Network Working Group
Request for Comments: 5537
Obsoletes: 1036
Category: Standards Track

R. Allbery, Ed.
Stanford University
C. Lindsey

November 2009

Netnews Architecture and Protocols

Abstract

This document defines the architecture of Netnews systems and specifies the correct manipulation and interpretation of Netnews articles by software that originates, distributes, stores, and displays them. It also specifies the requirements that must be met by any protocol used to transport and serve Netnews articles.

Status of This Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Table of Contents

1. Introduction

1.1. Basic Concepts

"Netnews" is a set of protocols for generating, storing, and retrieving news "articles" whose format is defined in [RFC5536], and for exchanging them amongst a readership that is potentially widely distributed. It is organized around "newsgroups", with the expectation that each reader will be able to see all articles posted to each newsgroup in which he participates. These protocols most commonly use a flooding algorithm that propagates copies throughout a network of participating servers. Typically, only one copy is stored per server, and each server makes it available on demand to readers able to access that server.

"Usenet" is a particular worldwide, publicly accessible network based on the Netnews protocols. It is only one such possible network; there are deployments of the Netnews protocols other than Usenet (such as ones internal to particular organizations). This document discusses the more general Netnews architecture and protocols.

1.2. Scope

This document defines the architecture of Netnews systems and specifies the correct manipulation and interpretation of Netnews articles by software that originates, distributes, stores, and displays them. It addresses protocol issues that are independent of transport protocols such as the Network News Transfer Protocol (NNTP) [RFC3977], and specifies the requirements Netnews places on those underlying transport protocols. It also specifies the handling of control messages.

The format and syntax of Netnews articles are specified in [RFC5536], which should be read in conjunction with this document.

1.3. Requirements Notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

1.4. Syntax Notation

Syntax defined in this document uses the Augmented Backus-Naur Form (ABNF) notation (including the Core Rules) defined in [RFC5234] and constructs defined in [RFC5536] and [RFC5322].

The ABNF rules defined elsewhere and used in this document are:

```
CRLF                = <see [RFC5234] Appendix B.1>
DIGIT               = <see [RFC5234] Appendix B.1>
HTAB                = <see [RFC5234] Appendix B.1>
SP                  = <see [RFC5234] Appendix B.1>
WSP                 = <see [RFC5234] Appendix B.1>
VCHAR               = <see [RFC5234] Appendix B.1>

argument            = <see [RFC5536] Section 3.2.3>
article-locator     = <see [RFC5536] Section 3.2.14>
component           = <see [RFC5536] Section 3.1.4>
control-command     = <see [RFC5536] Section 3.2.3>
diag-keyword        = <see [RFC5536] Section 3.1.5>
diag-match          = <see [RFC5536] Section 3.1.5>
diag-other          = <see [RFC5536] Section 3.1.5>
dist-name           = <see [RFC5536] Section 3.2.4>
msg-id              = <see [RFC5536] Section 3.1.3>
newsgroup-name      = <see [RFC5536] Section 3.1.4>
path-diagnostic     = <see [RFC5536] Section 3.1.5>
path-identity       = <see [RFC5536] Section 3.1.5>
path-nodot          = <see [RFC5536] Section 3.1.5>
tail-entry          = <see [RFC5536] Section 3.1.5>
verb                = <see [RFC5536] Section 3.2.3>

display-name        = <see [RFC5322] Section 3.4>
local-part          = <see [RFC5322] Section 3.4.1>
mailbox             = <see [RFC5322] Section 3.4>
```

1.5. Definitions

Any term used in this document that is defined in Section 1.5 of [RFC5536] is used with the definition given there. In addition, the following terms will be used:

A "hierarchy" is the set of all newsgroups whose names share a first <component> (as defined in Section 3.1.4 of [RFC5536]). A "sub-hierarchy" is the set of all newsgroups whose names share several initial components.

A "news server" is further distinguished into the roles of "injecting agent", "relaying agent", and "serving agent". An "injecting agent" accepts a proto-article with the goal of distributing it to relaying and serving agents and hence to readers. A "relaying agent" accepts articles from other relaying agents or injecting agents and distributes them to other relaying agents or serving agents. A "serving agent" receives an article from a relaying agent or injecting agent and makes it available to readers.

A "user agent" is further distinguished into the roles of "posting agent" and "reading agent". A "posting agent" is software that assists in the preparation of a proto-article and then passes it to an injecting agent. A "reading agent" is software that retrieves articles from a serving agent for presentation to a reader.

"Injecting" an article is the processing of a proto-article by an injecting agent. Normally, this action is done once and only once for a given article. "Multiple injection" is passing the same article to multiple injecting agents, either serially or in parallel, by one or several posting agents.

A "gateway" is software that receives news articles and converts them to messages of some other kind (such as [RFC5322] mail messages), receives messages of some other kind and converts them to news articles, or conveys articles between two separate Netnews networks.

2. Transport

The exact means used to transmit articles from one agent to another is not specified. NNTP [RFC3977] is the most common transport mechanism for Netnews networks. Other methods in use include the Unix-to-Unix Copy Protocol [UUCP] (extensively used in the early days of Usenet) and physically delivered magnetic and optical media. Any mechanism may be used in conjunction with this protocol provided that it can meet the requirements specified here.

Transports for Netnews articles MUST treat news articles as uninterpreted sequences of octets, excluding the values %d00 (which may not occur in Netnews articles), %d13, and %d10 (which MUST only appear in Netnews articles as a pair in that order and which, together, denote a line separator). These octets are the US-ASCII [ASCII] characters NUL, CR, and LF respectively.

> NOTE: This corresponds to the range of octets permitted in MIME 8bit data [RFC2045]. Transports for Netnews are not required to support transmission of MIME binary data.

In particular, transports MUST convey all header fields unmodified (including header fields within message/rfc822 objects in article bodies), even if they contain octets in the range of 128 to 255. Furthermore, transports for relaying and serving agents MUST, and transports for other agents SHOULD, convey lines even if they exceed 998 characters in length, especially in article bodies. (This requirement is stricter than MIME 8bit data.) These requirements include the transport paths between posting agents, injecting agents, serving agents, and reading agents.

3. Duties of Agents

The following section specifies the duties of the agents involved in the creation, relaying, and serving of Netnews articles. This protocol is described by following the life of a typical Usenet article: it is prepared by a posting agent, given to an injecting agent, transferred through one or more relaying agents, accepted by a serving agent, and finally retrieved by a reading agent. Articles submitted to moderated groups go through an additional process, which is described separately (see Section 3.5.1 and Step 7 of Section 3.5). Finally, the additional duties and requirements of a gateway are discussed.

At each step, each agent has a set of checks and transformations of the article that it is required to perform. These are described as sequences of steps to be followed, but it should be understood that it is the effect of these sequences that is important, and implementations may use any method that produces the same effect.

Many news servers combine the functions of injecting agent, relaying agent, and serving agent in a single software package. For the purposes of this specification, such combined agents should conceptually be treated as an injecting agent that sends articles to a serving agent and, optionally, to a relaying agent. The requirements of all three agents MUST still be met when the news server is performing the functions of those agents.

On news servers that accept them, control messages may have additional effects than those described below. Those effects are described in Section 5.

3.1. General Principles

There are two important principles that news implementors and administrators need to keep in mind. The first is the well-known Internet Robustness Principle:

> Be liberal in what you accept, and conservative in what you send.

As applied to Netnews, this primarily means that unwanted or non-compliant articles SHOULD be rejected as early as possible, but once they are in general circulation, relaying and serving agents may wish to accept them where possible rather than lose information. Posting agents and injecting agents SHOULD therefore be maximally strict in their application of both this protocol and [RFC5536], and reading agents SHOULD be robust in the presence of violations of the Netnews article format where possible.

In the case of Netnews, there is an even more important principle, derived from a much older code of practice, the Hippocratic Oath (we may thus call this the Hippocratic Principle):

> First, do no harm.

It is vital to realize that decisions that might be merely suboptimal in a smaller context can become devastating mistakes when amplified by the actions of thousands of hosts within a few minutes.

No Netnews agent is ever required to accept any article. It is common for injecting, relaying, and serving agents to reject well-formed articles for reasons of local policy (such as not wishing to carry a particular newsgroup or attempting to filter out unwanted articles). This document specifies how articles are to be treated if they are accepted and specifies some cases where they must be rejected, but an agent MAY always reject any article for other reasons than those stated here.

A primary goal of the Netnews protocol is to ensure that all readers receiving a particular article (as uniquely identified by the content of its Message-ID header field) see the identical article, apart from allowable divergence in trace headers and local metadata. Accordingly, agents (other than moderators) MUST NOT modify articles in ways other than described here. Unacceptable articles MUST be rejected rather than corrected.

3.2. The Path Header Field

All news server components (injecting agents, relaying agents, and serving agents) MUST identify themselves, when processing an article, by prepending their <path-identity> (defined in Section 3.1.5 of [RFC5536]) to the Path header field. Injecting agents MUST also use the same identity in Injection-Info header fields that they add, and serving and relaying agents SHOULD use the same identity in any Xref header fields they add.

The <path-identity> used by an agent may be chosen via one of the following methods (in decreasing order of preference):

1. The fully qualified domain name (FQDN) of the system on which the agent is running.

2. A fully qualified domain name (FQDN) within a domain affiliated with the administrators of the agent and guaranteed to be unique by the administrators of that domain. For example, the

uniqueness of server.example.org could be guaranteed by the administrator of example.org even if there is no DNS record for server.example.org itself.

3. Some other (arbitrary) name in the form of a <path-nodot>, believed to be unique and registered at least with all the other news servers to which that relaying agent or injecting agent sends articles. This option SHOULD NOT be used unless the earlier options are unavailable or unless the name is of longstanding usage.

Some existing implementations treat <path-identity> as case-sensitive, some as case-insensitive. The <path-identity> therefore SHOULD be all lowercase and implementations SHOULD compare identities case-insensitively.

3.2.1. Constructing the Path Header Field

If a relaying or serving agent receives an article from an injecting or serving agent that is part of the same news server, it MAY leave the Path header field of the article unchanged. Otherwise, every injecting, relaying, or serving agent that accepts an article MUST update the Path header field as follows. Note that the Path header field content is constructed from right to left by prepending elements.

1. The agent MUST prepend "!" to the Path header field content.

2. An injecting agent SHOULD prepend the <path-diagnostic> "!.POSTED", optionally followed by "." and the FQDN or IP address of the source, to the Path header field content.

3. A relaying or serving agent SHOULD prepend a <path-diagnostic> to the Path header field content, where the <path-diagnostic> is chosen as follows:

   * If the expected <path-identity> of the source of the article matches the leftmost <path-identity> of the Path header field's content, use "!" (<diag-match>), resulting in two consecutive "!"s.

   * If the expected <path-identity> of the source of the article does not match, use "!.MISMATCH." followed by the expected <path-identity> of the source or its IP address.

   * If the relaying or serving agent is not willing or able to check the <path-identity>, use "!.SEEN." followed by the FQDN, IP address, or expected <path-identity> of the source.

The "expected <path-identity> of the source of the article" is a <path-identity> for the injecting or relaying agent that passed the article to this relaying or serving agent, determined by properties of the connection via which the article was received (for example, an authentication identity or a peer IP address). Be aware that [RFC1036] did not include <path-diagnostic>. Implementations that predate this specification will add only single "!" characters between <path-identity> strings.

4. The agent MAY then prepend to the Path header field content "!" or "!!" followed by an additional <path-identity> for itself other than its primary one. Using "!!", and thereby adding a <diag-match> since the <path-identity> clearly is verified, is RECOMMENDED. This step may be repeated any number of times. This is permitted for agents that have multiple <path-identity>s (such as during a transition from one to another). Each of these <path-identity>s MUST meet the requirements set out in Section 3.2.

5. Finally, the agent MUST prepend its primary <path-identity> to the Path header field content. The primary <path-identity> is the <path-identity> it normally advertises to its peers for their use in generating <path-diagnostic>s as described above.

Any agent that modifies the Path header field MAY fold it by inserting FWS (folding white space) immediately after any <path-identity> or <diag-other> it added (see Section 3.1.5 of [RFC5536] for allowable locations for FWS).

3.2.2. Path Header Field Example

Here is an example of a Path header field created by following the rules for injecting and relaying agents.

```
Path: foo.isp.example!.SEEN.isp.example!foo-news
  !.MISMATCH.2001:DB8:0:0:8:800:200C:417A!bar.isp.example
  !!old.site.example!barbaz!!baz.isp.example
  !.POSTED.dialup123.baz.isp.example!not-for-mail
```

This article was injected by baz.isp.example as indicated by the <diag-keyword> "POSTED". The injector has recorded that it received the article from dialup123.baz.isp.example. "not-for-mail" is a common <tail-entry>.

The article was relayed to the relaying agent known, at least to old.site.example, as "barbaz". That relaying agent confirmed to its satisfaction that "baz.isp.example" was an expected <path-identity> for the source of the article and therefore used <diag-match> ("!") for its <path-diagnostic>.

barbaz relayed it to old.site.example, which does not support <diag-keyword> and therefore used the old "!" delimiter. This indicates that the identity of "barbaz" was not verified and may have been forged.

old.site.example relayed it to a news server using the <path-identity> of bar.isp.example and claiming (by using the "!" <path-diagnostic>) to have verified that it came from old.site.example.

bar.isp.example relayed it to foo-news, which, not being convinced that it truly came from bar.isp.example, inserted the <diag-keyword> "MISMATCH" and then stated that it received the article from the IPv6 address [2001:DB8:0:0:8:800:200C:417A]. (This is not to say that bar.isp.example was not a correct <path-identity> for that source but simply that the identity did not match the expectations of foo-news.)

foo-news then passed the article to foo.isp.example, which declined to validate its <path-identity> and instead appended the <diag-keyword> "SEEN" to indicate it knows the source of the article as isp.example. This may be either an expected <path-identity> or the FQDN of the system from which it received the article. Presumably, foo.isp.example is a serving agent that then delivered the article to a reading agent.

baz.isp.example, bar.isp.example, and foo-news folded the Path header field.

3.3. Article History and Duplicate Suppression

Netnews normally uses a flood-fill algorithm for propagation of articles in which each news server offers the articles it accepts to multiple peers, and each news server may be offered the same article from multiple other news servers. Accordingly, duplicate suppression is key; if a news server accepted every article it was offered, it may needlessly accept (and then potentially retransmit) dozens of copies of every article.

Relaying and serving agents therefore MUST keep a record of articles they have already seen and use that record to reject additional offers of the same article. This record is called the "history" file or database.

Each article is uniquely identified by its message identifier, so a relaying or serving agent could satisfy this requirement by storing a record of every message identifier that agent has ever seen. Such a history database would grow without bound, however, so it is common and permitted to optimize based on the Injection-Date or Date header field of an article as follows. (In the following discussion, the "date" of an article is defined to be the date represented by its Injection-Date header field, if present; otherwise, by its Date header field.)

- Agents MAY select a cutoff interval and reject any article with a date farther in the past than that cutoff interval. If this interval is shorter than the time it takes for an article to propagate through the network, the agent might reject an article it had not yet seen, so it ought not to be aggressively short. For Usenet, for example, a cutoff interval of no less than seven days is conventional.

- Agents that enforce such a cutoff MAY then drop records of articles that had dates older than the cutoff from their history databases. If such an article were offered to the agent again, it would be rejected due to the cutoff date, so the history record is no longer required to suppress the duplicate.

- Alternatively, agents MAY drop history records according to the date when the article was first seen by that agent rather than the date of the article. In this case, the history retention interval MUST be at least 24 hours longer than the cutoff interval to allow for articles dated in the future. This interval matches the allowable error in the date of the article (see Section 3.5).

These are just two implementation strategies for article history, albeit the most common ones. Relaying and serving agents are not required to use these strategies, only to meet the requirement of not accepting an article more than once. However, these strategies are safe and widely deployed, and implementors are encouraged to use one of them, especially if they do not have extensive experience with Netnews and the subtle effects of its flood-fill algorithm.

3.4. Duties of a Posting Agent

A posting agent is the component of a user agent that assists a poster in creating a valid proto-article and forwarding it to an injecting agent.

Posting agents SHOULD ensure that proto-articles they create are valid according to [RFC5536] and any other applicable policies. They MUST NOT create any Injection-Info header field; this header field may only be added by the injecting agent.

If the proto-article already contains both Message-ID and Date header fields, posting agents MAY add an Injection-Date header field to that proto-article immediately before passing that proto-article to an injection agent. They SHOULD do so if the Date header field (representing the composition time of the proto-article) is more than a day in the past at the time of injection. They MUST do so if the proto-article is being submitted to more than one injecting agent; see Section 3.4.2.

The Injection-Date header field is new in this revision of the Netnews protocol and is designed to allow the Date header field to hold the composition date (as recommended in Section 3.6.1 of [RFC5322]), even if the proto-article is not to be injected for some time after its composition. However, note that all implementations predating this specification ignore the Injection-Date header field and use the Date header field in its stead for rejecting articles older than their cutoff (see Section 3.3), and injecting agents predating this specification do not add an Injection-Date header. Articles with a Date header field substantially in the past will still be rejected by implementations predating this specification, regardless of the Injection-Date header field, and hence may suffer poorer propagation.

Contrary to [RFC5322], which implies that the mailbox or mailboxes in the From header field should be that of the poster or posters, a poster who does not, for whatever reason, wish to use his own mailbox MAY use any mailbox ending in the top-level domain ".invalid" [RFC2606].

Posting agents meant for use by ordinary posters SHOULD reject any attempt to post an article that cancels or supersedes (via the Supersedes header field) another article of which the poster is not the author or sender.

3.4.1. Proto-Articles

A proto-article is an article in the format used by a posting agent when offering that article to an injecting agent. It may omit certain header fields that can be better supplied by the injecting agent and will not contain header fields that are added by the injecting agent. A proto-article is only for transmission to an injecting agent and SHOULD NOT be transmitted to any other agent.

A proto-article has the same format as a normal article except that the Injection-Info and Xref header fields MUST NOT be present, the Path header field SHOULD NOT contain a "POSTED" <diag-keyword>, and any of the following mandatory header fields MAY be omitted: Message-ID, Date, and Path. In all other respects, a proto-article MUST be a valid Netnews article. In particular, the header fields that may be omitted MUST NOT be present with invalid content.

If a posting agent intends to offer the same proto-article to multiple injecting agents, the header fields Message-ID, Date, and Injection-Date MUST be present and identical in all copies of the proto-article. See Section 3.4.2.

3.4.2. Multiple Injection of Articles

Under some circumstances (for example, when posting to multiple, supposedly disjoint, networks, when using injecting agents with spotty connectivity, or when desiring additional redundancy), a posting agent may wish to offer the same article to multiple injecting agents. In this unusual case, the goal is not to create multiple independent articles but rather to inject the same article at multiple points and let the normal duplicate suppression facility of Netnews (see Section 3.3) ensure that any given agent accepts the article only once, even if supposedly disjoint networks have unexpected links.

Whenever possible, multiple injection SHOULD be done by offering the same proto-article to multiple injecting agents. The posting agent MUST supply the Message-ID, Date, and Injection-Date header fields, and the proto-article as offered to each injecting agent MUST be identical.

In some cases, offering the same proto-article to all injecting agents may not be possible (such as when gatewaying, after injection, articles found on one Netnews network to another supposedly unconnected one). In this case, the posting agent MUST remove any Xref header field and rename or remove any Injection-Info, Path, and other trace header fields before passing it to another injecting agent. (This converts the article back into a proto-article.) It MUST retain unmodified the Message-ID, Date, and Injection-Date header fields. It MUST NOT add an Injection-Date header field if it is missing from the existing article.

> NOTE: Multiple injection inherently risks duplicating articles. Multiple injection after injection, by converting an article back to a proto-article and injecting it again, additionally risks loops, loss of trace information, unintended repeat injection into the same network, and other problems. It should be done with care

and only when there is no alternative. The requirement to retain Message-ID, Date, and Injection-Date header fields minimizes the possibility of a loop and ensures that the newly injected article is not treated as a new, separate article.

Multiple injection of an article that lists one or more moderated newsgroups in its Newsgroups header field SHOULD only be done by a moderator and MUST only be done after the proto-article has been approved for all moderated groups to which it is to be posted and after an Approved header field has been added (see Section 3.9). Multiple injection of an unapproved article intended for moderated newsgroups will normally only result in the moderator receiving multiple copies, and if the newsgroup status is not consistent across all injecting agents, may result in duplication of the article or other problems.

3.4.3. Followups

A followup is an article that contains a response to the contents of an earlier article, its precursor. In addition to its normal duties, a posting agent preparing a followup is also subject to the following requirements. Wherever in the following it is stated that, by default, a header field is said to be inherited from one of those header fields in the precursor, it means that its initial content is to be a copy of the content of that precursor header field (with changes in folding permitted). However, posters MAY then override that default before posting.

Despite the historic practice of some posting agents, the Keywords header field SHOULD NOT be inherited by default from the precursor article.

1. If the Followup-To header field of the precursor article consists of "poster", the followup MUST NOT be posted by default but, by default, is to be emailed to the address given in the precursor's Reply-To or From header field following the rules for an email reply [RFC5322]. This action MAY be overridden by the poster, in which case the posting agent should continue as if the Followup-To header field in the precursor did not exist.

2. The Newsgroups header field SHOULD, by default, be inherited from the precursor's Followup-To header field if present; otherwise, it is inherited from the precursor's Newsgroups header field.

3. The Subject header field SHOULD, by default, be inherited from that of the precursor. The case-sensitive string "Re: " (including the space after the colon) MAY be prepended to the content of its Subject header field unless it already begins with that string.

   NOTE: Prepending "Re: " serves no protocol function and hence is not required, but it is widely expected and not doing so would be surprising.

4. The Distribution header field SHOULD, by default, be inherited from the precursor's Distribution header field, if present.

5. The followup MUST have a References header field referring to its precursor, constructed in accordance with Section 3.4.4.

3.4.4. Construction of the References Header Field

The following procedure is to be used whenever some previous article (the "parent") is to be referred to in the References header field of a new article, whether because the new article is a followup and the parent is its precursor or for some other reason.

The content of the new article's References header field MUST be formed from the content of the parent's References header field if present, followed by the content of the Message-ID header field of the parent. If the parent had a References header, FWS as defined in [RFC5536] MUST be added between its content and the Message-ID header field content.

If the resulting References header field would, after unfolding, exceed 998 characters in length (including its field name but not the final CRLF), it MUST be trimmed (and otherwise MAY be trimmed). Trimming means removing any number of message identifiers from its content, except that the first message identifier and the last two MUST NOT be removed.

An essential property of the References header field, guaranteed by the above procedure and REQUIRED to be maintained by any extensions to this protocol, is that an article MUST NOT precede one of its parents.

3.5. Duties of an Injecting Agent

An injecting agent takes a proto-article from a posting agent and either forwards it to a moderator or passes it to a relaying or serving agent or agents. An injecting agent bears the primary responsibility for ensuring that any article it injects conforms with

the rules of the Netnews standards. The administrator of an injecting agent is also expected to bear some responsibility towards the rest of the Netnews network to which it is connected for the articles the injecting agent accepts.

Injecting agents, when rejecting articles, are encouraged to communicate the reason for rejection to the posting agent by using whatever facility is provided by the underlying transport. The injecting agent is in a unique position to communicate the reason for rejection; relaying agents and serving agents normally have to reject messages silently. The injecting agent therefore bears much of the burden of diagnosing broken posting agents or communicating policy violations to posters.

An injecting agent MUST have available a list (possibly empty) of moderated groups for which it accepts articles and the corresponding submission addresses. It SHOULD have available a list of valid newsgroups to catch articles not posted to a valid newsgroup and therefore likely to be silently discarded by relaying and serving agents. Usually, an injecting agent is deployed in conjunction with a serving agent and maintains these lists based on control messages received by the serving agent.

An injecting agent processes proto-articles as follows:

1. It SHOULD verify that the article is from a trusted source (for example, by relying on the authorization capability of the underlying transport used to talk to the posting agent).

2. It MUST reject any proto-article that does not have the proper mandatory header fields for a proto-article, that has Injection-Info or Xref header fields, that has a Path header field containing the "POSTED" <diag-keyword>, or that is not syntactically valid as defined by [RFC5536]. It SHOULD reject any proto-article that contains a header field deprecated for Netnews (see, for example, [RFC3798]). It MAY reject any proto-article that contains trace header fields (e.g., NNTP-Posting-Host) indicating that it was already injected by an injecting agent that did not add Injection-Info or Injection-Date.

3. It SHOULD reject any article whose Injection-Date or Date header field is more than 24 hours into the future (and MAY use a margin less than 24 hours). It SHOULD reject any article whose Injection-Date header field is too far in the past (older than the cutoff interval of a relaying agent that the injecting agent is using, for example). It SHOULD similarly reject any article whose Date header field is too far in the past, since not all news servers support Injection-Date and only the injecting agent

can provide a useful error message to the posting agent. In either case, this interval SHOULD NOT be any shorter than 72 hours into the past.

4. It SHOULD reject any proto-article whose Newsgroups header field does not contain at least one <newsgroup-name> for a valid group, or that contains a <newsgroup-name> reserved for specific purposes by Section 3.1.4 of [RFC5536] unless that specific purpose or local agreement applies to the proto-article being processed. Crossposting to unknown newsgroups is not precluded provided that at least one of the newsgroups in the Newsgroups header is valid.

5. The Message-ID and Date header fields with appropriate contents MUST be added when not present in the proto-article.

6. The injecting agent MUST NOT alter the body of the article in any way (including any change of Content-Transfer-Encoding). It MAY add other header fields not already provided by the poster, but injecting agents are encouraged to use the Injection-Info header for such information and to minimize the addition of other headers. It SHOULD NOT alter, delete, or reorder any existing header field except the Path header field. It MUST NOT alter or delete any existing Message-ID header field.

7. If the Newsgroups header contains one or more moderated groups and the proto-article does not contain an Approved header field, the injecting agent MUST either forward it to a moderator as specified in Section 3.5.1 or, if that is not possible, reject it. This forwarding MUST be done after adding the Message-ID and Date headers if required, and before adding the Injection-Info and Injection-Date headers.

8. Otherwise, a Path header field with a <tail-entry> MUST be added if not already present.

9. The injecting agent MUST then update the Path header field as described in Section 3.2.1.

10. An Injection-Info header field SHOULD be added that identifies the source of the article and possibly other trace information as described in Section 3.2.8 of [RFC5536].

11. If the proto-article already had an Injection-Date header field, it MUST NOT be modified or replaced. If the proto-article had both a Message-ID header field and a Date header field, an Injection-Date header field MUST NOT be added, since the proto-article may have been multiply injected by a posting agent that

predates this standard. Otherwise, the injecting agent MUST add an Injection-Date header field containing the current date and time.

12. Finally, the injecting agent forwards the article to one or more relaying agents, and the injection process is complete.

3.5.1. Forwarding Messages to a Moderator

An injecting agent MUST forward the proto-article to the moderator of the leftmost moderated group listed in the Newsgroups header field, customarily via email. There are two standard ways in which it may do this:

1. The complete proto-article is encapsulated, header fields and all, within the email. This SHOULD be done by creating an email message with a Content-Type of application/news-transmission with the usage parameter set to "moderate". The body SHOULD NOT contain any content other than the message. This method has the advantage of removing any possible conflict between Netnews and email header fields and any changes to those fields during transport through email.

2. The proto-article is sent as an email with the addition of any header fields required for an email as defined in [RFC5322], and possibly with the addition of other header fields conventional in email, such as To and Received. The existing Message-ID header field SHOULD be retained.

Although both of these methods have been used in the past and the first has clear technical advantages, the second is in more common use and many moderators are not prepared to deal with messages in the first format. Accordingly, the first method SHOULD NOT be used unless the moderator to which it is being forwarded is known to be able to handle this method.

NOTE: Deriving the email address of the moderator of a group is outside the scope of this document. It is worth mentioning, however, that a common method is to use a forwarding service that handles submissions for many moderated groups. For maximum compatibility with existing news servers, such forwarding services generally form the submission address for a moderated group by replacing each "." in the <newsgroup-name> with "-" and then using that value as the <local-part> of a <mailbox> formed by appending a set domain.

Forwarding proto-articles to moderators via email is the most general method and the most common in large Netnews networks such as Usenet, but any means of forwarding the article that preserves it without injecting it MAY be used. For example, if the injecting agent has access to a database used by the moderator to store proto-articles awaiting processing, it may place the proto-article directly into that database. Such methods may be more appropriate for smaller Netnews networks.

3.6. Duties of a Relaying Agent

A relaying agent accepts injected articles from injecting and other relaying agents and passes them on to relaying or serving agents. To avoid bypass of injecting agent policies and forgery of Path and Injection-Info headers, relaying agents SHOULD accept articles only from trusted agents.

An article SHOULD NOT be relayed unless the sending agent has been configured to supply, and the receiving agent to receive, at least one of the <newsgroup-name>s in its Newsgroups header field and at least one of the <dist-name>s in its Distribution header field (if present). Exceptionally, control messages creating or removing newsgroups (newgroup or rmgroup control messages, for example) SHOULD be relayed if the affected group appears in its Newsgroups header field and both the sending and receiving relaying agents are configured to relay a newsgroup of that name (whether or not such a newsgroup exists).

In order to avoid unnecessary relaying attempts, an article SHOULD NOT be relayed if the <path-identity> of the receiving agent (or some known alias thereof) appears as a <path-identity> (excluding within the <tail-entry> or following a "POSTED" <diag-keyword>) in its Path header field.

A relaying agent processes an article as follows:

1. It MUST reject any article without a Newsgroups header field or Message-ID header field, or without either an Injection-Date or Date header field.

2. It MUST examine the Injection-Date header field or, if absent, the Date header field, and reject the article if that date is more than 24 hours into the future. It MAY reject articles with dates in the future with a smaller margin than 24 hours.

3. It MUST reject any article that has already been accepted. If it implements one of the mechanisms described in Section 3.3, this means that it MUST reject any article whose date falls outside the cutoff interval since it won't know whether or not such articles had been accepted previously.

4. It SHOULD reject any article that does not include all the mandatory header fields. It MAY reject any article that contains header fields that do not have valid contents.

5. It SHOULD reject any article that matches an already-received cancel control message or the contents of the Supersedes header field of an accepted article, provided that the relaying agent has chosen (on the basis of local site policy) to honor that cancel control message or Supersedes header field.

6. It MAY reject any article without an Approved header field posted to a newsgroup known to be moderated. This practice is strongly encouraged, but the information necessary to do so is not required to be maintained by a relaying agent.

7. It MUST update the Path header field as described in Section 3.2.1.

8. It MAY delete any Xref header field already present. It MAY add a new Xref header field for its own use (but recall that [RFC5536] permits at most one such header field).

9. Finally, it passes the article on to other relaying and serving agents to which it is configured to send articles.

Relaying agents SHOULD, where possible in the underlying transport, inform the agent that passed the article to the relaying agent if the article was rejected. Relaying agents MUST NOT inform any other external entity of the rejection of an article unless that external entity has explicitly requested that it be informed of such errors.

Relaying agents MUST NOT alter, delete, or rearrange any part of an article except for the Path and Xref header fields. They MUST NOT modify the body of articles in any way. If an article is not acceptable as is, the article MUST be rejected rather than modified.

3.7. Duties of a Serving Agent

A serving agent accepts articles from a relaying agent or injecting agent, stores them, and makes them available to reading agents. Articles are normally indexed by newsgroup and <article-locator> (Section 3.2.14 of [RFC5536]), usually in the form of a decimal number.

If the serving agent stores articles by newsgroup, control messages SHOULD NOT be stored in the newsgroups listed in the control message's Newsgroups header field. Instead, they SHOULD be stored in a newsgroup in the hierarchy "control", which is reserved for this purpose. Conventionally, control messages are stored in newsgroups named for the type of control message (such as "control.cancel" for cancel control messages).

A serving agent MUST have available a list (possibly empty) of moderated groups for which it accepts articles so that it can reject unapproved articles posted to moderated groups. Frequently, a serving agent is deployed in combination with an injecting agent and can use the same list as the injecting agent.

A serving agent processes articles as follows:

1. It MUST reject any article that does not include all the mandatory header fields or any article that contains header fields that do not have valid contents.

2. It MUST examine the Injection-Date header field or, if absent, the Date header field, and reject the article if that date is more than 24 hours into the future. It MAY reject articles with dates in the future with a smaller margin than 24 hours.

3. It MUST reject any article that has already been accepted. If it implements one of the mechanisms described in Section 3.3, this means that it MUST reject any article whose date falls outside the cutoff interval since it won't know whether or not such articles had been accepted previously.

4. It SHOULD reject any article that matches an already-received and honored cancel message or Supersedes header field, following the same rules as a relaying agent (Section 3.6).

5. It MUST reject any article without an Approved header field posted to any newsgroup listed as moderated.

6. It MUST update the Path header field as described in Section 3.2.1.

7. It MUST remove any Xref header field from each article (except when specially configured to preserve the <article-locator>s set by the sending site). It then MAY (and usually will) add a new Xref header field (but recall that [RFC5536] permits at most one such header field).

8. Finally, it stores the article and makes it available for reading agents.

Serving agents MUST NOT create new newsgroups simply because an unrecognized <newsgroup-name> occurs in a Newsgroups header field. Newsgroups are normally created via control messages (Section 5.2.1).

Serving agents MUST NOT alter, delete, or rearrange any part of an article except for the Path and Xref header fields. They MUST NOT modify the body of the articles in any way. If an article is not acceptable as is, the article MUST be rejected rather than modified.

3.8. Duties of a Reading Agent

Since a reading agent is only a passive participant in a Netnews network, there are no specific protocol requirements placed on it. See [USEAGE] for best-practice recommendations.

3.9. Duties of a Moderator

A moderator receives news articles, customarily by email, decides whether to approve them and, if so, either passes them to an injecting agent or forwards them to further moderators.

Articles are normally received by the moderator in email, either encapsulated as an object of Content-Type application/ news-transmission (or possibly encapsulated but without an explicit Content-Type header field) or else directly as an email already containing all the header fields appropriate for a Netnews article (see Section 3.5.1). Moderators who may receive articles via email SHOULD be prepared to accept articles in either format.

There are no protocol restrictions on what criteria are used for accepting or rejecting messages or on what modifications a moderator may make to a message (both header fields and body) before injecting it. Recommended best practice, however, is to make the minimal required changes. Moderators need to be aware that modifications made to articles may invalidate signatures created by the poster or previous moderators. See [USEAGE] for further best-practice recommendations.

Moderators process articles as follows:

1. They decide whether to approve or reject a proto-article and, if approved, prepare the proto-article for injection. If the proto-article was received as an unencapsulated email message, this will require converting it back to a valid Netnews proto-article. If the article is rejected, it is normally rejected for all newsgroups to which it was posted and nothing further is done. If it is approved, the moderator proceeds with the following steps.

2. If the Newsgroups header field contains further moderated newsgroups for which approval has not already been given, they may either reach some agreement with the other moderators on the disposition of the article or, more generally, add an indication (identifying both the moderator and the name of the newsgroup) that they approve the article and then forward it to the moderator of the leftmost unapproved newsgroup. This forwarding SHOULD be done following the procedure in Section 3.5.1. It MAY be done by rotating the <newsgroup-name>s in the Newsgroups header field so that the leftmost unapproved newsgroup is the leftmost moderated newsgroup in that field and then posting it, letting the injecting agent do the forwarding. However, when using this mechanism, they MUST first ensure that the article contains no Approved header field.

3. If the Newsgroups header field contains no further unapproved moderated groups, they add an Approved header field (see Section 3.2.1 of [RFC5536]) identifying the moderator and, insofar as is possible, all the other moderators who have approved the article. The moderator who takes this step assumes responsibility for ensuring that the article was approved by the moderators of all moderated newsgroups to which it was posted.

4. Moderators are encouraged to retain the Message-ID header field unless it is invalid or the article was significantly changed from its original form. Moderators are also encouraged to retain the Date header field unless it appears to be stale (72 hours or more in the past) for reasons understood by the moderator (such as delays in the moderation process), in which case they MAY substitute the current date. Any Injection-Date, Injection-Info, or Xref header fields already present MUST be removed.

5. Any Path header field MUST either be removed or truncated to only those entries following its "POSTED" <diag-keyword>, if any.

6. The moderator then passes the article to an injecting agent, having first observed all the duties of a posting agent.

3.10. Duties of a Gateway

A gateway transforms an article into the native message format of another medium, or translates the messages of another medium into news articles, or transforms articles into proto-articles for injection into a separate Netnews network. Encapsulation of a news article into a message of MIME type application/news-transmission, or the subsequent undoing of that encapsulation, is not gatewaying since it involves no transformation of the article.

There are two basic types of gateway, the outgoing gateway that transforms a news article into a different type of message, and the incoming gateway that transforms a message from another network into a news proto-article and injects it into a Netnews network. These are handled separately below.

Transformation of an article into another medium stands a very high chance of discarding or interfering with the protection inherent in the news system against duplicate articles. The most common problem caused by gateways is loops that repeatedly reinject previously posted articles. To prevent this, a gateway MUST take precautions against loops, as detailed below.

The transformations applied to the message SHOULD be as minimal as possible while still accomplishing the gatewaying. Every change made by a gateway potentially breaks a property of one of the media or loses information, and therefore only those transformations made necessary by the differences between the media should be applied.

If bidirectional gatewaying (both an incoming and an outgoing gateway) is being set up between Netnews and some other medium, the incoming and outgoing gateways SHOULD be coordinated to avoid unintended reinjection of gated articles. Circular gatewaying (gatewaying a message into another medium and then back into Netnews) SHOULD NOT be done; encapsulation of the article SHOULD be used instead where this is necessary.

Safe bidirectional gatewaying between a mailing list and a newsgroup is far easier if the newsgroup is moderated. Posts to the moderated group and submissions to the mailing list can then go through a single point that does the necessary gatewaying and then sends the message out to both the newsgroup and the mailing list at the same time, eliminating most of the possibility of loops. Bidirectional gatewaying between a mailing list and an unmoderated newsgroup, in contrast, is difficult to do correctly and is far more fragile. Newsgroups intended to be bidirectionally gated to a mailing list SHOULD therefore be moderated where possible, even if the moderator

is a simple gateway and injecting agent that correctly handles crossposting to other moderated groups and otherwise passes all traffic.

3.10.1. Duties of an Outgoing Gateway

From the perspective of Netnews, an outgoing gateway is just a special type of reading agent. The exact nature of what the outgoing gateway will need to do to articles depends on the medium to which the articles are being gated. Because it raises the danger of loops due to the possibility of one or more corresponding incoming gateways back from that medium to Netnews, the operation of the outgoing gateway is subject to additional constraints.

The following practices are encouraged for all outgoing gateways, regardless of whether there is known to be a related incoming gateway, both as precautionary measures and as guidelines to quality of implementation:

1. The message identifier of the news article should be preserved if at all possible, preferably as or within the corresponding unique identifier of the other medium. However, if it is not preserved in this way, then at least it should be preserved as a comment in the message. This helps greatly with preventing loops.

2. The Date and Injection-Date of the news article should also be preserved if possible, for similar reasons.

3. The message should be tagged in some way so as to prevent its reinjection into Netnews. This may be impossible to do without knowledge of potential incoming gateways, but it is better to try to provide some indication even if not successful; at the least, a human-readable indication that the article should not be gated back to Netnews can help locate a human problem.

4. Netnews control messages should not be gated to another medium unless they would somehow be meaningful in that medium.

3.10.2. Duties of an Incoming Gateway

The incoming gateway has the responsibility of ensuring that all of the requirements of this protocol are met by the articles that it forms. In addition to its special duties as a gateway, it bears all of the duties and responsibilities of a posting agent, and it has the same responsibility of a relaying agent to reject articles that it has already gatewayed.

An incoming gateway MUST NOT gate the same message twice. It may not be possible to ensure this in the face of mangling or modification of the message, but at the very least a gateway, when given a copy of a message that it has already gated and that is identical except for trace header fields (like Received in Email or Path in Netnews), MUST NOT gate the message again. An incoming gateway SHOULD take precautions against having this rule bypassed by modifications of the message that can be anticipated.

News articles prepared by gateways MUST be valid news proto-articles (see Section 3.4.1). This often requires the gateway to synthesize a conforming article from non-conforming input. The gateway MUST then pass the article to an injecting agent, not directly to a relaying agent.

Incoming gateways MUST NOT pass control messages (articles containing a Control or Supersedes header field) without removing or renaming that header field. Gateways MAY, however, generate cancel control messages for messages they have gatewayed. If a gateway receives a message that it can determine is a valid equivalent of a cancel control message in the medium it is gatewaying, it SHOULD discard that message without gatewaying it, generate a corresponding cancel control message of its own, and inject that cancel control message.

> NOTE: It is not unheard of for mail-to-news gateways to be used to post control messages, but encapsulation should be used for these cases instead. Gateways by their very nature are particularly prone to loops. Spews of normal articles are bad enough; spews of control messages with special significance to the news system, possibly resulting in high processing load or even in emails being sent for every message received, are catastrophic. It is far preferable to construct a system specifically for posting control messages that can do appropriate consistency checks and authentication of the originator of the message.

If there is a message identifier that fills a role similar to that of the Message-ID header field in news, it SHOULD be used in the formation of the message identifier of the news article, perhaps with transformations required to meet the uniqueness requirement of Netnews and with the removal of any comments so as to comply with the syntax in Section 3.1.3 of [RFC5536]. Such transformations SHOULD be designed so that two messages with the same identifier generate the same Message-ID header field.

> NOTE: Message identifiers play a central role in the prevention of duplicates, and their correct use by gateways will do much to prevent loops. Netnews does, however, require that message identifiers be unique, and therefore message identifiers from

other media may not be suitable for use without modification. A balance must be struck by the gateway between preserving information used to prevent loops and generating unique message identifiers.

Exceptionally, if there are multiple incoming gateways for a particular set of messages, each to a different newsgroup(s), each one SHOULD generate a message identifier unique to that gateway. Each incoming gateway nonetheless MUST ensure that it does not gate the same message twice.

NOTE: Consider the example of two gateways of a given mailing list into two separate Usenet newsgroups, both of which preserve the email message identifier. Each newsgroup may then receive a portion of the messages (different sites seeing different portions). In these cases, where there is no one "official" gateway, some other method of generating message identifiers has to be used to avoid collisions. It would obviously be preferable for there to be only one gateway that crossposts, but this may not be possible to coordinate.

If no date information is available, the gateway MAY supply a Date header field with the gateway's current date. If only partial information is available (such as date but not time), this SHOULD be fleshed out to a full Date by adding default values rather than by discarding this information. Only in very exceptional circumstances should Date information be discarded, as it plays an important role in preventing reinjection of old messages.

An incoming gateway MUST add a Sender header field to the news article it forms by containing the <mailbox> of the administrator of the gateway. Problems with the gateway may be reported to this <mailbox>. The <display-name> portion of this <mailbox> SHOULD indicate that the entity responsible for injection of the message is a gateway. If the original message already had a Sender header field, it SHOULD be renamed to Original-Sender so that its contents can be preserved. See Section 3.10.3 for the specification of that header field.

3.10.3. Original-Sender Header Field

The Original-Sender header field holds the content of a Sender header field in an original message received by an incoming gateway, preserving it while the incoming gateway adds its own Sender header field. The content syntax makes use of syntax defined in [RFC5536] and [RFC5322].

```
header              =/ Original-Sender-header
Original-Sender-header
                    = "Original-Sender" ":" SP
                         Original-Sender-content
Original-Sender-content
                    = mailbox
```

The Permanent Message Header Field Repository entry for this header field is:

Header field name:	Original-Sender
Applicable protocol:	Netnews
Status:	standard
Author/Change controller:	IETF
Specification document(s):	RFC 5537

3.10.4. Gateway Example

To illustrate the type of precautions that should be taken against loops, here is an example of the measures taken by one particular combination of mail-to-news and news-to-mail gateways designed to handle bidirectional gatewaying between mailing lists and unmoderated groups:

1. The news-to-mail gateway preserves the message identifier of the news article in the generated email message. The mail-to-news gateway likewise preserves the email message identifier, provided that it is syntactically valid for Netnews. This allows the news system's built-in suppression of duplicates to serve as the first line of defense against loops.

2. The news-to-mail gateway adds an X-* header field to all messages it generates. The mail-to-news gateway discards any incoming messages containing this header field. This is robust against mailing list managers that replace the message identifier and against any number of email hops, provided that the other message header fields are preserved.

3. The mail-to-news gateway prepends the host name from which it received the email message to the content of the Path header field. The news-to-mail gateway refuses to gateway any message that contains the list server name in its Path header field (including in the tail section). This is robust against any amount of munging of the message header fields by the mailing list, provided that the email only goes through one hop.

4. The mail-to-news gateway is designed never to generate bounces to the envelope sender. Instead, articles that are rejected by the news server (for reasons not warranting silent discarding of the message) result in a bounce message sent to an errors address that is known not to forward to any mailing lists. In this way, they can be handled by the news administrators.

These precautions have proven effective in practice at preventing loops for this particular application (bidirectional gatewaying between mailing lists and locally distributed newsgroups where both gateways can be designed together). General gatewaying to world-wide newsgroups poses additional difficulties; one must be very wary of strange configurations, such as a newsgroup gated to a mailing list that is in turn gated to a different newsgroup.

4. Media Types

This document defines several media types, which have been registered with IANA as provided for in [RFC4288].

The media type message/news, as previously registered with IANA, is hereby declared obsolete. The intent of this media type was to define a standard way of transmitting news articles via mail for human reading. However, it was never widely implemented, and its default treatment as application/octet-stream by agents that did not recognize it was counter-productive. The media type message/rfc822 (defined in Section 5.2.1 of [RFC2046]) SHOULD be used in its place.

The updated MIME media type definition of message/news is:

MIME type name:	message
MIME subtype name:	news
Required parameters:	none
Optional parameters:	none
Encoding considerations:	same as message/rfc822
Security considerations:	News articles may constitute "control messages", which can have effects on a host's news system beyond just addition of information. Since control messages may occur in normal news flow, most hosts are suitably defended against undesired effects already, but transmission of news articles via mail may bypass

firewall-type defenses. Reading a news article transmitted by mail involves no hazards beyond those of mail, but submitting it to news software for processing should be done with care.

Interoperability considerations:
Rarely used, and therefore often handled as application/octet-stream. Disposition should by default be inline.

Published specification: RFC 5537

Applications that use this media type:
Some old mail and news user agents.

Intended usage: OBSOLETE

Author: Henry Spencer

Change controller: IETF

4.1. application/news-transmission

The media type application/news-transmission is intended for the encapsulation of complete news articles where the intention is that the recipient should then inject them into Netnews. This application type provides one of the methods for mailing articles to moderators (see Section 3.5.1) and may be used to convey messages to an injecting agent. This encapsulation removes the need to transform an email message into a Netnews proto-article and provides a way to send a Netnews article using MIME through a transport medium that does not support 8bit data.

The MIME media type definition of application/news-transmission is:

MIME type name: application

MIME subtype name: news-transmission

Required parameters: none

Optional parameters: One and only one of "usage=moderate", "usage=inject", or "usage=relay".

Encoding considerations: A transfer-encoding different from that of the article transmitted MAY be supplied to ensure correct transmission over some 7bit transport medium.

Security considerations: News articles may constitute "control messages", which can have effects on a host's news system beyond just addition of information. Since control messages may occur in normal news flow, most hosts are suitably defended against undesired effects already, but transmission of news articles via mail may bypass firewall-type defenses.

Published specification: RFC 5537

Body part: A complete proto-article ready for injection into Netnews or an article being relayed to another agent.

Applications that use this media type: Injecting agents, Netnews moderators.

Intended usage: COMMON

Change controller: IETF

usage=moderate indicates the article is intended for a moderator, usage=inject for an injecting agent, and usage=relay for a relaying agent. The entity receiving the article may only implement one type of agent, in which case the parameter MAY be omitted.

Contrary to the prior registration of this media type, article batches are not permitted as a body part. Multiple messages or a message with multiple application/news-transmission parts may be used instead.

4.2. application/news-groupinfo

The application/news-groupinfo media type is used in conjunction with the newgroup control message (see Section 5.2.1). Its body part contains brief information about a newsgroup: the newsgroup name, its description, and its moderation status.

The MIME media type definition of application/news-groupinfo is:

MIME type name: application

MIME subtype name: news-groupinfo

Required parameters: none

Optional parameters: charset, which MUST be a charset registered for use with MIME text types. It has the same syntax as the parameter defined for text/plain [RFC2046]. Specifies the charset of the body part. If not given, the charset defaults to US-ASCII [ASCII].

Encoding considerations: 7bit or 8bit encoding MUST be used to maintain compatibility.

Security considerations: None.

Interoperability considerations: Disposition should by default be inline.

Applications that use this media type: Control message issuers, relaying agents, serving agents.

Published specification: RFC 5537

Intended usage: COMMON

Change controller: IETF

The content of the application/news-groupinfo body part is defined as:

```
groupinfo-body      = [ newsgroups-tag CRLF ]
                        newsgroups-line CRLF
newsgroups-tag      = %x46.6F.72 SP %x79.6F.75.72 SP
                        %x6E.65.77.73.67.72.6F.75.70.73 SP
                        %x66.69.6C.65.3A
                        ; case sensitive
                        ; "For your newsgroups file:"
newsgroups-line     = newsgroup-name
                        [ 1*HTAB newsgroup-description ]
                        [ *WSP moderation-flag ]
newsgroup-description
                    = eightbit-utext *( *WSP eightbit-utext )
moderation-flag     = SP "(" %x4D.6F.64.65.72.61.74.65.64 ")"
```

```
                                ; SPACE + case sensitive "(Moderated)"
      eightbit-utext      = VCHAR / %d127-255
```

This unusual format is backward-compatible with the scanning of the body of newgroup control messages for descriptions done by Netnews implementations that predate this specification. Although optional, the <newsgroups-tag> SHOULD be included for backward compatibility.

The <newsgroup-description> MUST NOT contain any occurrence of the string "(Moderated)" within it. Moderated newsgroups MUST be marked by appending the case-sensitive text " (Moderated)" at the end.

While a charset parameter is defined for this media type, most existing software does not understand MIME header fields or correctly handle descriptions in a variety of charsets. Using a charset of US-ASCII where possible is therefore RECOMMENDED; if not possible, UTF-8 [RFC3629] SHOULD be used. Regardless of the charset used, the constraints of the above grammar MUST be met and the <newsgroup-name> MUST be represented in that charset using the same octets as would be used with a charset of US-ASCII.

4.3. application/news-checkgroups

The application/news-checkgroups media type contains a list of newsgroups within a hierarchy or hierarchies, including their descriptions and moderation status. It is primarily for use with the checkgroups control message (see Section 5.2.3).

The MIME media type definition of application/news-checkgroups is:

MIME type name:	application
MIME subtype name:	news-checkgroups
Required parameters:	none
Optional parameters:	charset, which MUST be a charset registered for use with MIME text types. It has the same syntax as the parameter defined for text/plain [RFC2046]. Specifies the charset of the body part. If not given, the charset defaults to US-ASCII [ASCII].
Encoding considerations:	7bit or 8bit encoding MUST be used to maintain compatibility.

Security considerations: This media type provides only a means for conveying a list of newsgroups and does not provide any information indicating whether the sender is authorized to state which newsgroups should exist within a hierarchy. Such authorization must be accomplished by other means.

Interoperability considerations: Disposition should by default be inline.

Applications that use this media type: Control message issuers, relaying agents, serving agents.

Published specification: RFC 5537

Intended usage: COMMON

Change controller: IETF

The content of the application/news-checkgroups body part is defined as:

```
checkgroups-body    = *( valid-group CRLF )
valid-group         = newsgroups-line ; see Section 4.2
```

The same restrictions on charset, <newsgroup-name>, and <newsgroup-description> apply for this media type as for application/news-groupinfo.

One application/news-checkgroups message may contain information for one or more hierarchies and is considered complete for any hierarchy for which it contains a <valid-group> unless accompanied by external information limiting its scope (such as a <chkscope> parameter to a checkgroups control message, as described in Section 5.2.3). In other words, an application/news-checkgroups body part consisting of

```
example.moderated         A moderated newsgroup (Moderated)
example.test              An unmoderated test group
```

is a statement that the example.* hierarchy contains two newsgroups, example.moderated and example.test, and no others. This media type therefore MUST NOT be used for conveying partial information about a hierarchy; if a group from a given hierarchy is present, all groups

that exist in that hierarchy MUST be listed unless its scope is limited by external information, in which case all groups SHOULD be listed.

Spaces are used in this example for formatting reasons. In an actual message, the newsgroup name and description MUST be separated by one or more tabs (HTAB, ASCII %d09), not spaces.

5. Control Messages

A control message is an article that contains a Control header field and thereby indicates that some action should be taken by an agent other than distribution and display. Any article containing a Control header field (defined in Section 3.2.3 of [RFC5536]) is a control message. Additionally, the action of an article containing a Supersedes header field is described here; while such an article is not a control message, it specifies an action similar to the cancel control message.

The <control-command> of a Control header field comprises a <verb>, which indicates the action to be taken, and one or more <argument> values, which supply the details. For some control messages, the body of the article is also significant. Each recognized <verb> (the control message type) is described in a separate section below. Agents MAY accept other control message types than those specified below, and MUST either ignore or reject control messages with unrecognized types. Syntactic definitions of valid <argument> values and restrictions on control message bodies are given in the section for each control message type.

Contrary to [RFC1036], the presence of a Subject header field starting with the string "cmsg " MUST NOT cause an article to be interpreted as a control message. Agents MAY reject an article that has such a Subject header field and no Control header field as ambiguous. Likewise, the presence of a <newsgroup-name> ending in ".ctl" in the Newsgroups header field or the presence of an Also-Control header field MUST NOT cause the article to be interpreted as a control message.

5.1. Authentication and Authorization

Control messages specify actions above and beyond the normal processing of an article and are therefore potential vectors of abuse and unauthorized action. There is, at present, no standardized means of authenticating the sender of a control message or verifying that the contents of a control message were sent by the claimed sender. There are, however, some unstandardized authentication mechanisms in common use, such as [PGPVERIFY].

Agents acting on control messages SHOULD take steps to authenticate control messages before acting on them, as determined by local authorization policy. Whether this is done via the use of an unstandardized authentication protocol, by comparison with information obtained through another protocol, by human review, or by some other means is left unspecified by this document. Further extensions or revisions of this protocol are expected to standardize a digital signature mechanism.

Agents are expected to have their own local authorization policies for which control messages will be honored. No Netnews agent is ever required to act on any control message. The following descriptions specify the actions that a control message requests, but an agent MAY always decline to act on any given control message.

5.2. Group Control Messages

A group control message is any control message type that requests some update to the list of newsgroups known to a news server. The standard group control message types are "newgroup", "rmgroup", and "checkgroups".

Before honoring any group control message, an agent MUST check the newsgroup or newsgroups affected by that control message and decline to create any newsgroups not in conformance with the restrictions in Section 3.1.4 of [RFC5536].

All of the group control messages MUST have an Approved header field (Section 3.2.1 of [RFC5536]). Group control messages without an Approved header field SHOULD NOT be honored.

Group control messages affecting specific groups (newgroup and rmgroup control messages, for example) SHOULD include the <newsgroup-name> for the group or groups affected in their Newsgroups header field. Other newsgroups MAY be included in the Newsgroups header field so that the control message will reach more news servers, but due to the special relaying rules for group control messages (see Section 3.6) this is normally unnecessary and may be excessive.

5.2.1. The newgroup Control Message

The newgroup control message requests that the specified group be created or, if already existing, that its moderation status or description be changed. The syntax of its Control header field is:

```
control-command     =/ Newgroup-command
Newgroup-command    = "newgroup" Newgroup-arguments
Newgroup-arguments  = 1*WSP newsgroup-name
```

```
                                [ 1*WSP newgroup-flag ]
      newgroup-flag       = "moderated"
```

If the request is honored, the moderation status of the group SHOULD be set in accordance with the presence or absence of the <newgroup-flag> "moderated". "moderated" is the only flag defined by this protocol. Other flags MAY be defined by extensions to this protocol and accepted by agents. If an agent does not recognize the <newgroup-flag> of a newgroup control message, it SHOULD ignore that control message.

The body of a newgroup message SHOULD contain an entity of type application/news-groupinfo specifying the description of the newsgroup, either as the entire body or as an entity within a multipart/mixed object [RFC2046]. If such an entity is present, the moderation status specified therein MUST match the moderation status specified by the <newgroup-flag>. The body of a newgroup message MAY contain other entities (encapsulated in multipart/mixed) that provide additional information about the newsgroup or the circumstances of the control message.

In the absence of an application/news-groupinfo entity, a news server MAY search the body of the message for the line "For your newsgroups file:" and take the following line as a <newsgroups-line>. Prior to the standardization of application/news-groupinfo, this was the convention for providing a newsgroup description.

If the request is honored and contains a newsgroup description, and if the news server honoring it stores newsgroup descriptions, the stored newsgroup description SHOULD be updated to the description specified in the control message, even if no other property of the group has changed.

5.2.1.1. newgroup Control Message Example

A newgroup control message requesting creation of the moderated newsgroup example.admin.info.

```
From: "example.* Administrator" <admin@noc.example>
Newsgroups: example.admin.info
Date: 27 Feb 2002 12:50:22 +0200
Subject: cmsg newgroup example.admin.info moderated
Approved: admin@noc.example
Control: newgroup example.admin.info moderated
Message-ID: <ng-example.admin.info-20020227@noc.example>
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="nxtprt"
Content-Transfer-Encoding: 8bit
```

```
This is a MIME control message.
--nxtprt
Content-Type: application/news-groupinfo; charset=us-ascii

For your newsgroups file:
example.admin.info      About the example.* groups (Moderated)

--nxtprt
Content-Type: text/plain; charset=us-ascii

A moderated newsgroup for announcements about new newsgroups in
the example.* hierarchy.

--nxtprt--
```

Spaces are used in this example for formatting reasons. In an actual message, the newsgroup name and description MUST be separated by one or more tabs (HTAB, ASCII %d09), not spaces.

5.2.2. The rmgroup Control Message

The rmgroup control message requests that the specified group be removed from a news server's list of valid groups. The syntax of its Control header field is:

```
control-command     =/ Rmgroup-command
Rmgroup-command     = "rmgroup" Rmgroup-arguments
Rmgroup-arguments   = 1*WSP newsgroup-name
```

The body of the control message MAY contain anything, usually an explanatory text.

5.2.3. The checkgroups Control Message

The checkgroups control message contains a list of all the valid groups in a hierarchy with descriptions and moderation status. It requests that a news server update its valid newsgroup list for that hierarchy to include the groups specified, remove any groups not specified, and update group descriptions and moderation status to match those given in the checkgroups control message. The syntax of its Control header field is:

```
control-command     =/ Checkgroup-command
Checkgroup-command  = "checkgroups" Checkgroup-arguments
Checkgroup-arguments= [ chkscope ] [ chksernr ]
chkscope            = 1*( 1*WSP ["!"] newsgroup-name )
chksernr            = 1*WSP "#" 1*DIGIT
```

A checkgroups message is interpreted as an exhaustive list of the valid groups in all hierarchies or sub-hierarchies with a prefix listed in the <chkscope> argument, excluding any sub-hierarchy where the <chkscope> argument is prefixed by "!". For complex cases with multiple <chkscope> arguments, start from an empty list of groups, include all groups in the checkgroups control message matching <chkscope> arguments without a "!" prefix, and then exclude all groups matching <chkscope> arguments with a "!" prefix. Follow this method regardless of the order of the <chkscope> arguments in the Control header field.

If no <chkscope> argument is given, it applies to all hierarchies for which group statements appear in the body of the message.

Since much existing software does not honor the <chkscope> argument, the body of the checkgroups control message MUST NOT contain group statements for newsgroups outside the intended scope and SHOULD contain a correct newsgroup list even for sub-hierarchies excluded with "!" <chkscope> terms. News servers, however, MUST honor <chkscope> as specified here.

The <chksernr> argument may be any positive integer. If present, it MUST increase with every change to the newsgroup list, MUST NOT ever decrease, and MUST be included in all subsequent checkgroups control messages with the same scope. If provided, news servers SHOULD remember the <chksernr> value of the previous checkgroups control message honored for a particular hierarchy or sub-hierarchy and decline to honor any subsequent checkgroups control message for the same hierarchy or sub-hierarchy with a smaller <chksernr> value or with no <chksernr> value.

There is no upper limit on the length of <chksernr>, other than the limitation on the length of header fields. Implementations may therefore want to do comparisons by zero-padding the shorter of two <chksernr> values on the left and then doing a string comparison, rather than assuming <chksernr> can be manipulated as a number.

For example, the following Control header field

```
Control: checkgroups de !de.alt #2009021301
```

indicates that the body of the message will list every newsgroup in the de.* hierarchy, excepting the de.alt.* sub-hierarchy, and should not be honored if a checkgroups control message with a serial number greater than 2009021301 was previously honored. The serial number in this example was formed from the date in YYYYMMDD format, followed by a two-digit sequence number within that date.

The body of the message is an entity of type application/ news-checkgroups. It SHOULD be declared as such with appropriate MIME headers, but news servers SHOULD interpret checkgroups messages that lack the appropriate MIME headers as if the body were of type application/news-checkgroups for backward compatibility.

5.3. The cancel Control Message

The cancel control message requests that a target article be withdrawn from circulation and access. The syntax of its Control header field is:

```
control-command     =/ Cancel-command
Cancel-command      = "cancel" Cancel-arguments
Cancel-arguments    = 1*WSP msg-id
```

The argument identifies the article to be cancelled by its message identifier. The body of the control message MAY contain anything, usually an explanatory text.

A serving agent that elects to honor a cancel message SHOULD make the article unavailable to reading agents (perhaps by deleting it completely). If the cancel control message arrives before the article it targets, news servers choosing to honor it SHOULD remember the message identifier that was cancelled and reject the cancelled article when it arrives.

To best ensure that it will be relayed to the same news servers as the original message, a cancel control message SHOULD have the same Newsgroups header field as the message it is cancelling.

Cancel control messages listing moderated newsgroups in their Newsgroups header field MUST contain an Approved header field like any other article in a moderated newsgroup. This means that cancels posted to a moderated newsgroup will normally be sent to the moderator first for approval. Outside of moderated newsgroups, cancel messages are not required to contain an Approved header field.

Contrary to [RFC1036], cancel control messages are not required to contain From and Sender header fields matching the target message. This requirement only encouraged cancel issuers to conceal their identity and provided no security.

5.4. The Supersedes Header Field

The presence of a Supersedes header field in an article requests that the message identifier given in that header field be withdrawn in exactly the same manner as if it were the target of a cancel control

message. Accordingly, news servers SHOULD apply to a Supersedes header field the same authentication and authorization checks as they would apply to cancel control messages. If the Supersedes header field is honored, the news server SHOULD take the same actions as it would take when honoring a cancel control message for the given target article.

The article containing the Supersedes header field, whether or not the Supersedes header field is honored, SHOULD be handled as a normal article and SHOULD NOT receive the special treatment of control messages described in Section 3.7.

5.5. The ihave and sendme Control Messages

The ihave and sendme control messages implement a predecessor of the NNTP [RFC3977] protocol. They are largely obsolete on the Internet but still see use in conjunction with some transport protocols such as UUCP [UUCP]. News servers are not required to support them.

ihave and sendme control messages share similar syntax for their Control header fields and bodies:

```
control-command     =/ Ihave-command
Ihave-command       = "ihave" Ihave-arguments
Ihave-arguments     = 1*WSP *( msg-id 1*WSP ) relayer-name
control-command     =/ Sendme-command
Sendme-command      = "sendme" Sendme-arguments
Sendme-arguments    = Ihave-arguments
relayer-name        = path-identity  ; see 3.1.5 of [RFC5536]
ihave-body          = *( msg-id CRLF )
sendme-body         = ihave-body
```

The body of the article consists of a list of <msg-id>s, one per line. The message identifiers SHOULD be put in the body of the article, not in the Control header field, but news servers MAY recognize and process message identifiers in the Control header field for backward compatibility. Message identifiers MUST NOT be put in the Control header field if they are present in the body of the control message.

The ihave message states that the named relaying agent has received articles with the specified message identifiers, which may be of interest to the relaying agents receiving the ihave message. The sendme message requests that the agent receiving it send the articles having the specified message identifiers to the named relaying agent. Contrary to [RFC1036], the relayer-name MUST be given as the last argument in the Control header field.

Upon receipt of the sendme message (and a decision to honor it), the receiving agent sends the article or articles requested. The mechanism for this transmission is unspecified by this document and is arranged between the sites involved.

These control messages are normally sent as point-to-point articles between two sites and not then sent on to other sites. Newsgroups beginning with "to." are reserved for such point-to-point communications and are formed by prepending "to." to a <relayer-name> to form a <newsgroup-name>. Articles with such a group in their Newsgroups header fields SHOULD NOT be sent to any news server other than the one identified by <relayer-name>.

5.6. Obsolete Control Messages

The following control message types are declared obsolete by this document and SHOULD NOT be sent or honored:

sendsys
version
whogets
senduuname

6. Security Considerations

Netnews is designed for broad dissemination of public messages and offers little in the way of security. What protection Netnews has against abuse and impersonation is provided primarily by the underlying transport layer. In large Netnews networks where news servers cannot be relied upon to enforce authentication and authorization requirements at the transport layer, articles may be trivially forged and widely read, and the identities of article senders and the privacy of articles cannot be assured.

See Section 5 of [RFC5536] for further security considerations related to the format of articles.

6.1. Compromise of System Integrity

Control messages pose a particular security concern since acting on unauthorized control messages may cause newsgroups to be removed, articles to be deleted, and unwanted newsgroups to be created. Administrators of news servers SHOULD therefore take steps to verify the authenticity of control messages as discussed in Section 5.1. Articles containing Supersedes header fields are effectively cancel control messages and SHOULD be subject to the same checks as

discussed in Section 5.4. Currently, many sites are ignoring all cancel control messages and Supersedes header fields due to the difficulty of authenticating them and their widespread abuse.

Cancel control messages are not required to have the same Newsgroups header field as the messages they are cancelling. Since they are sometimes processed before the original message is received, it may not be possible to check that the Newsgroup header fields match. This allows a malicious poster to inject a cancel control message for an article in a moderated newsgroup without adding an Approved header field to the control message, and to hide malicious cancel control messages from some reading agents by using a different Newsgroups header field so that the cancel control message is not accepted by all news servers that accepted the original message.

All agents should be aware that all article content, most notably including the content of the Control header field, is potentially untrustworthy and malicious. Articles may be constructed in syntactically invalid ways to attempt to overflow internal buffers, violate hidden assumptions, or exploit implementation weaknesses. For example, some news server implementations have been successfully attacked via inclusion of Unix shell code in the Control header field. All article contents, and particularly control message contents, SHOULD be handled with care and rigorously verified before any action is taken on the basis of the contents of the article.

A malicious poster may add an Approved header field to bypass the moderation process of a moderated newsgroup. Injecting agents SHOULD verify that messages approved for a moderated newsgroup are being injected by the moderator using authentication information from the underlying transport or some other authentication mechanism arranged with the moderator. There is, at present, no standardized method of authenticating approval of messages to moderated groups, although some unstandardized authentication methods such as [PGPMOOSE] are in common use.

A malicious news server participating in a Netnews network may bypass checks performed by injecting agents, forge Path header fields and other trace information (such as Injection-Info header fields), and otherwise compromise the authorization requirements of a Netnews network. News servers SHOULD use the facilities of the underlying transport to authenticate their peers and reject articles from injecting and relaying agents that do not follow the requirements of this protocol or the Netnews network.

6.2. Denial of Service

The proper functioning of individual newsgroups can be disrupted by the excessive posting of unwanted articles; by the repeated posting of identical or near identical articles; by posting followups that either are unrelated to their precursors or that quote their precursors in full with the addition of minimal extra material (especially if this process is iterated); by crossposting to, or requesting followups to, totally unrelated newsgroups; and by abusing control messages and the Supersedes header field to delete articles or newsgroups.

Such articles intended to deny service, or other articles of an inflammatory nature, may also have their From or Reply-To addresses set to valid but incorrect email addresses, thus causing large volumes of email to descend on the true owners of those addresses. Users and agents should always be aware that identifying information in articles may be forged.

A malicious poster may prevent an article from being seen at a particular site by including in the Path header field of the proto-article the <path-identity> of that site. Use of the <diag-keyword> "POSTED" by injecting agents to mark the point of injection can prevent this attack.

Primary responsibility for preventing such attacks lies with injecting agents, which can apply authentication and authorization checks via the underlying transport and prevent those attempting denial-of-service attacks from posting messages. If specific injecting agents fail to live up to their responsibilities, they may be excluded from the Netnews network by configuring relaying agents to reject articles originating from them.

A malicious complainer may submit a modified copy of an article (with an altered Injection-Info header field, for instance) to the administrator of an injecting agent in an attempt to discredit the author of that article and even to have his posting privileges removed. Administrators SHOULD therefore obtain a genuine copy of the article from their own serving agent before taking action in response to such a complaint.

6.3. Leakage

Articles that are intended to have restricted distribution are dependent on the goodwill of every site receiving them. Restrictions on dissemination and retention of articles may be requested via the Archive and Distribution header fields, but such requests cannot be enforced by the protocol.

The flooding algorithm used by Netnews transports such as NNTP [RFC3977] is extremely good at finding any path by which articles can leave a subnet with supposedly restrictive boundaries, and substantial administrative effort is required to avoid this. Organizations wishing to control such leakage are advised to designate a small number of gateways to handle all news exchanges with the outside world.

The sendme control message (Section 5.5), insofar as it is still used, can be used to request articles that the requester should not have access to.

7. IANA Considerations

IANA has registered the following media types, described elsewhere in this document, for use with the Content-Type header field, in the IETF tree in accordance with the procedures set out in [RFC4288].

```
application/news-transmission  (4.1)
application/news-groupinfo     (4.2)
application/news-checkgroups   (4.3)
```

application/news-transmission is a change to a previous registration.

IANA has registered the new header field, Original-Sender, in the Permanent Message Header Field Repository, using the template in Section 3.10.3.

IANA has changed the status of the message/news media type to "OBSOLETE". message/rfc822 should be used instead. An updated template is included in Section 4.

8. References

8.1. Normative References

[ASCII] American National Standard for Information Systems, "Coded Character Sets - 7-Bit American National Standard Code for Information Interchange (7-Bit ASCII)", ANSI X3.4, 1986.

[RFC2046] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, November 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

[RFC4288] Freed, N. and J. Klensin, "Media Type Specifications and Registration Procedures", BCP 13, RFC 4288, December 2005.

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322, October 2008.

[RFC5536] Murchison, K., Ed., Lindsey, C., and D. Kohn, "Netnews Article Format", RFC 5536, November 2009.

8.2. Informative References

[PGPMOOSE] Rose, G., "PGP Moose", November 1998.

[PGPVERIFY] Lawrence, D., "Signing Control Messages", August 2001, <ftp://ftp.isc.org/pub/pgpcontrol/FORMAT>.

[RFC1036] Horton, M. and R. Adams, "Standard for interchange of USENET messages", RFC 1036, December 1987.

[RFC2045] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996.

[RFC2606] Eastlake, D. and A. Panitz, "Reserved Top Level DNS Names", BCP 32, RFC 2606, June 1999.

[RFC3798] Hansen, T. and G. Vaudreuil, "Message Disposition Notification", RFC 3798, May 2004.

[RFC3977] Feather, C., "Network News Transfer Protocol (NNTP)", RFC 3977, October 2006.

[SON-OF-1036] Spencer, H., "News Article Format and Transmission", Work in Progress, May 2009.

[USEAGE] Lindsey, C., "Usenet Best Practice", Work in Progress, March 2005.

[UUCP] O'Reilly, T. and G. Todino, "Managing UUCP and Usenet", O'Reilly & Associates Ltd., January 1992.

Appendix A. Changes to the Existing Protocols

This document prescribes many changes, clarifications, and new features since the protocol described in [RFC1036]. Most notably:

- A new, backward-compatible Path header field format that permits standardized embedding of additional trace and authentication information is now RECOMMENDED. See Section 3.2. Folding of the Path header is permitted.

- Trimming of the References header field is REQUIRED, and a mechanism for doing so is defined.

- Addition of the new Injection-Date header field is required in some circumstances for posting agents (Section 3.4.2) and injecting agents (Section 3.5), and MUST be used by news servers for date checks (Section 3.6). Injecting agents are also strongly encouraged to put all local trace information in the new Injection-Info header field.

- A new media type is defined for transmitting Netnews articles through other media (Section 4.1), and moderators SHOULD prepare to receive submissions in that format (Section 3.5.1).

- Certain control messages (Section 5.6) are declared obsolete, and the special significance of "cmsg" at the start of a Subject header field is removed.

- Additional media types are defined for improved structuring, specification, and automated processing of control messages (Sections 4.2 and 4.3).

- Two new optional parameters are added to the checkgroups control message.

- The message/news media type is declared obsolete.

- Cancel control messages are no longer required to have From and Sender header fields matching those of the target message, as this requirement added no real security.

- The relayer-name parameter in the Control header field of ihave and sendme control messages is now required.

In addition, many protocol steps and article verification requirements that are unmentioned or left ambiguous by [RFC1036] but are widely implemented by Netnews servers have been standardized and specified in detail.

Appendix B. Acknowledgements

This document is the result of a twelve-year effort and the number of people that have contributed to its content are too numerous to mention individually. Many thanks go out to all past and present members of the USEFOR Working Group of the Internet Engineering Task Force (IETF) and the accompanying mailing list.

Special thanks are due to Henry Spencer, whose [SON-OF-1036] draft served as the initial basis for this document.

Authors' Addresses

Russ Allbery (editor)
Stanford University
P.O. Box 20066
Stanford, CA 94309
US

EMail: rra@stanford.edu
URI: http://www.eyrie.org/~eagle/

Charles H. Lindsey
5 Clerewood Avenue
Heald Green
Cheadle
Cheshire SK8 3JU
United Kingdom

Phone: +44 161 436 6131
EMail: chl@clerew.man.ac.uk