Chen, J., Wang, Q., Zhang, H., Wang, C., Zhuge, J., and H. Duan, "Inbox Invasion: Exploiting MIME Ambiguities to Evade Email Attachment Detectors", CCS 2024, October 2024, <https://www.jianjunchen.com/p/inbox-invasion.CCS24.pdf>.

## Author’s Address

Jianjun Chen
Tsinghua University
China
Email: jianjun@tsinghua.edu.cn