Network Working Group
Request for Comments: 3459
Updates: 3204
Category: Standards Track

E. Burger
SnowShore Networks

January 2003

Critical Content Multi-purpose Internet Mail Extensions (MIME) Parameter

Status of this Memo

Copyright Notice

Abstract

This document describes the use of a mechanism for identifying body parts that a sender deems critical in a multi-part Internet mail message. The mechanism described is a parameter to Content-Disposition, as described by RFC 3204.

By knowing what parts of a message the sender deems critical, a content gateway can intelligently handle multi-part messages when providing gateway services to systems of lesser capability. Critical content can help a content gateway to decide what parts to forward. It can indicate how hard a gateway should try to deliver a body part. It can help the gateway to pick body parts that are safe to silently delete when a system of lesser capability receives a message. In addition, critical content can help the gateway chose the notification strategy for the receiving system. Likewise, if the sender expects the destination to do some processing on a body part, critical content allows the sender to mark body parts that the receiver must process.

Table of Contents

1. Conventions used in this document

This document refers generically to the sender of a message in the masculine (he/him/his) and the recipient of the message in the feminine (she/her/hers). This convention is purely for convenience and makes no assumption about the gender of a message sender or recipient.

The key words "MUST", "MUST NOT", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14, RFC 2119 [2].

The word "REQUIRED" in this document does not follow the definition found in RFC 2119. This is because this document defines a parameter named "REQUIRED". There is no requirement in this document that is "REQUIRED", so there is no confusion.

In this document, the "sending agent" is the originator of the message. It could be a mail user agent (MUA) for an Internet message, or a SIP User Agent Client (UAC) for a SIP [3] message. The "endpoint" is the receiving device, of lesser capability than the sending agent.

NOTE: Notes, such as this one, provide additional nonessential information that the reader may skip without missing anything essential. The primary purpose of these non-essential notes is to convey information about the rationale of this document, or to place this document in the proper historical or evolutionary context. Readers whose sole purpose is to construct a conformant implementation may skip such information. However, it may be of use to those who wish to understand why we made certain design choices.

2. Introduction

The specification of Critical Content is small and compact. For the benefit of developers, the specification comes first, the rationale after.

One concept that an implementer must understand is the content gateway. Section 10 describes the content gateway. In brief, a content gateway has knowledge of the receiving system's capabilities. The content gateway passes messages the receiving system can process, render or store. The content gateway can modify a message, for example by deleting unrenderable or storable body parts, for delivery

to the receiving system. Finally, the content gateway can reject a message that the receiving system cannot handle.

Although Critical Content processing is not an OPES service, the protocol machinery described in this document meets all of the OPES IAB requirements as stated by RFC 3238 [4]. Section 14 describes this in detail. In particular, unlike the current situation where content gateways silently modified messages, or had abstract rules for modifying them (see the content transformation rules in VPIM, for example), the Critical Content mechanism allows for the sending user to explicitly indicate desired content handling by content gateways

NOTE: This document updates RFC 3204 [5] to separate the Handling parameter from the ISUP/QSIG transport mechanism. The protocol described here is identical in functionality to RFC 3204 with respect to SIP. Future versions of RFC 3204 should reference this document for the Handling parameter, as it is orthogonal to the tunneling of signaling.

3. Handling Parameter

The Handling parameter is a Content-Disposition [6] parameter inserted by the sending agent to indicate to the content gateway whether to consider the marked body part critical.

A REQUIRED body part is one the sender requires the receiving system to deliver for him to consider the message delivered.

An OPTIONAL body part is one the sender doesn’t care whether the receiving system delivers it or not. A content gateway can silently delete such body parts if the receiving system cannot deliver the part.

The terms "entity" and "body part" have the meanings defined in [6].

3.1. REQUIRED

"Handling=REQUIRED" signifies that this body part is critical to the sender.

If the content gateway cannot pass a body part marked REQUIRED, then the entire message has failed. In this case, the content gateway MUST take the appropriate failure action.

NOTE: We say "appropriate action", because the sender may have suppressed all notifications. In this case, the appropriate action is to silently discard the message. In addition, as a general MIME parameter, the MIME body part may not be in an Internet Mail message.

Moreover, in the SIP case, the appropriate notification is a status return code, not a delivery notification.

3.2. OPTIONAL

"Handling=OPTIONAL" signifies that the sender does not care about notification reports for this body part.

If the content gateway cannot pass a body part marked OPTIONAL, the receiving system may silently delete the body part. The receiving system MUST NOT return a delivery failure, unless parts marked REQUIRED have also failed.

3.3. Default Values

The default value for Handling for a given body part is REQUIRED. This enables the existing notification mechanisms to work for sending agents that do not know about the content notification entity. All body parts are critical, because they have the default marking of REQUIRED.

NOTE: In the case of Internet mail, critical content processing is a function of the content gateway and not the mail transfer agent (MTA) or user agent (UA). Often, the entity performing content gateway processing is the receiving UA. However, in this case the UA is acting as a content gateway. Thus the default action for any Content-Disposition [6]-compliant user agent to ignore unrecognized disposition parameters ensures that this mechanism is compatible with the Internet architecture.

NOTE: This parameter is fully backwards compatible and works as expected for Internet mail and SIP.

NOTE: Some VPIMv2 implementations can receive arbitrary e-mail from the Internet. However, these systems are really acting in the capacity of an Internet Voice Mail system. In this case, one would expect the implementation to provide Internet Voice Mail semantics to Internet Voice Mail messages.

3.4. Other Values

The content gateway MUST treat unrecognized values as REQUIRED. This is to provide backward compatibility with future uses of the Content-Criticality entity.

NOTE: A possible new value is IMPORTANT. An IMPORTANT body part is something the sender wants the receiver to get, but would not want the message rejected outright if the IMPORTANT body part fails, but

they do want notification of the failure. However, as no implementations do IMPORTANT, it is not important to this version of this document.

4. Collected Syntax

The format of the collected syntax is in accordance with the ABNF of [7]. Note that per RFC 2183 [6], the HANDLING Content-Disposition parameter is not case sensitive. In addition, the notification-type is not case sensitive.

```
"handling" "=" notification-type CRLF

notification-type = "REQUIRED" / "OPTIONAL" /
                    other-handling / generic-param

other-handling    =  token
```

5. Notification

One obvious application of critical content is generating a (non-) delivery notification in the Internet mail environment. If the value of the field is OPTIONAL, the content gateway MUST NOT generate a notification. If the value of the field is REQUIRED, the content gateway MAY generate a notification, based on the normal notification request mechanisms. Normal notification request mechanisms include specifying the NOTIFY parameter to the SMTP RCPT command [8] and the Disposition-Notification-To header [9].

In SIP, all requests have responses. These responses provide notification in the status code of the response. For the RFC 3204 case, a content gateway generates a 415 (Unsupported Media Type) response if the field is REQUIRED.

If the sending system requests a notification, and a REQUIRED part fails, the content gateway MUST generate a notification for the whole message. Conversely, if the gateway cannot pass on a body part marked OPTIONAL, the gateway MUST NOT generate a notification.

NOTE: This implies that the content gateway must examine the entire message to determine whether it needs to generate a notification. However, the content gateway need not examine the message if it knows it can store and forward all media types. Said differently, Internet e-mail MTAs or gateways can, by default, handle any arbitrary MIME-encapsulated type. Some voice mail systems, on the other hand, cannot store binary attachments at all, such as application/ms-word. The voice mail content gateway, in this example, would be scanning for non-renderable body parts in any event.

5.1. DSN vs. MDN Generation

The content gateway generates a delivery status notification (DSN) [9] if it operates as a gateway. The content gateway generates a Message Disposition Notification (MDN) [10] if it operates as a mail user agent. Section 6 describes the operating modes of a content gateway. In short, if there is a MTA that "delivers" the message to the content gateway for processing, the MTA takes responsibility for DSN processing. In this case, the only option available to the content gateway is to generate MDNs. If the content gateway operates as a MTA, then it generates DSNs. DSN generation is the preferred option.

If the content gateway is part of a SIP endpoint, then it generates the appropriate success or error response code.

5.2. Summary

The following table summarizes the actions expected of a conforming content gateway.

NOTE: This section is normative: it suggests what a content gateway should put into the DSN or MDN.

NOTE: In the case of SIP, this section is informative. See RFC 3204 for the normative set of actions on failure.

Table 1 - Expected Actions

	Sending UA Has Marked Body Part	
	REQUIRED	OPTIONAL
Body Part is Deliverable	Appropriate Action	ignore
Body Part is Undeliverable	Fail Entire Message	ignore

The "Appropriate Action" is the action the content gateway would take given the context of execution. For example, if a sender requests return receipt and the receiver reads a HANDLING body part, the receiving UA must generate the appropriate MDN (following the rules for MDN). Likewise, if the content gateway cannot deliver the body part and the body part is critical, the content gateway generates the appropriate DSN or MDN.

"Optional" means the content gateway ignores the disposition of the body part. The content gateway treats the message as if the body part was not present in the message.

6. Signed Content

RFC 1847 [11] describes how to apply digital signatures to a MIME body part. In brief, a multipart/signed body part encapsulates the body part of interest, or the "content object", in a MIME body part and the control information needed to verify the object, or the "protocol" in the lexicon of RFC 1847, in a second MIME body part. Here is an example taken from RFC 1847.

```
Content-Type: multipart/signed; protocol="TYPE/STYPE";
        micalg="MICALG"; boundary="Signed Boundary"

--Signed Boundary
Content-Type: text/plain; charset="us-ascii"

This is some text to be signed although it could be
any type of data, labeled accordingly, of course.

--Signed Boundary
Content-Type: TYPE/STYPE

CONTROL INFORMATION for protocol "TYPE/STYPE" would be here

--Signed Boundary--
```

Figure 1 - Signed Content MIME Type

There are three places where one may place the criticality indicator for a multipart/signed body part. One could mark the multipart/signed object, the content object, the control object, or any combination of the three.

The disposition of REQUIRED body parts follow the guidelines found in RFC 2480 [12].

A critical content indicator on a multipart/signed body part means the sending party requires true end-to-end signature verification. Thus the gateway needs to pass the enclosure intact. If the system or network of lesser capability cannot do signature verification and the signed enclosure is REQUIRED, the gateway MUST reject the message.

A critical content indicator on a signature means that either the receiving endpoint must be able to do signature verification, or the gateway needs to verify the signature before forwarding the message. If the content does not pass verification, the gateway MUST reject the message.

A critical content indicator on the enclosed material specifies whether that material is critical to the message as a whole. If the signature is marked OPTIONAL and the enclosed material is marked REQUIRED, the gateway MAY strip out the signature information if the system or network of lesser capability cannot do signature verification. However, if possible, we STRONGLY RECOMMEND the gateway do signature verification and indicate tampering to the recipient.

7. Encrypted Content

RFC 1847 [11] describes how to encrypt a MIME body part. In brief, a multipart/encrypted body part encapsulates the control information ("protocol" in the lexicon of RFC 1847) for the encrypted object and the second containing the encrypted data (application/octet-stream). Here is an example taken from RFC 1847.

```
Content-Type: multipart/encrypted; protocol="TYPE/STYPE";
        boundary="Encrypted Boundary"

--Encrypted Boundary
Content-Type: TYPE/STYPE

CONTROL INFORMATION for protocol "TYPE/STYPE" would be here

--Encrypted Boundary
Content-Type: application/octet-stream

    Content-Type: text/plain; charset="us-ascii"
    All of this indented text, including the indented headers,
    would be unreadable since it would have been encrypted by
    the protocol "TYPE/STYPE". Also, this encrypted data could
    be any type of data, labeled accordingly, of course.

--Encrypted Boundary--
```

One may sensibly place a criticality indicator on the encrypted enclosure (multipart/encrypted) body part. If the endpoint can decrypt the message, then the gateway passes the body part in its entirety.

If one marks the control object REQUIRED, then the sending UA requires end-to-end encryption. If the endpoint cannot decrypt the message, then the gateway MUST reject the message.

If the control object is OPTIONAL, and the endpoint cannot decrypt the message, and the gateway can decrypt the message, then the gateway MAY decrypt the message and forward the cleartext message. The sending user has explicitly given permission for the gateway to decrypt the message by marking the control object OPTIONAL. Recall that the default indication for MIME body parts is REQUIRED. Thus if the user takes no explicit action, the content gateway will assume the user wished end-to-end encryption.

Marking the encrypted content, without marking the encrypted enclosure, is problematic. This is because the gateway has to decrypt the encrypted data to retrieve the header. However, it is unlikely for the gateway to have the capability (e.g., keys) to decrypt the encrypted data. If a sending UA wishes to mark encrypted data as not REQUIRED, the sending UA MUST mark the encrypted content as not REQUIRED. Clearly, if the sending UA marks the encrypted content as REQUIRED, the gateway will apply the REQUIRED processing rules. Moreover, if the sending UA does not mark the encrypted content as REQUIRED, the gateway, unless it can decrypt the data, will treat the encrypted content as REQUIRED. This occurs because gateways always treat unmarked content as REQUIRED (see Section 3.3).

8. Status Code

The critical content indication, in itself, does not guarantee any notification. Notification follows the rules described in [3], [8], and [9].

NOTE: The content of actual DSNs or MDNs are beyond the scope of this document. This document only specifies how to mark a critical body part. On the other hand, we do envision sensible DSN and MDN contents. For example, DSNs should include the appropriate failure code as enumerated in [13]. Likewise, MDNs should include the failure code in the MDN "Failure:" field.

If the receiving system is to generate a notification based on its inability to render or store the media type, the notification should use the status code 5.6.1, "Media not supported", from [10].

For the SIP case, all requests have notification provided by the status response message. Per RFC 3204, a content gateway generates a 415 (Unsupported Media Type) response.

9. Requirements for Critical Content

This section is informative.

9.1. Needs

The need for a critical content identification mechanism comes about because of the internetworking of Internet mail systems with messaging systems that do not fulfill all of the semantics of Internet mail. Such legacy systems have a limited ability to render or store all parts of a given message. This document will use the case of an Internet mail system exchanging electronic messages with a legacy voice messaging system for illustrative purposes.

Electronic mail has historically been text-centric. Extensions such as MIME [14] enable the user agents to send and receive multi-part, multimedia messages. Popular multimedia data types include binary word processing documents, binary business presentation graphics, voice, and video.

Voice mail has historically been audio-centric. Many voice-messaging systems only render voice. Extensions such as fax enable the voice mail system to send and receive fax images as well as create multi-part voice and fax messages. A few voice mail systems can render text using text-to-speech or text-to-fax technology. Although theoretically possible, none can today render video.

An important aspect of the interchange between voice messaging services and desktop e-mail client applications is that the rendering capability of the voice-messaging platform is often much less than the rendering capability of a desktop e-mail client. In the e-mail case, the sender has the expectation that the recipient receives all components of a multimedia message. This is so even if the recipient cannot render all body parts. In most cases, the recipient can either find the appropriate rendering tool or tell the sender that she cannot read the particular attachment.

This is an important issue. By definition, a MIME-enabled user agent, conforming to [15], will present or make available all of the body parts to the recipient. However, a voice mail system may not be capable of storing non-voice objects. Moreover, the voice mail system may not be capable of notifying the recipient that there were undeliverable message parts.

The inability of the receiving system to render a body part is usually a permanent failure. Retransmission of the message will not improve the likelihood of a future successful delivery. Contrast this with the case with normal data delivery. Traditional message

failures, such as a garbled message or disabled link will benefit from retransmission.

This situation is fundamentally different from normal Internet mail. In the Internet mail case, either the system delivered the message, or it didn't. There is no concept of a system partially delivering a message.

In addition, there are many situations where the sender would not mind if the system did not deliver non-critical parts of a message. For example, the sender's user agent may add body parts to a message unbeknownst to the sender. If the receiving system rejected the message because it could not render a hidden body part, the sender would be understandably confused and upset.

Thus, there is a need for a method of indicating to a Mail Transfer Agent (MTA) or User Agent (UA) that the sender considers parts of a message to be critical. From the sender's perspective, he would not consider the message delivered if the system did not deliver the critical parts.

9.2. Current Approaches

One method of indicating critical content of a message is to define a profile. The profile defines rules for silently deleting mail body parts based on knowledge of the UA capabilities. Citing the example above, a voice profile can easily declare that MTAs or UAs can silently delete TNEF data and yet consider the message successfully delivered. This is, in fact, the approach taken by VPIMv2 [16].

Since one aspect of the issue is deciding when to notify the sender that the system cannot deliver part of a message, one could use a partial non-delivery notification mechanism to indicate a problem with delivering a given body part. However, this requires the user request a delivery notification. In addition, the sender may not be aware of parts added by the sending user agent. In this case, a failure notice would mystify the sender.

A straightforward alternative implementation method for marking a body part critical is to use a Critical-Content MIME entity. This has the benefit that criticality is meta information for the body part. However, IMAP servers in particular would need to either put Critical-Content into the BODYSTRUCTURE method or create a new method to retrieve arbitrary MIME entities. Given the experience of trying to get Content-Location accepted by IMAP vendors, we chose not to go that route.

What we need is a way of letting the sender indicate what body parts he considers to be critical. The mechanism must not burden the sender with failure notifications for non-critical body parts. The mechanism must conform to the general notification status request mechanism for positive or negative notification. When requested, the mechanism must indicate to the sender when a receiving system cannot deliver a critical body part.

10. The Content Gateway

This section is informative.

In this section, we use the definition found in RFC 2156 [17] for the term "gateway."

We do not strictly use the definition found in RFC 2821 [18] for the term "gateway." In particular, RFC 2821 is discussing a gateway that should not examine the message itself. An RFC 2821 gateway is a transport gateway, that mostly deals with transformations of the SMTP information.

A content gateway is a gateway that connects a first network to a second network. The second network often has lesser capability than the first network. The canonical topology follows. "[MTA]", with square brackets, signifies an optional component.

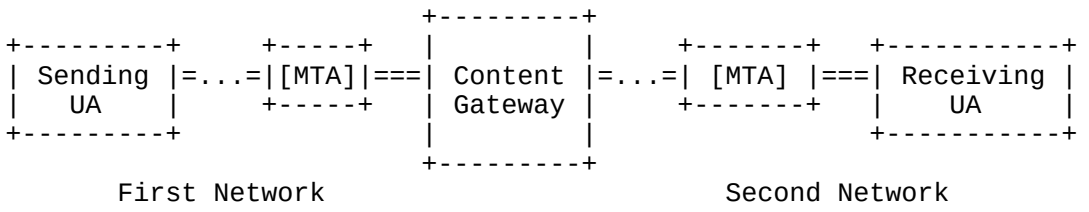

Figure 2 - Content Gateway Topology

The content gateway can be the last hop before the receiving MTA. The content gateway can be between networks, and thus not the last hop before the receiving MTA. The content gateway can be the first MTA the sending UA contacts. Finally, the content gateway can be an integrated component of the receiving MTA.

For the SIP case, consider each MTA as a SIP Proxy, the Sending UA as a SIP User Agent Client, and the Receiving UA as a SIP User Agent Server.

10.1. Integrated Content Gateway

In this situation, the receiving user agent is integrated with the content gateway. The integrated content gateway knows the capabilities of the user agent. The topology is as follows.

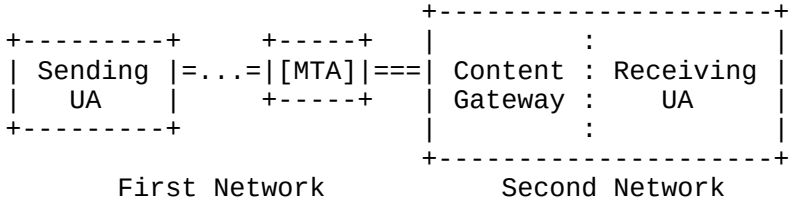

Figure 3 - Integrated Content Gateway

The processing of ISUP and QSIG objects, as described in [5], is an example of an integrated gateway.

10.2. Disaggregated Delivery Network

A degenerate case, although one that does occur, is where the content gateway sits behind the final MTA. This happens when one implements the content gateway as a post-processing step to a normal delivery. For example, one could configure a mail handling system to deliver the message to a queue or directory, where the content gateway process picks up the message. If there were any directives for DSN processing, the delivering MTA would execute them. For example, the message could have requested notification on successful delivery. The delivering MTA, having delivered the message to the queue, would consider the message delivered and thus notify the sender of such. However, the content gateway process could then discover that the receiving UA cannot render the message. In this case, the content gateway generates a NDN, as it is the only option available.

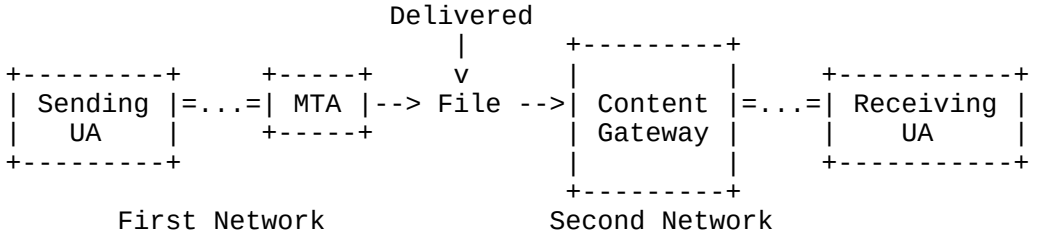

Figure 4 - Disaggregated Delivery Network

11. Backward Compatibility Considerations

DSN requires ESMTP. If MTAs in the path from the sending UA to the receiving UA do not support ESMTP, then that MTA will reject the DSN request. In addition, the message will default to notification on delay or failure. While not ideal, the sender will know that DSN is not available, and that critical content that fails will get notification.

12. MIME Interactions

12.1. multipart/alternative

As is true for all Content-Disposition parameters, handling is only in effect for the selected alternative. If the selected alternative has the critical content indicator, then the entire alternative takes on the criticality indicated. That is, if the alternative selected has HANDLING=OPTIONAL, then the content gateway MUST NOT generate any delivery notifications.

NOTE: This statement explicitly shows that HANDLING overrides the DSN and MDN request mechanisms.

It is unlikely for a selected alternative to fail, as the content gateway presumably picks the alternative specifically because it can render it.

If the selected alternative is a message/rfc822 that encloses a multipart MIME message or the selected alternative is itself a multipart MIME type, the individual top-level body parts follow the HANDLING mechanism described in this document.

NOTE: This means that a forwarded message’s criticality will not affect the forwarding agent’s intentions.

12.2. multipart/related

Criticality fits in rather well with the multipart/related construction. For example, consider a multipart/related message consisting of a Macintosh data fork and a Macintosh resource fork. For a Microsoft Word document, the data fork is likely to be critical. The receiving system can safely ignore the resource fork.

12.3. message/rfc822

Criticality only affects the outermost level of the message or, in the case of multipart/alternative, the outermost level of the selected alternative. Specifically, the receiving system ignores

criticality indicators in embedded body parts. This avoids the situation of a forwarded message triggering or suppressing undesired reporting. This simply implements the procedures described in [6].

12.4. multipart/signed

See Section 6.

12.5. multipart/encrypted

See Section 7.

13. Implementation Examples

This section is an informative part of the definition of Criticality. We hope it helps implementers understand the mechanics of the Handling mechanism.

We will examine two cases. They are how a content gateway processes a message and how a disaggregated content gateway processes a message.

13.1. Content Gateways

Content gateways examine the contents of a message from a first network before the gateway forwards the message to a second network. For the purposes of this example, we assume the second network has less capability than the first network. In particular, we expect there will be certain message body types that the gateway cannot pass onto the second network.

Consider a gateway between the Internet and a text-only short message service. A message comes through the gateway containing a text part and a tnef part. The sender marks the text part REQUIRED. The gateway, knowing the capability of the short message service, silently deletes the non-critical, tnef part, passing the critical content to the short message service network. Any subsequent notifications, such as failure notices or delivery notices, follow the normal rules for notification.

Note the gateway, by silently deleting non-critical content, may affect proprietary message correlation schemes. One can envision the sending UA inserting a body part for tracking purposes. By deleting non-critical content, the content gateway will break such a scheme. If a sending UA understands how to mark critical content, it should use Internet standard mechanisms for tracking messages, such as Message-ID [19].

What if no body parts have critical content indicators? In this case, the entire message is critical. Thus, when the gateway sees the tnef part, it will reject the entire message, generating a DSN with a status code 5.6.1, "Media not supported".

Likewise, consider a three part message with a text annotation (part 1) to a voice message (part 2) with a vCard [20] (part 3). The sender marks the first two parts REQUIRED. Now, let us assume the receiving MTA (gateway) is a voice mail only system, without even the capability to store text. In this case, the gateway, acting as the receiving MTA, will reject the message, generating a DSN with the status code 5.6.1, "Media not supported".

13.2. Disaggregated Content Gateway

For this example, we will examine the processing of a three-part message. The first part is a text annotation of the second part, an audio message. The third part is the sender's vCard. The sender marks the first and second parts REQUIRED. In addition, the sender marks the message for read receipt.

For the purposes of example, the telephone user interface (TUI) does not perform text-to-speech conversion. A TUI is a mail user agent (UA) that uses DTMF touch-tone digits for input and audio for output (display).

The TUI is unable to render the first part of the message, the text part. In addition, it is unable to render the third part of the message, the vCard part. Since the sender did not mark the third part of the message REQUIRED, the system ignores the failure of the TUI to render the third part of the message. However, since the sender did mark the first part REQUIRED, and the TUI is unable to render text, the message fails.

What happens next is implementation dependent. If the TUI is part of a unified messaging system, a reasonable action is to hold the message for the user. The user can access the message at a later time from a terminal that can render all of the critical body parts. It would be reasonable for the TUI to notify the user about the undeliverable body part.

If the TUI is part of a voice messaging system, or if the user does not subscribe to a text-to-speech service, a reasonable action is for the TUI to return a MDN with the disposition "failed" and the failure modifier "5.6.1 (Media not supported)".

14. OPES Considerations

Critical Content processing is not a web service. However, some in the Internet community may draw parallels between web services that modify content and an e-mail, SIP, or other MIME-transport service that modifies content.

This section will analyze the Critical Content protocol machinery against the requirements stated in RFC 3238 [4]. The summary is that the protocol described in this document meets all of the requirements of RFC 3238.

14.1. Consideration (2.1): One-Party Consent

This is the heart of Critical Content. Critical Content enables the sending party to give consent to have the message modified. Gateways that conform to this document will ensure that gateways only modify messages that the sending party has given consent to modify.

14.2. Consideration (2.2): IP-layer Communications

The content gateway is an addressable IP-entity. Moreover, all of the relevant protocols (SMTP, SIP, HTTP, etc.) all explicitly make the presence of the gateway known to the endpoints.

14.3. Consideration (3.1): Notification - Sender

Again, this is the point of this document. The sender explicitly gets notification if the gateway would remove a Critical Content body part.

14.4. Consideration (3.2): Notification - Receiver

The nature of the receiving system dictates that end users understand that the messages have been changed.

14.5. Consideration (3.3): Non-Blocking

By definition, the endpoint cannot receive non-modified content, so this requirement does not apply.

14.6. Consideration (4.1): URI Resolution

Clearly, one is sending mail (SMTP), a message (SIP), or fetching a document (HTTP). The machinery described in this document does not alter the content itself or the access mechanism. Thus it is compliant with this requirement.

14.7. Consideration (4.2): Reference Validity

Since the protocol described in this document does not alter the content itself, inter- and intra-document references are not altered. However, intra-document references to removed body parts will fail. On the other hand, the sender explicitly marked those body parts as being disposable. Thus the sender is aware of the possibility the parts may not arrive at the receiver.

14.8. Consideration (4.3): Architecture Extensions

Since the protocol described in this document meets Considerations 4.1 and 4.2, this requirement does not apply.

14.9. Consideration (5.1): Privacy

The privacy policy of this protocol is explicit. In particular, the protocol honors end-to-end security.

15. Security Considerations

Sending UA's can use signatures over critical content indicators to ensure the integrity of the indicator.

The gateway MUST honor signature processing. In particular, if the sending UA marks the signature components REQUIRED, and the endpoint cannot do MIME signature processing, the gateway MUST establish an appropriate signature mechanism between the gateway and the endpoint. In this case, the gateway must be secure, as it can become a target point for tampering with the signed components of the message.

Receiving systems and users should not place any authentication value on the Handling parameter.

Note that by design, and under the sending user's request, a content gateway will silently delete unimportant body parts. Critical content gives the sender the ability to determine the acceptable level integrity of the delivered message. That is, the message as the content gateway actually passes it on is, in fact, representative of the sender's intentions.

16. IANA Considerations

RFC 3204 already registered the Handling parameter. It is collected here only for reference and as a placeholder for use both for further expansion in the future and as the normative reference for other documents that need to reference the Handling parameter.

Per section 9 of [6], here is the IANA registration for Handling.

To: IANA@IANA.ORG Subject: Registration of new Content-Disposition parameter

Content-Disposition parameter name: HANDLING

Allowable values for this parameter: REQUIRED OPTIONAL

Description: Marks the body part as required for delivery (REQUIRED) or can be silently discarded (OPTIONAL). See RFC <this document> and RFC 3204.

Per RFC 2183, the Content-Disposition parameter name is not case sensitive. Per RFC 3459, the values of the parameter are also not case sensitive.

17. References

17.1 Normative References

[1] Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.

[2] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[3] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, P., Sparks, R., Handley, M. and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[4] IAB, Floyd, S. and L. Daigle, "IAB Architectural and Policy Considerations for Open Pluggable Edge Services", RFC 3238, January 2002.

[5] Zimmerer, E., Peterson, E., Vemuri, A., Ong, L., Audet, F., Watson, M. and M. Zonoun, "MIME media types for ISUP and QSIG Objects", RFC 3204, December 2001.

[6] Troost, R., Dorner, S. and K. Moore, Ed., "Communicating Presentation Information in Internet Messages: The Content-Disposition Header Field", RFC 2183, August 1997.

[7] Crocker, D. and P. Overell, Eds., "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

[8] Moore, K., "Simple Mail Transfer Protocol (SMTP) Service Extension for Delivery Status Notifications (DSNs)", RFC 3461, January 2003.

[9] Moore, K. and G. Vaudreuil, "An Extensible Message Format for Delivery Status Notifications", RFC 3464, January 2003.

[10] Fajman, R., "An Extensible Message Format for Message Disposition Notifications", RFC 2298, March 1998.

[11] Galvin, J., Murphy, S., Crocker, S. and N. Freed, "Security Multiparts for MIME: Multipart/Signed and Multipart/Encrypted", RFC 1847, October 1995.

[12] Freed, N., "Gateways and MIME Security Multiparts", RFC 2480, January 1999.

[13] Vaudreuil, G., "Enhanced Mail System Status Codes", RFC 3463, January 2003.

[14] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996.

[15] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, November 1996.

[16] Vaudreuil, G. and G. Parsons, "Voice Profile for Internet Mail - version 2", RFC 2421, September 1998.

[17] Kille, S., "MIXER (Mime Internet X.400 Enhanced Relay): Mapping between X.400 and RFC 822/MIME", RFC 2156, January 1998.

[18] Klensin, J., Ed., "Simple Mail Transfer Protocol", RFC 2821, April 2001.

[19] Crocker, D., "Standard for the Format of ARPA Internet Text Messages", RFC 822, August 1982.

17.2 Informative Reference

[20] Dawson, F. and T. Howes, "vCard MIME Directory Profile", RFC 2426, September 1998.

18. Acknowledgments

Emily Candell of Comverse Network Systems was instrumental in helping work out the base issues in the -00 document in Adelaide.

Ned Freed pointed out that this mechanism was about criticality, not notification. That insight made the concept and descriptions infinitely more straightforward. If it's still confusing, it's my fault!

Ned Freed also was instrumental in crafting the sections on multipart/signed and multipart/encrypted. As AD, he provided invaluable commentary to help progress this document.

Keith Moore for helped tighten-up the explanations, and he approved of the use of Content-Disposition.

Dropping the IMPORTANT critical content type took away one of the reasons for partial non-delivery notification. That makes Jutta Degener very happy!

Harald Alvestrand and Chris Newman suggested some implementation examples.

Greg White asked THE key question that let us realize that critical content processing was a gateway function, and not a MTA or UA function.

Jon Peterson cleared up how handling actually does work in the SIP environment.

An enormous thank you to Michelle S. Cotton at IANA for helping me craft the original IANA Considerations section in 2000, and for catching the functional overlap with RFC 3204 in January 2002.

Any errors, omissions, or silliness are my fault.

19. Intellectual Property Rights Notice

20. Author's Address

Eric Burger
SnowShore Networks, Inc.
285 Billerica Rd.
Chelmsford, MA 01824-4120
USA

Phone: +1 978 367 8400
Fax: +1 603 457 5944
EMail: e.burger@ieee.org

21. Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.