SIPPING WG
Internet-Draft
Expires: August 15, 2005

J. Peterson
NeuStar

February 14, 2005

Retargeting and Security in SIP: A Framework and Requirements
draft-peterson-sipping-retarget-00

Status of this Memo

Copyright Notice

Abstract

Retargeting, the alteration by intermediaries of the destination of a Session Initiation Protocol (SIP) request, is a common practice performed during the routing of a call. Some forms of retargeting, however, give rise to security problems and potential functional gaps in SIP. This document provides a general framework for discussion of the security problems relating to retargeting, and gives requirements for mechanisms that might overcome these problems.

Table of Contents

1. Introduction 3
2. Problems with Retargeting 4
2.1 Legitimacy of Retargeting 5
2.2 Response Identity 6
2.2.1 The Unanticipated Respondent Problem 7
2.3 Establishment of Security Parameters 8
2.4 Consent of the UAC 10
2.5 Furtherance of Transitive Trust 11
3. Summary of Threats 12
4. Benefits of Retargeting 12
5. Requirements for Securing Retargeting 14
6. Security Considerations 14
7. IANA Considerations 14
Author's Address 15
8. References 14
8.1 Normative References 14
8.2 Informative References 15
A. Acknowledgments 15
Intellectual Property and Copyright Statements 16

1. Introduction

Retargeting is the process by which a SIP intermediary (such as a proxy server) alters the Request-URI of a SIP request before it forwards that request. A proxy server may retarget when it determines (by accessing a location service, perhaps) that one or more new target URIs are eligible to receive the request. Retargeting to a contact address URI discovered by a location service for the address-of-record in the Request-URI of a request is common. However, proxy servers may also retarget to other proxy servers, potentially leading to significant signaling chains between the UAC and UAS.

When one or more new targets have been identified for a request, RFC3261-compliant intermediaries can choose either to retarget or redirect (i.e., send a 3xx response code) on a per-request basis. Perhaps the most important reason for retargeting is efficiency, in terms of overall messages required. In the simplest case (one new target for the request is discovered):

A redirecting intermediary must send a 3xx response to the UAC. The UAC must both acknowledge this response and send a new request to the new target. Three messages total are required.

A retargeting intermediary sends only one message to the destination UAS - nothing is sent back to the UAC.

However, retargeting has its downside: namely, the UAC does not know where its request will be going. This might not immediately seem like a serious problem; after all, when one places a telephone call, one never really knows if it will be forwarded to a different number, who will pick up the line when it rings, and so on. The user-driven signaling environment of SIP, however, makes this condition much more problematic than its analog in traditional telephony.

Currently, RFC3261 allows retargeting to occur only under very

specific circumstances: a proxy server may retarget only if it is 'responsible' for the domain in the host portion of the Request-URI. Unfortunately, this concept of 'responsibility' is not sufficiently specified, and moreover, a UAC has no way of knowing which proxy server in the network performed any retargeting. Consequently, the UAC has no assurance at a protocol level against malicious or misguided retargeting by intermediaries which are not 'responsible' for the target.

In fact, RFC3261 includes a number of normative constraints on intermediary behavior which suggest, tacitly, an authorization relationship between the UAC and a proxy server. However, since there is no explicit account of what exactly a UAC does and does not

authorize a proxy server to do, there is little by way of mechanism in SIP to chaperone proxy server behavior. Since everything that is not prevented by mechanism tends to be permitted in practice, the reality is that SIP proxy servers behave as if UACs have no authority to exert any policy controls over the handling of their requests and accordingly, the normative constraints in RFC3261 are routinely flouted.

Unfortunately, this inability to police intermediary behavior leads to practicable attacks against SIP and various weaknesses in the authorization policies needed by user agents. In order to address such problems, a UAC must be capable of implementing authorization policies informed by questions that are, today, essentially unanswerable, such as:

How can a UAC determine the URI that a request has eventually reached?

How can a UAC determine which intermediary has changed the target of the request?

This draft attempts to tease out the actual authority which a UAC invests in a proxy server, in so far as it relates to retargeting, by examining the negative space: the problem cases where clearly, the UAC cannot grant permissions to intermediaries without exposing itself to attacks or policy failures. It then provides some requirements for a corresponding security mechanism that might actually constrain the retargeting behavior of intermediaries in order to improve the overall security of the protocol.

2. Problems with Retargeting

SIP is a protocol that relies on intermediaries to perform application-layer routing functions. It is necessary for intermediaries to perform this function in order to support an architectural layer of indirection between the identifiers of users (the SIP "address-of-record" or AoR) and the identifiers of devices (SIP "contact addresses"). The mappings between these two types of identifiers are provided by the abstract location service function of SIP, which is accessed by proxy servers when they make forwarding decisions. This function is essential and integral to SIP's operation.

However, this function also makes it difficult to differentiate an intermediary that is behaving legitimately from an attacker. If SIP is designed in such a way that malicious attacks against the protocol cannot be distinguished from the legitimate activities of intermediaries, then SIP will never be securable.

Ultimately, intermediaries can discharge their responsibilities while

giving user agents enough information to detect attacks and to make reasonable authorization decisions. In order to understand how intermediaries would need to behave for this to be the case, a detailed examination of the gaps in SIP's current security story relating to retargeting is necessary.

The problems discussed in this section exhibit the following three qualities:

Undetectable: The originating user agent has no means of anticipating that the condition will arise, nor any means of determining that it has occurred until the negative consequences have already been realized.

Unpreventable: There is no existing security mechanism that might be employed by the originating user agent in order to guarantee that the condition will not arise.

Entailed by retargeting: If retargeting did not exist, this problem would not arise.

2.1 Legitimacy of Retargeting

RFC3261 allows retargeting to occur only if the retargeting intermediary is responsible for the domain indicated by the Request-URI of the request. For example, if a request was sent to sip:bob@example.com, then a proxy server at example.com would be authorized to retarget the request, but a proxy server at example.org would not be authorized to retarget the request.

However, RFC3261 offers no way to enforce this rule. There is no way for the UAC or UAS to detect which intermediary retargeted a request. Accordingly, if due to local policy a UAC were forced to send its request through a proxy server example.org on its way to the proxy server at example.com, the example.org proxy server might malicious retarget the request to, say, a user at example.org, and the UAC would have no way to determine that example.com had not authorized this retargeting.

The rule in RFC3261 is intended to combat various forms of service hijacking. The domain in the Request-URI is permitted to retarget because it presumably has a business relationship with the target user, since that user can provision its local service with registrations or other administrative means. Other domains, however, have no such relationship with the target user, and might retarget to their own advantage. If we imagine that example.org is a hotel, for example, the example.org proxy server might retarget requests for various local pizza shops (sip:orders@pizza.example.com) to a specific, preferred pizza shop (sip:orders@pizza.example.net) with which the hotel enjoys an underhanded business arrangement.

Nothing will be able to prevent intermediaries from changing the Request-URI of a SIP request. It is impossible to provide integrity protection over the Request-URI because there are cases where it changes of necessity (such as intradomain routing of GRUUs and so on). However, if it were possible to determine which proxy server was responsible for changing the target of the request, the UAC could abandon requests that have been maliciously mishandled and perhaps take up some recourse against the misbehaving domain. As it stands, since the condition in undetectable, domains might engage in this behavior without fear of reproach.

2.2 Response Identity

When Alice sends a request to Bob, Bob may want to send a secure

response back to Alice. What is the purpose of a secure response? Bob secures his response so that Alice can make certain authorization and policy decisions based on the security of the response. Usually, this requires at least providing integrity protection over the response (as a whole or in part) and providing cryptographic authentication of the sender of the response. Alice might accept the response, or perhaps discard it, on the basis of those security properties. In the absence of this authentication and integrity, it might be possible for a third-party attacker, whom we'll call Edgar, to eavesdrop on the request, and forge their own response (impersonating Bob) with an inappropriate response code, or inappropriate SDP answer, or what have you.

In most respects, a security mechanism to defeat this impersonation threat in responses would be very similar to a comparable mechanism for requests - except for the problem of retargeting. If Alice's request to Bob were retargeted to Carol, and Carol wished to send a secure response to Alice, this can create a number of headaches for existing security practices.

The first, and perhaps most important, problem is that Carol's domain may not be the domain to which Alice sent the request. Any authentication or integrity provided by a solution like sip-identity [6] would require a signature over the response by the responding domain. If Alice sent the request to sip:bob@biloxi.example.com, and the request is retargeted to sip:carol@cleveland.example.org, then an identity service in Carol's domain, clearly, would not be able to sign for the domain in the To header field of the request and response (which would be biloxi.example.com). This is called the response identity problem (although this problem also arises for requests sent in the backwards direction within a dialog).

The root problem here is, of course, that the To header field of the request and response does not contain the identity of the actual

respondent. In these cases, Carol's domain cannot provide an identity signature, and accordingly responses cannot be secured when retargeting has occurred. This creates opportunities for eavesdroppers to launch impersonation attacks in responses.

The solution space for this problem generally involves providing an identity for the respondent which the responding domain can sign.

There are a number of ways this might be achieved; one common suggestion is that some kind of supplemental identity field, colloquially known as the 'connected party' field, should be added to responses in order to identify the party that was actually reached. Presumably, this field would be understood to clobber the value in the To header field of responses and represent the genuine identity of the respondent. However, this sort of solution gives rise to a new class of 'unanticipated respondent' problems.

2.2.1 The Unanticipated Respondent Problem

When retargeting has occurred, the respondent to a request may not be identified by the To header field of the request and response. For example, the To header field of a request may designate sip:bob@biloxi.example.com, but an intermediary may retarget that request to sip:carol@cleveland.example.org. That request might then land at a UAS controlled by Carol, and Carol may be the respondent to that request.

In this case we would say that Carol is the 'connected party'. In order to provide secure responses to Alice's request, one could argue that the 'connected party' information, i.e., some URI that identifies Carol, should be communicated to Alice in responses. This would let Alice know that some party other than the anticipated party (Bob) was reached by the request, and give her an identity of this unanticipated party. This information could then be cryptographically signed by the responding domain, presumably resolving any concerns about providing response identity.

The problem is that Edgar the eavesdropper can do the same thing that Carol just did. That is, Edgar can construct a response with a 'connected party' value indicating himself. He can even get his domain (evil.example.net) to sign that 'connected party' value. When Edgar sends the response back to Alice, how can Alice tell the difference between his forged reply and Carol's legitimate reply? Ironically, 'connected party' does not provide any new leverage against the very problem that response identity is supposed to solve - preventing an eavesdropper from sending a forged reply.

The reason why this is true is because allowing unanticipated respondents makes Alice's authorization decisions very complicated.

If Alice authorize is to authorize some set of respondents other than Bob to reply to her request, how does she arrive at that set? In the current retargeting regime, Alice has no way of knowing what targets might be selected for her request by an intermediary, and accordingly, she has no choice but to accept any respondents that contact her.

What is needed to address the response identity and unanticipated respondent problems, then, is a mechanism by which Alice can learn the set of targets which might be respondents to her request. This information would need to come from the domain that is changing the target of the request. In other words, when Alice sends a request to sip:bob@biloxi.example.com, biloxi.example.com would tell Alice that it was redirecting her request to sip:carol@cleveland.example.org. Alice could then set an authorization policy for this transaction that would only accept responses coming from Carol. This would allow Alice to tell the difference between a legitimate target of the request and an attacker.

The major distinction between this explicit indication to the UAC of a change of target and the traditional concept of 'connected party' is the source of the indication - 'connected party' assumes that the respondent or respondent's domain provides the identity of the respondent. The requirement articulated here is that the domain that changes the target provides the new target for the request. Of course, it is possible that the target of the request will change several times during the routing of a request; in that case, several such indications would need to be given to the UAC by the mechanism so that a complete chain can be formed from the original form of the Request-URI to the final target.

Note that the considerations above treat 'connected party' only as it relates to security and response identity; some have argued that 'connected party' is a nice feature to have in an environment with retargeting even if it were inert from a security perspective.

2.3 Establishment of Security Parameters

Consider a case where Alice and Bob would like to share a voice conversation over the Internet which is secured by sRTP. In order to do so, Alice must generate a session key that will be used to encrypt media sent to her and place that session key inside an SDP offer. She must then send that offer to Bob in an INVITE. If the body of that invite is not encrypted, however, an eavesdropper might learn her symmetric key and use it to decrypt the media sent to her by Bob. Accordingly, she must learn a public key for Bob, and use that key to encrypt her SDP offer. One manner in which she might learn a public key for Bob is described in the certs [7] draft.

What happens if Alice's request to Bob is retargeted to Carol? Since Carol does not (and should not) possess the private key corresponding to Bob's public key, she would send some sort of failure response code (perhaps a 493 Undecipherable). Carol and Alice might then reconnect in any of a number of ways; the manner suggest in RFC3261 is that Carol will return her certificate in the body of the 493 response, and the Alice will re-initiate the session, encrypting with Carol's certificate.

Retargeting gives rise to a couple of problems here. One is just a variant of the previously-discussed 'unanticipated respondent' problem - Alice has no way of knowing if Carol is actually an attacker who sends a 493 in order to bid-down the security for the ensuing RTP session [note: this is a very serious problem, and is only being de-emphasized here because an essentially identical problem has already been discussed above]. But even beyond determining whether the 493 should be accepted, recovery from a 493 is awkward and fraught with risks because the scope of the key returned in the 493 is ambiguous when retargeting has occurred. When Alice receives a 493, whose key does she think she is learning from the body? What if Alice re-initiates her request, this time with the body encrypted with Carol's key, but the new request lands on a UA controlled by Bob? These cases are especially problematic when Carol's key is self-signed.

Ultimately, the very potential for retargeting significantly discourages the use of confidentiality in SIP. Since Alice cannot anticipate when retargeting will occur, she cannot anticipate when she should expect an encrypted message to be accepted. The complexity of managing 493 responses and making the resulting authorization decisions and re-initiated messaging exchanges is fairly prohibitive.

The 'connected party' mindset would suggest that Carol should disambiguate the 493 by providing her identity along with the certificate, informing Alice to re-originate the quest directly to Carol and encrypt with Carol's key. In this case, the 493 would behave something like a 3xx response, redirecting Alice's request from Bob to Carol. The problem with this approach is the same as the comparable 'connected party' problems above - the respondent is not the one authorized to retarget the request, and accordingly, the respondent should not be the one to tell the UAC the request's new target.

This whole class of failure would be preventable if Alice were able to guess who would eventually receive her request. Since Alice can't access the location service at the domain which her request targets, though, she has to rely on that domain to tell her. If the

intermediary that changed the target of the request could know, hypothetically, that Alice would need to rekey her request if the target changes, it could avoid the whole step of forwarding the retargeting the request to Carol and simply tell Alice (through a 3xx or something comparable) that Carol is the new target.

While the example given here is specific to keying, there are a number of ways in which a request might need to be modified if its destination were to change.

2.4 Consent of the UAC

Alice may be friends with Bob, but not at all cordial with Bob's friend Carol. If Bob configures his SIP service to forward messages to Carol, then Alice might reach Carol when she sends a request to Bob. While the consequences of this for an INVITE might not be so bad (Alice could hang up when she hears Carol's voice, or what have you), some SIP requests (such as the MESSAGE method) contain content that might be viewed by the recipient without any further activity on the part of the sender.

. Because this sort of problem is a fact of life in the existing telephone system, it might seem unavoidable for SIP. However, there are Internet systems where this class of problem doesn't arise. One example would be the ad-blocker software built into web browsers. When a web browser with an ad-blocker downloads a page, it inspects the list of hyperlinks on the page and determines if any of them contain a blacklisted domain known to be associated with an advertiser. Those hyperlinks are never traversed. Ultimately, this works because HTTP has an endpoint consent model - when you access a web resource, it tells you what related resources you might want to access rather than pre-emptively accessing related resources on your behalf.

In fact, retargeting is the reason why SIP does not have an endpoint consent model. The lack of an endpoint consent model prevents user

agents from exercise valuable authorization policies. Consider the question of how a blacklist might operate. If we imagine that Alice hates Carol, then Alice might add Carol to her user agent's blacklist. Whenever Carol calls Alice, the blacklist is automatically consulted for an authorization decision and Alice's user agent rejects, politely or impolitely, the session initiation request. When Alice places a call, though, she has no control over the target set that will be selected for the call. If Carol is selected by a retargeting intermediary, then Alice will be put in contact with Carol against her will; effectively, this circumvents Alice's blacklist. Of course, if the intermediary sent a redirection

At a high level, the main problem is when retargeting occurs, new targets are chosen by an intermediary without the consent of the UAC. By sending a request to a proxy server, the UAC is essentially giving "implied consent" that the request can be retargeted arbitrarily. To remedy this authorization policy defect, the UAC would have to have some ability to review the new targets for a request that have been chosen by an intermediary, and decide which of these targets it might want to pursue or not pursue before the request is forwarded to a new target.

If the UAC does not learn the new target until the respondent responds (i.e., the 'connected party' approach), this will be of little help when the response is an acknowledgment of a MESSAGE request that is grossly inappropriate for the recipient.

2.5 Furtherance of Transitive Trust

At a higher level, retargeting is also a source of transitivity for SIP. This is not in and of itself a direct threat, but an overall negative implication for the security model.

If Alice sends a request to sip:bob@biloxi.example.com, and that request is retargeted by biloxi.com to sip:carol@cleveland.example.com, it places Alice at a significant disadvantage if she received a Digest authentication challenge response (i.e., a 407) from cleveland.example.com. Per RFC3261 26.3.2.1, the establishment of a direct connection to a challenging proxy allows the UAC to compare the subject of the site certificate presented by that proxy via TLS with the "realm" parameter used in

Digest authentication. A correspondence between the two provides reference integrity - it ensures Alice that the challenge she is receiving from the realm 'cleveland.example.com' is actually coming from a proxy server at 'cleveland.example.com' rather than an imposter trying to capture her Digested credentials (which might then be attacked offline).

If biloxi.com retargets and proxies the request, Alice will only have established a connection to biloxi.example.com, and thus she will have no may to match the challenge she received against the certificate of cleveland.example.com. If biloxi.example.com redirects, however, Alice can form her own TLS connection to cleveland.example.com to make this determination. Effectively, Alice must rely on Bob's domain to relay messages to and from Carol's domain faithfully, and if there is something suspicious about the response, Alice will not be able to determine authoritatively if Bob's domain or Carol's domain is responsible. In this sense, retargeting has put Alice into a position of increased transitive trust.

3. Summary of Threats

In terms of the typical Internet threat model, the sorts of threats described above require a relatively high level of sophistication in an attacker. In order for an attacker to impersonate a respondent, for example, the attacker must be able to capture dialog identifying parameters (which entails inspecting SIP signaling in transit), create plausible responses on the fly, insert this responses back into the signaling stream (or make it appear so to the originator through various forms of spoofing), and possibly squelch legitimate responses that might alert the originator to malfeasance. This essentially requires the attacker to be a man-in-the-middle. The use of judicious channel security, such as TLS, between proxy servers can prevent eavesdropping and many forms of signaling insertion or squelching.

Unfortunately, in order to police the behavior of proxy servers themselves, however, one must guard against precisely that: a man-in-the-middle. The primary attacker envisioned by this draft is an intermediary which is legitimately within the path of SIP signaling.

In summary, the threats that have been discussed in this section include:

Service hijacking: Unscrupulous domains can retarget requests, disobeying the rules of RFC3261, without significant risk of detection.

Insecure responses: When retargeting has occurred, traditional signatures cannot be applied to SIP responses because the identity of the sender is not reflected in the response. Modifying responses in order to communicate the identity of the sender does not seem to fix the problem.

Confidentiality problems: It is easy to bid-down attempts to use confidentiality in SIP message bodies, and when SIP responses are used as a key distribution mechanism, retargeting makes the intended scope of those keys ambiguous.

Circumvention of blacklists: Retargeting can cause a SIP user agent to come into contact with an entity that has been blacklisted.

Rampant transitivity: Retargeting increases the amount of indirection between the originating user agent and various intermediaries and endpoints with which it might need to establish a security relationship.

4. Benefits of Retargeting

Given that an intermediary has two different ways that it can choose

to change the target of a request - redirection and retargeting - what are the advantages of retargeting over redirection?

Messaging Efficiency: Instead sending a response back to the UAC whenever the target needs to change, intermediaries just forward the request. This can be significant in environments where UACs are constrained in terms of computational power or bandwidth. However, it can significantly increase the amount of signaling and state that an intermediary needs to manage, especially in forking cases.

Target set privacy: If an intermediary discovers more than one potential target for a request, then the policy of the original target user may favor concealment of the whole target set from the UAC. In retargeting cases, a certain amount of information about the new target would be revealed in successful responses (the Contact header and so on), but unresponding targets would be

totally hidden from the UAC. Redirection of course discloses the target set to the UAC.

Target set ordering policy: If there is more than one possible target for a request, an intermediary can guarantee strict ordering of the target set by performing sequential forking itself. If the intermediary redirects to the UAC, it may use qvalues to suggest and ordering, but it has no guarantee that the ordering will be followed.

Dialog control: By redirecting a dialog-forming request, an intermediary may put itself out of the loop of future signaling in the dialog. When an intermediary retargets a request, it may add a Record-Route header and thus remain in the path of future messages in the dialog.

NAT traversal: In some cases, an intermediary that changes the target of a request is, in fact, the one SIP entity that can contact the new target. If, for example, the new target is behind a NAT, and maintains a persistent TLS connection to the retargeting intermediary, any requests heading to the target must go through the intermediary.

The most important question about these benefits is the following: are they genuinely achievable only if retargeting transpires, or can they replicated via redirection? For example, the ordering of a target set can be preserved if the targets are shared via redirection one at a time with the UAC (though this will substantially decrease messaging efficiency). Many other effects, including privacy, dialog control, and even NAT traversal can be replicated by redirecting to an synthetic target (like a GRUU or an anonymized URI) which dereferences to an intermediary under the control of the retargeting domain.

5. Requirements for Securing Retargeting

In an ideal world, it would be possible for a UAC to have a strong assurance that intermediaries were behaving properly, and furthermore to have the capability to differentiate between properly-behaving intermediaries and attackers.

It MUST be possible for a UAC to detect when a request has been retargeted.

A domain that changes the target of a request MUST be capable of informing the UAC of the new target(s).
The mechanism MUST allow simple intradomain retargeting in cases where persistent TLS connections are used as a NAT traversal mechanism.
It must be possible for a domain that changes the target of a request to inform the UAC of the new target(s) prior to contacting any of the new target(s). There MUST furthermore be a way for intermediaries to determine when UACs require prior information about new targets.
It MUST be possible to preserve the privacy of targets and potential targets of requests.
It MUST be possible to preserve the ordering of a target set desired by the domain that changes the target of a request.

6. Security Considerations

This document provides a framework and requirements for addressing an important gap in SIP's existing security practices.

7. IANA Considerations

This document contains no considerations for the IANA.

8. References

8.1 Normative References

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

[2] Bradner, S., "Key words for use in RFCs to indicate requirement levels", RFC 2119, March 1997.

[3] Peterson, J., "A Privacy Mechanism for the Session Initiation Protocol (SIP)", RFC 3323, November 2002.

[4] Rosenberg, J. and H. Schulzrinne, "Session Initiation Protocol (SIP): Locating SIP Servers", RFC 3263, June 2002.

[5] Peterson, J., "Session Initiation Protocol (SIP) Authenticated

Identity Body (AIB) Format", RFC 3893, September 2004.

[6] Peterson, J., "Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP)", draft-ietf-sip-identity-04 (work in progress), February 2005.

[7] Peterson, J. and C. Jennings, "Certificate Management Service for SIP", draft-ietf-sip-certs-01 (work in progress), February 2005.

8.2 Informative References

Author's Address

Jon Peterson
NeuStar, Inc.
1800 Sutter St
Suite 570
Concord, CA 94520
US

Phone: +1 925/363-8720
EMail: jon.peterson@neustar.biz
URI: http://www.neustar.biz/

Appendix A. Acknowledgments

Intellectual Property Statement

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 or 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.

Disclaimer of Validity

Copyright Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the

Internet Society.