Classifying Response Codes to Support Multiple Path Routing in the Session Initiation Protocol (SIP)
draft-worley-redundancy-response-01

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on December 23, 2007.

Copyright Notice

Copyright (C) The IETF Trust (2007).
Abstract

An increasing number of SIP architectures implement multiple path routing (MPR), which is the providing of more than one path for a call to reach a destination user agent (UA). To implement MPR well, the proxy which forks a request onto the redundant paths needs to be able to determine if a fork that failed reached the destination UA and was rejected by the UA (and so an alternate path should not be tried), or if the fork failed before reaching the UA (and so an alternate path should be attempted). In order that the proxy can distinguish these two situations, response codes should unambiguously identify which of these situations applies. This Internet-Draft begins classifying the current usage of response codes, describes their implications for MPR, and suggests possible improvements.

Table of Contents

1. Background .................................................. 4
2. What Is a Final Response? ................................. 6
3. Response Codes ................................................ 7
   3.1. 400 Bad Request .................................. 7
   3.2. 401 Unauthorized .................................. 7
   3.3. 402 Payment Required ............................. 7
   3.4. 403 Forbidden ..................................... 7
   3.5. 404 Not Found ..................................... 7
   3.6. 405 Method Not Allowed ............................ 8
   3.7. 406 Not Acceptable ................................. 8
   3.8. 407 Proxy Authentication Required ................. 8
   3.9. 408 Request Timeout ............................... 8
   3.10. 410 Gone .......................................... 8
   3.11. 412 Conditional Request Failed ................... 8
   3.12. 413 Request Entity Too Large .................... 9
   3.13. 414 Request-URI Too Long ......................... 9
   3.14. 415 Unsupported Media Type ....................... 9
   3.15. 416 Unsupported URI Scheme ....................... 9
   3.16. 417 Unknown Resource-Priority .................... 9
   3.17. 420 Extension Not Supported ....................... 9
   3.18. 421 Extension Required ........................... 9
   3.19. 422 Session Interval Too Small ................... 10
   3.20. 423 Interval Too Brief ........................... 10
   3.21. 428 Use Identity Header .......................... 10
   3.22. 429 Provide Referrer Identity ..................... 10
   3.23. 436 Bad Identity-Info ............................ 11
   3.24. 437 Unsupported Certificate ....................... 11
   3.25. 438 Invalid Identity Header ....................... 11
   3.26. 480 Temporarily Unavailable ...................... 11
   3.27. 481 Call/Transaction Does Not Exist .............. 11
3.28. 482 Loop Detected ........................................ 11
3.29. 483 Too Many Hops ........................................ 11
3.30. 484 Address Incomplete .................................... 12
3.31. 485 Ambiguous ............................................. 12
3.32. 486 Busy Here ............................................. 12
3.33. 487 Request Terminated .................................... 12
3.34. 488 Not Acceptable Here ................................... 12
3.35. 489 Bad Event .............................................. 13
3.36. 491 Request Pending ....................................... 13
3.37. 493 Undecipherable ......................................... 13
3.38. 494 Security Agreement Required .......................... 13
3.39. 500 Server Internal Error .................................. 13
3.40. 501 Not Implemented ........................................ 14
3.41. 502 Bad Gateway ............................................ 14
3.42. 503 Service Unavailable .................................... 14
3.43. 504 Server Time-out ......................................... 14
3.44. 505 Version Not Supported .................................. 14
3.45. 513 Message Too Large ...................................... 14
3.46. 580 Precondition Failure .................................... 14
3.47. 6xx .......................................................... 14
4.  Ambiguous Response Codes ..................................... 16
5.  Proposal ........................................................ 17
  5.1. 420 Extension Not Supported By UA ........................ 17
  5.2. 424 Extension Not Supported By Proxy ...................... 17
  5.3. 480 Temporarily Unavailable At UA ........................ 17
  5.4. 495 Temporarily Unavailable At Proxy ...................... 17
  5.5. 487 Request Canceled ....................................... 17
  5.6. 496 Ring-No-Answer ......................................... 17
  5.7. 500 Server Internal Error .................................. 17
  5.8. 506 Service Unavailable Downstream ....................... 17
6.  Security Considerations ....................................... 18
7.  Revision History ............................................... 19
  7.1. draft-worley-sip-redundancy-response-00 .................. 19
  7.2. Changes from draft-worley-sip-redundancy-response-00 to draft-worley-sip-redundancy-response-01 ............... 19
8.  Normative References ......................................... 20
Author’s Address .................................................. 21
Intellectual Property and Copyright Statements ................... 22
1. Background

An increasing number of Session Initiation Protocol (SIP) [1] system architectures implement multiple path routing (MPR), the feature of providing more than one path for a call to reach a destination user agent (UA). (MPR is also called path redundancy (PR) or alternate path retry (APR).) Typical situations are:

- multiple gateway devices that connect a SIP network to the PSTN, such that if one gateway is occupied to capacity, calls should be routed to the next gateway
- a PSTN gateway device as fallback when SIP connection over the Internet fails
- sending a request to multiple services that determine how to reach a destination, with an order of precedence as to which service is to be used (e.g., connecting to an ENUM contact, falling back to a PSTN gateway)

From a protocol point of view, a proxy is presented with the situation where a request (typically an INVITE) should be serially forked to more than one SIP URI, and from an application-layer point of view, all of the URIs are expected to ultimately reach the same user agent (UA). (Here we assume the proxy knows that this is the situation. How the proxy does so is beyond the scope of this document. Also beyond the scope of this document is the question of when two destinations are considered "the same".) Of course, if one fork of the request succeeds, the proxy should not attempt any further forks. But if a request fails, in order to know whether to attempt any further forks, the proxy needs to know if the request succeeded in reaching the destination UA or not.

If the request did not reach the destination UA, then in order implement MPR, the proxy must attempt the next fork. But if the request did reach the destination UA, and the UA returned a failure response, sending the request to the same UA via a different path is unlikely to yield success, and may even degrade the user experience. For example, if the first request was not accepted by the attending human (ring-no-answer), sending a second request to the UA by a different path, which will cause the UA to alert again, is not the desired behavior.

Ideally, in order that the proxy can distinguish these cases, each final response code should identify whether the request was rejected by the UA or an intermediate agent, or equivalently, whether the request reached the destination UA or not. (There are also some special cases when a failure response is returned by an intermediate
agent, but MPR should not be attempted.) This Internet-Draft begins classifying the current usage of response codes and their implications for MPR, and begins suggesting improvements to the system of response codes.

Initially, this document is intended as a statement of current practice and a discussion of the implications of supporting MPR. It should evolve into a statement of best common practice. If problems with the existing semantics of responses are discovered, it may evolve into a draft standard that updates the semantics of response codes.
2. What Is a Final Response?

First, we need to clarify what is meant by a "final response" for MPR purposes. For implementing MPR, a final response is one which is the result of the proxy's actions in attempting to reach a particular target URI in the set of redundant targets. A final response is always the "best response" chosen from the set of responses from the set forks generated while attempting to reach one of the target URI. It is this response that is examined to determine whether to attempt the next target URI of the set of redundant URIs.

(We note that the proxy might have the set of redundant targets contained as a sub-set within a set of non-redundant targets for the incoming request-URI. That case is harder to keep straight in one's mind, but it is handled in the straightforward way: The proxy proceeds through the sub-sets using the MPR rules, but when moving from one sub-set to the next, it terminates when it receives a success response and continues when it receives a failure response.)

Redirection (3xx) responses are final if they are considered for being returned as the "best response" to an incoming request, that is, if the proxy will not act on the 3xx response to generate further forks whose responses are to be put in the response set. But if the proxy will act on the 3xx to generate new forks in place of the original target URI, the 3xx is not put into the response set and will not become a final response -- Rather, the final responses of the new forks are added to the response set to produce the final response for the target URI. Whether the proxy chooses to act on a 3xx response ("recurse on a redirect response") or to consider it final for its fork is a matter of local policy. (But implementors should be aware of the requirement of section 16.5 of [1] to not recurse on a 3xx response unless the incoming request-URI was for a domain that the proxy "is responsible for".)

A proxy is sometimes permitted to act on a 416 (Unsupported URI Scheme) response by rewriting the request-URI into a sip: or sips: URI and acting on that new URI. (See section 16.7 of [1].) In such as case, the 416 is treated in the same way as a 3xx. But if the 416 does not trigger rewriting, it is added to the response set and may become the final response.
3. Response Codes

The following enumeration of response codes was taken from the IANA registry (http://www.iana.org/assignments/sip-parameters) on 31 Jan 2007.

3.1. 400 Bad Request

The 400 response is used by an agent to report that a request is malformed. This indicates positive knowledge on the part of the agent that the request is malformed (and not that the agent has detected an an extension that it cannot understand). Hence, MPR should not be attempted on the request. (This is an example of a failure that suppresses MPR, but is not from a destination UA.)

3.2. 401 Unauthorized

The 401 response is used by a UA to report that the request is not adequately authorized to be processed, but if the UAC provides authorization, the request might be processable. Since an MPR fork will presumably be met with the same authorization requirement, MPR should not be attempted. Instead, the 401 should be returned to the UAC, which should retry the request with authorization.

3.3. 402 Payment Required

The 402 response is reserved for further standardization. However, in regard to MPR, it is likely that it will prove useful to separate this response into two response codes, "UAS Payment Required" and "Transport Payment Required". However, dealing with this can be postponed to such time as 402 is standardized.

3.4. 403 Forbidden

The 403 response is used by a UAS to indicate that it will not carry out the requested operation, even if provided with credentials. MPR should not be attempted.

3.5. 404 Not Found

The 404 response is used to indicate that an agent had positive knowledge that the request-URI presented to it does not exist. However, since the UAC has no way to know what request-URI was presented, this assertion of positive knowledge has limited usefulness. Because the 404 response is relative to the routing of a particular fork of the request, MPR should be attempted.
3.6. 405 Method Not Allowed

Since all well-behaved proxies do not filter requests based on method, 405 responses should only be generated by destination UAs. Hence, MPR should not be attempted for 405 responses.

3.7. 406 Not Acceptable

The 406 response is used by UAs to reject a request because they cannot service any of the media formats listed in the Accept header. 406 responses should only be generated by destination UAs. Hence, MPR should not be attempted for 406 responses.

3.8. 407 Proxy Authentication Required

The 407 response is like the 401 response, but it is generated by proxies and gateways to demand authentication before allowing access to a transport resource. Since the 407 response is relative to the transport path, MPR should be attempted for 407 responses.

3.9. 408 Request Timeout

The 408 response is used by a SIP agent to report or describe that its attempt to contact the next agent downstream failed with no response. Since the 408 response is always a transport error, MPR should be attempted for 408 responses.

Note that 408 is not used by a UAS to report a ring-no-answer condition. The limitation of ringing time is described to the UAS by an Expires header in the INVITE, but the termination of the INVITE is done by an upstream agent sending a CANCEL. See [1] section 13.2.1. So the response generated by ring-no-answer is 487. (See Section 3.33.)

3.10. 410 Gone

The 410 response, like the 404 response, is relative to the request-URI that is in the request when it reaches the agent generating the response, and so is transport-dependent. Hence, MPR should be attempted for 410 responses.

3.11. 412 Conditional Request Failed

The 412 response [3] is used by a destination UA to indicate that the request cannot be processed because UA does not have information which is a precondition for successfully processing the request. Hence, MPR should not be attempted for 412 responses.
3.12. 413 Request Entity Too Large

According to [1] section 21.4.11, the 413 response is generated only by destination a UAS to indicate that it cannot process the body of the request. Hence, MPR should not be attempted for 413 responses.

3.13. 414 Request-URI Too Long

According to [1] section 21.4.12, the 414 response is generated only by destination a UAS to indicate that the request-URI is longer than it can process. The request-URI of a request can depend on the path by which the request reached the UAS, but it not expected in practice that the request-URI seen by the UAS (almost always one of its contact URIs) will be different in an MPR fork. Hence, MPR should not be attempted for 414 responses.

3.14. 415 Unsupported Media Type

According to [1] section 21.4.13, the 415 response is generated only by destination a UAS to indicate that it cannot process the body of the request. Hence, MPR should not be attempted for 415 responses.

3.15. 416 Unsupported URI Scheme

According to [1] section 21.4.14, the 416 response is generated only by destination a UAS to indicate that the request-URI has a scheme that the UAS does not support. The request-URI of a request can depend on the path by which the request reached the UAS, but it not expected in practice that the request-URI seen by the UAS will be different in an MPR fork. Hence, MPR should not be attempted for 416 responses.

3.16. 417 Unknown Resource-Priority

The 417 response [4] means that an actor did not understand the Resource-Priority header. Since the Resource-Priority header controls access to transport resources, an MPR fork may avoid the problem, and so MPR should be attempted.

3.17. 420 Extension Not Supported

The 420 response means that an actor did not support an extension that was named in a Require or Proxy-Require header. If the failure was due to Proxy-Require, MPR should be attempted. If the failure was due to Require, MPR should not be attempted. Unfortunately, these two conditions cannot be distinguished, which indicates that it is desirable to create a new response code for one of these cases. In principal an actor can determine from the option-tag listed in the Unsupported header of the response whether the unsupported feature is
a proxy feature or an endpoint feature. But implementing this strategy requires knowing the nature of all option-tags that will be used.

Unfortunately, splitting the 420 response code into two code may confuse UACs, as a UAC may be designed so that if it receives a 420 it resends the request without demanding support for the option-tag. Old UACs receiving the new response code may be preventing from executing this strategy.

3.18. 421 Extension Required

The 421 response is used by a UAS to indicate that a response to the request cannot be generated without using an extension that is not listed in the Supported header of the request. Since the response is only generated by a UAS, MPR should not be attempted.

3.19. 422 Session Interval Too Small

The 422 response [5] can be generated with by a UAS or by a proxy, so it is not possible to determine whether to attempt MPR or not. However, as far as I can tell, in all cases the best recovery strategy is to return the response to the UAC so that it can re-send the request with a larger minimum session timer in the Min-SE header, as detailed in [5]. So a 422 response should not cause MPR.

3.20. 423 Interval Too Brief

The 423 response is generated by a registrar in response to a REGISTER request. Hence, MPR should never be attempted for 423 responses.

3.21. 428 Use Identity Header

The 428 response, along with the 436, 437, and 438 responses, are generated by a SIP agent acting as a verifier [2]. Since the invocation of the verifier can depend on the transport path of the request, MPR should be attempted for these responses.

3.22. 429 Provide Referrer Identity

The 429 response [6] is used by UA that is a referee or a refer-target to demand that the REFER or the REFER-generated request contain a Referred-By header. Hence, MPR should not be attempted for a 429 response.
3.23. 436 Bad Identity-Info

See Section 3.21.

3.24. 437 Unsupported Certificate

See Section 3.21.

3.25. 438 Invalid Identity Header

See Section 3.21.

3.26. 480 Temporarily Unavailable

The 480 response is used to indicate various conditions which prevent the user from being contacted, but which may change with time. These conditions include "no UA registered for this AOR" and "do not disturb". Unfortunately, for transport-related errors like "no UA registered for this AOR", MPR would be helpful, but for end-user conditions like "do not disturb", MPR would not be helpful. So this response code should be split into two response codes.

3.27. 481 Call/Transaction Does Not Exist

The 481 response is used by UAs to indicate that handling the request requires the existence of a referenced dialog, but the UA has no knowledge of such a dialog. This response is used to reject apparent within-dialog requests for which the necessary dialog does not exist, or for which the necessary dialog usage does not exist within the dialog. In addition, it is used to reject a CANCEL request which does not match a current transaction. In all of these cases, MPR is of no use.

3.28. 482 Loop Detected

The 482 response indicates that an agent has discovered a loop in request routing. Since an MPR fork might travel over a route that does not lead into a loop, MPR should be attempted for 482 responses.

3.29. 483 Too Many Hops

The 483 response indicates that an agent has discovered a loop in request routing via exhaustion of the Max-Forwards count. Since an MPR fork might travel over a route that does not lead into a loop, MPR should be attempted for 483 responses.
3.30.  484 Address Incomplete

The 484 response is returned to a UAC only by an originating proxy which is cooperating with the UAC to accumulate a dial string. Since such processing does not admit MPR processing, the MPR forking decision will never be presented with a 484 response.

3.31.  485 Ambiguous

The 485 response indicates that a SIP agent could not determine how to resolve an ambiguous request-URI. Like all responses commenting on a request-URI, it is path-dependent, and so MPR should be attempted for this response.

3.32.  486 Busy Here

The 486 response is used by a UAS to indicate that it or its user is busy and so cannot accept another INVITE at this time. (486 is also used to indicate a Do-Not-Disturb condition.) MPR should not be attempted on a 486 response.

3.33.  487 Request Terminated

The 487 response is used to indicate that a request has been prematurely terminated because the UAC or agent received a CANCEL request before the request was fully processed. This can result from the UAC canceling the request, a destination proxy canceling a request to a UA because of ring-no-answer, or (slightly outside of the standard) a UA rejecting a request due to ring-no-answer. In any case, MPR would not help the situation.

Unfortunately, RFC 3261 leaves a UAS no proper response to report that it terminated an unanswered INVITE on its own, as the 487 response is supposed to be generated only when a CANCEL is received for an INVITE. In addition, the same response code is used if the UAC cancels an INVITE as if the destination proxy cancels a ring-no-answer call, which works against coherent call processing logic. For protocol cleanliness, it would be desirable to define a "Ring No Answer" 4xx response. However, MPR should not be tried either for requests that are canceled by the UAS or requests that met with ring-no-answer, and so, strictly speaking, no additional response code is needed to support MPR.

3.34.  488 Not Acceptable Here

The 488 response is used by a UAS to indicate that the proposed session offer in an INVITE is unacceptable. Hence, MPR forking should not be tried for a 488 response.
3.35.  489 Bad Event

The 489 response [7] is used by a UAS to indicate that it does not implement the Event header in a SUBSCRIBE request. Hence, MPR forking should not be tried for a 489 request.

3.36.  491 Request Pending

The 491 response is used by a UAS to reject a re-INVITE which it received while it has an INVITE request outstanding on the dialog. Since 491 can only be returned within-dialog, MPR forking never applies.

3.37.  493 Undecipherable

The 493 response indicates that the UAS cannot decrypt the request. Hence, MPR forking should not be attempted.

3.38.  494 Security Agreement Required

The 494 response [8] is used by a SIP agent to tell the UAC what security mechanisms it supports. Since it depends on the transport path, MPR forking should be attempted for 494 responses.

3.39.  500 Server Internal Error

The 500 response code is used by SIP agents to report a number of error conditions, some of them intrinsic to the request, and others internal to the agent:

- failure of an otherwise valid request due to an internal error ([1] section 10.3)
- a request has a CSeq value less than the highest CSeq received previously ([1] section 12.2.2 and 14.2)
- a proxy received only 503 responses to a request, but the proxy itself is capable of processing further requests ([1] section 16.7)

The first two cases are discovered by UAs, and MPR would be no help. But (since 503 responses are in practice always generated by proxies) the final case is a path-dependent failure of a proxy, and MPR should be attempted. Thus, it would help if the 500 response code was split into two codes.
3.40. 501 Not Implemented

The 501 response, like the similar 405 response, should never be generated by a proxy, and so MPR should not be attempted for it.

3.41. 502 Bad Gateway

The 502 response is used by a proxy or gateway to indicate that it received an invalid response from a downstream agent. Thus, this response is path-dependent, and MPR should be attempted for it.

3.42. 503 Service Unavailable

From RFC 3261 [1] section 21.5.4:

A client (proxy or UAC) receiving a 503 (Service Unavailable) SHOULD attempt to forward the request to an alternate server.

Thus, MPR should be attempted for 503 responses.

3.43. 504 Server Time-out

Like the 502 response, the 504 response is path-dependent, and so MPR should be attempted for it.

3.44. 505 Version Not Supported

Like the 501 response, the 505 response is path-dependent, and so MPR should be attempted for it.

3.45. 513 Message Too Large

Like the 502 response, the 513 response is path-dependent, and so MPR should be attempted for it.

3.46. 580 Precondition Failure

A 580 response [9] is generated by a server to indicate that it is unwilling to meet the preconditions in an SDP offer. Thus, MPR should not be attempted for it.

3.47. 6xx

All 6xx responses request termination of all forking for the request, and a fortiori, prevent any attempt at MPR.

(Since an automated agent cannot reliably determine the original source of the request (which may have been forwarded from one AOR to
another several times), 6xx responses are NOT RECOMMENDED, and if used, should only be generated by a UA under direct instruction from a user (or a user-specified policy).
4. Ambiguous Response Codes

The following response codes are defined in ways which prevents a proxy from determining whether it should attempt multiple path routing or not. Ideally, their function should be split into two different response codes, one for to be generated by proxies (for which MPR should be attempted) and one to be generated by UAs (for which MPR should not be attempted).

420 The 420 response code is used to report that a proxy or UAS does not support an extension listed in a Require or Proxy-Require header. But it does not distinguish whether a proxy or UAS is rejecting the request.

480 The 480 response code is used to report various conditions which prevent the user from being contacted, but which may change with time. These conditions include "no UA registered for this AOR" and "do not disturb". But it does not distinguish transport-related conditions from UA (or end-user) conditions.

487 The 487 response can be the response from a request canceled by the UAC or the result of a ring-no-answer condition. For protocol cleanliness, it would be helpful to split ring-no-answer from other uses of 487, and to allow a UAS to spontaneously time out an INVITE with a ring-no-answer response.

500 The 500 response code is used to report UA-related errors, as well as when a request could not reach its destination because intermediate agents were overloaded. For MPR, it would be helpful to split the latter condition into a separate response code. (This new response code would function much like the "reorder" signal of the PSTN.)
5. Proposal

Based on the above analysis, a tentative proposal for revised and new response codes is:

5.1. 420 Extension Not Supported By UA

An agent that would otherwise be a UAS for the request does not support an option-tag specified in the Require header of the request.

5.2. 424 Extension Not Supported By Proxy

An agent that would otherwise proxy the request does not support an option-tag specified in the Require-Proxy header of the request.

5.3. 480 Temporarily Unavailable At UA

An agent that would otherwise be a UAS for the request cannot contact the user.

5.4. 495 Temporarily Unavailable At Proxy

An agent that would otherwise proxy the request cannot contact a UAS for the user.

5.5. 487 Request Canceled

The request has been terminated due to receiving a CANCEL from upstream. (Note that if a proxy sends an INVITE to a UAS, and then after a timeout, sends a CANCEL in order to handle ring-no-answer situations, the UAS must send a 487 response, but the proxy may forward a 496 upstream.)

5.6. 496 Ring-No-Answer

The request reached the UAS but the user did not respond to the request.

5.7. 500 Server Internal Error

Unspecified failure of processing of a request.

5.8. 506 Service Unavailable Downstream

An element one or more steps downstream gave a 503 response. Immediately retrying the same request is unlikely to succeed, but other requests that would be sent to the same next-agent-downstream may succeed.
6. Security Considerations

Alternate path retry presents no security considerations that are known to the author beyond what is present in non-MPR SIP system architectures.
7. Revision History

7.1. draft-worley-sip-redundancy-response-00

First version.

7.2. Changes from draft-worley-sip-redundancy-response-00 to draft-worley-sip-redundancy-response-01

Add note that the new variant of 500 would be like "reorder" in the PSTN.

Add note that dealing with 402 can be postponed to when 402 is standardized.

Add section proposing how to split response codes.
8. Normative References


Author’s Address

Dale R. Worley
Pingtel Corp.
400 West Cummings Park, Suite 2200
Woburn, MA  01801
US

Phone: +1 781 938 5306
Email: dworley@pingtel.com
URI:  http://www.pingtel.com
Full Copyright Statement

Copyright (C) The IETF Trust (2007).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY, THE IETF TRUST AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).