Independent Submission
Internet-Draft
Intended status: Standards Track
Expires: September 11, 2017

M. Baeuerle
STZ Elektronik

March 10, 2017

Cancel-Locks in Netnews articles
draft-baeuerle-netnews-cancel-lock-02

Abstract

This document defines an extension to the Netnews Article Format that may be used to authenticate the cancelling and superseding of existing articles.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on September 11, 2017.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Table of Contents

1. Introduction

The authentication system defined in this document is intended to be used as a simple method to verify that the author of an article which cancels ([RFC5537] Section 5.3) or supersedes ([RFC5537] Section 5.4) another one is either the poster, posting agent, moderator or injecting agent that processed the original article when it was in its proto-article form.

One property of this system is that it prevents tracking of individual users.

There are other authentication systems available with different properties. When everybody should be able to verify who the originator is, e.g. for control messages to add or remove newsgroups ([RFC5537] Section 5.2), an OpenPGP [RFC4880] signature is suited.

1.1. Conventions Used in This Document

Any term not defined in this document has the same meaning as it does in [RFC5536] or [RFC5537].

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

1.2. Author's Note

Please write the letters "ae" in "Baeuerle" as an a-umlaut (U+00E4, "&#228;" in XML), the first letter in "Elie" with an acute accent (U+00C9, "&#201;" in XML), the letters "ss" in Janssen as an eszett (U+00DF, "&#223;" in XML) and the letters "ue" in Baden-Wuerttemberg as an u-umlaut (U+00FC, "&#252;" in XML) wherever this is possible.

2. Header Fields

This section describes the formal syntax of the new header fields using ABNF [RFC5234][RFC7405]. It extends the syntax in Section 3 of [RFC5536] and non-terminals not defined in this document are defined there. The [RFC5536] ABNF should be imported first before attempting to validate these rules.

The new header fields Cancel-Lock and Cancel-Key are defined by this document, they follow the rules described in [RFC5536] Section 2.2:

```
fields =/ *( cancel-lock / cancel-key )
```

Each of these header fields MUST NOT occur more than once in an article.

Both new header fields contain lists of encoded values. Every entry is based on a <scheme>:

```
scheme       = %s"sha-256" / 1*scheme-char / obs-scheme
scheme-char  = LOWER / DIGIT / "-"
LOWER        = %x61-7A  ; lowercase characters [a-z]
```

The hash algorithms for <scheme> are defined in [SHA], see also [RFC1321] and [RFC6151] for MD5, [RFC3174] for SHA1 and [RFC6234] for the SHA2 family. The Base64 encoding used is defined in Section 6.8 of [RFC2045].

This document defines one value for <scheme>: "sha-256". This scheme is mandatory to implement.

Note that the obsolete syntax <obs-scheme> was defined case-insensitive. This is changed in this document and the scheme MUST now be generated with lowercase letters.

The case sensitivity of <scheme> is defined to simplify the checks.

2.1. Cancel-Lock

```
cancel-lock     = "Cancel-Lock:" SP c-lock-list CRLF
c-lock-list     = c-lock *(CFWS c-lock) [CFWS]
c-lock          = scheme ":" c-lock-string
c-lock-string   = *(4base64-char) [base64-terminal]
base64-char     = ALPHA / DIGIT / "+" / "/"
base64-terminal = 2base64-char "==" / 3base64-char "="
```

If <scheme> is not supported by an implementation, the corresponding <c-lock> element MUST be skipped and potential following <c-lock> elements MUST NOT be ignored.

<c-lock-string> is the Base64 encoded output of a hash operation (defined by <scheme>) of the Base64 encoded key "K" that is intended to authenticate the person or agent that created or processed respectively the article up to injection (inclusively):

```
base64(hash(base64(K)))
```

Because of the one-way nature of the hash operation the key "K" is not revealed.

2.2. Cancel-Key

```
cancel-key   = "Cancel-Key:" SP c-key-list CRLF
c-key-list   = c-lock *(CFWS c-lock) [CFWS]
c-key        = scheme ":" c-key-string
c-key-string = 1*base64-octet
base64-octet = ALPHA / DIGIT / "+" / "/" / "="
```

If <scheme> is not supported by an implementation, the corresponding <c-key> element MUST be skipped and potential following <c-key> elements MUST NOT be ignored.

<c-key-string> is the Base64 encoded key "K" that was used to create the Cancel-Lock header field (as defined in Section 2.1 of this document) of the original article:

```
base64(K)
```

The relaxed syntax definition of <c-key-string> above is required for backward compatibility. New implementations always SHOULD generate valid Base64.

3. Use

3.1. Adding an initial Cancel-Lock header field to a proto-article

A Cancel-Lock header field MAY be added to a proto-article by the poster or posting agent which will include one or more <c-lock> elements.

If the poster or posting agent doesn't add a Cancel-Lock header field to an article, then an injecting agent (or moderator) MAY add one or more provided that it positively authenticates the author. The injecting agent (or moderator) MUST NOT add this header field to an article unless it is able to authenticate all cancelling or superseding attempts from the poster and automatically add working Cancel-Key header fields for such articles.

Other agents MUST NOT add this header to articles or proto-articles that they process.

3.2. Extending the Cancel-Lock header field of a proto-article

If a Cancel-Lock header field has already been added to a proto-article then any agent further processing the proto-article up to the injecting agent (inclusively) MAY append additional <c-lock> elements to those already in the header.

Use cases for extending the Cancel-Lock header field:

- A moderator wants the ability to cancel articles after approving them.

- A news administrator wants the ability to cancel articles that were injected by its system (because they e.g. violate its abuse policy).

- An injecting agent acts representitive for posting agents without support for the autentication system described in this document.

Once an article is injected then this header MUST NOT be altered. In particular, relaying agents beyond the injecting agent MUST NOT alter it.

3.3. Adding a Cancel-Key header field to a proto-article

A Cancel-Key header field MAY be added to a proto-article containing a Control or Supersedes header field by the poster or posting agent which will include one or more <c-key> elements. They will correspond to some or all of the <c-lock> elements in the article referenced by the Control (with a "cancel" command as defined in [RFC5537]) or Supersedes header field.

If, as mentioned in Section 3.2 an injecting agent (or moderator) has added a Cancel-Lock header field to an article listed in the Control (with "cancel" command as defined in [RFC5537]) or Supersedes header field then (given that it authenticates the poster as being the same as the poster of the original article) it MUST add (or extend, if already present) the Cancel-Key header field with at least one <c-key> element that correspond to that article.

Other agents MUST NOT alter this header.

3.4. Check a Cancel-Key header field

When a serving agent receives an article that attempts to cancel or supersede a previous article via Control (with a "cancel" command as defined in [RFC5537]) or Supersedes header field, the system defined in this document can be used for authentication. The general handling of articles containing such attempts as defined in [RFC5537] is not changed by this document.

To process the authentication, the received article must contain a Cancel-Key header field and the original article a Cancel-Lock header field. If this is not the case, the authentication is not possible (failed).

For the authentication check, every supported <c-key> element from the received article is processed as follows:

1. The <code-string> part of the <c-key> element is hashed using the algorithm defined by its <scheme> part.

2. For all <c-lock> elements with the same <scheme> in the original article their <code-string> part is compared to the calculated hash.

3. If one is equal, the authentication is passed and the processing of further elements can be aborted.

4. If no match was found and there are no more <c-key> elements to process, the authentication failed.

4. Calculating the key data

This section is informative, not normative.

It is suggested to use the function K = HMAC(mid+sec) to create the key "K" for an article with Message-ID <mid>, where HMAC is outlined in [RFC2104]. <sec> is a secret held locally that can be used for multiple articles. This method removes the need for a per-article database containing the keys used for every article.

The local secret <sec> should have a length of at least the output size of the hash function that is used by HMAC (32 octets for SHA-256). If the secret is not a random value, but e.g. some sort of human readable password, it should be much longer. In any case it is important that this secret can not be guessed.

Note that the hash algorithm used as base for the HMAC operation is not required to be the same as specified by <scheme>. An agent that verifies a Cancel-Key header field simply checks whether it matches one of the <c-lock> elements with the same <scheme> in the Cancel-Lock header field of the original article.

Common libraries like OpenSSL can be used for the cryptographic operations.

5. Examples

Example data for creation of a <c-lock> element with HMAC-SHA256 (as suggested in Section 4):

```
Message-ID: <12345@mid.example>
```

```
   mid: <12345@mid.example>
   sec: ExampleSecret
   K  : HMAC-SHA256(mid+sec) ;"mid" used as HMAC message, "sec" used as HMAC
key
```

Calculation of Base64(K) using the OpenSSL command line tools in a POSIX shell:

```
$ printf "%s" "<12345@mid.example>" \
  | openssl dgst -sha256 -hmac "ExampleSecret" -binary \
  | openssl enc -base64
qv1VXHYiCGjkX/N1nhfYKcAeUn8bCVhrWhoKuBSnpMA=
```

This can be used as <c-key-string> for canceling or superseding the article <12345@mid.example>.

Calculation of Base64(SHA256(Base64(K))) required for <c-lock-string> using the OpenSSL command line tools in a POSIX shell:

```
$ printf "%s" "qv1VXHYiCGjkX/N1nhfYKcAeUn8bCVhrWhoKuBSnpMA=" \
  | openssl dgst -sha256 -binary \
  | openssl enc -base64
s/pmK/3grrz++29ce2/mQydzJuc7iqHn1nqcJiQTPMc=
```

Inserted into the header of article <12345@mid.example> it looks like this:

```
Cancel-Lock: sha-256:s/pmK/3grrz++29ce2/mQydzJuc7iqHn1nqcJiQTPMc=
```

Inserted into the header of an article that should cancel or supersede article <12345@mid.example> it looks like this:

```
Cancel-Key: sha-256:qv1VXHYiCGjkX/N1nhfYKcAeUn8bCVhrWhoKuBSnpMA=
```

Other matching pair of Cancel-Lock and Cancel-Key header fields:

```
Cancel-Lock: sha-256:RrKLp7YCQc9T8HmgSbxwIDlnCDWsgy1awqtiDuhedRo=
Cancel-Key: sha-256:sSkDke97Dh78/d+Diu1i3dQ2Fp/EMK3xE2GfEqZlvK8=
```

With obsolete syntax (requires case-insensitive parsing of <scheme> and uses a <c-key-string> with invalid/missing Base64 padding):

```
Cancel-Lock: sha1:bNXHc6ohSmeHaRHHW56BIWZJt+4=
Cancel-Key: ShA1:aaaBBBcccDDDeeeFFF
```

Let's assume that all the examples above are associated to the same article (e.g. created by different agents):

```
   Cancel-Lock: sha-256:s/pmK/3grrz++29ce2/mQydzJuc7iqHn1nqcJiQTPMc=
                sha-256:RrKLp7YCQc9T8HmgSbxwIDlnCDWsgy1awqtiDuhedRo=
                sha1:bNXHc6ohSmeHaRHHW56BIWZJt+4=
   Cancel-Key: sha-256:qv1VXHYiCGjkX/N1nhfYKcAeUn8bCVhrWhoKuBSnpMA=
               sha-256:sSkDke97Dh78/d+Diu1i3dQ2Fp/EMK3xE2GfEqZlvK8=
               ShA1:aaaBBBcccDDDeeeFFF
```

Manual checks using the OpenSSL command line tools in a POSIX shell:

```
   $ printf "%s" "qv1VXHYiCGjkX/N1nhfYKcAeUn8bCVhrWhoKuBSnpMA=" \
     | openssl dgst -sha256 -binary \
     | openssl enc -base64
   s/pmK/3grrz++29ce2/mQydzJuc7iqHn1nqcJiQTPMc=

   $ printf "%s" "sSkDke97Dh78/d+Diu1i3dQ2Fp/EMK3xE2GfEqZlvK8=" \
     | openssl dgst -sha256 -binary \
     | openssl enc -base64
   RrKLp7YCQc9T8HmgSbxwIDlnCDWsgy1awqtiDuhedRo=

   $ printf "%s" "aaaBBBcccDDDeeeFFF" \
     | openssl dgst -sha1 -binary \
     | openssl enc -base64
   bNXHc6ohSmeHaRHHW56BIWZJt+4=
```

6. Obsolete Syntax

Implementations of earlier drafts of this specification allowed more liberal (case insensitive) syntax and defined a different value for <scheme> than this version. The following value for <scheme> is now deprecated and SHOULD NOT be generated anymore. Serving agents SHOULD still accept it for a transition period as long as the corresponding hash function is not considered unsafe. See Section 7 for details.

```
   obs-scheme   = "sha1"
```

<obs-scheme> MUST be parsed case-insensitive.

It is important for backward compatibility that the deprecated value for <scheme> is not phased out too early. Security and compatibility concerns should be carefully weighed before choosing to remove <obs-scheme> from existing implementations (or not implementing it in new ones).

7. Security Considerations

The important properties of the hash function used for <scheme> are the preimage and second preimage resistance. A successful preimage attack would reveal the real Cancel-Key that was used to create the Cancel-Lock of the original article. A successful second preimage attack would allow to create a new, different Cancel-Key that matches a Cancel-Lock too. Both cases would break the authentication system defined in this document.

Collision resistance of the hash function used for <scheme> is less important. Finding two Cancel-Keys that match an arbitary Cancel-Lock is not helpful to break the authentication system defined in this document (if a specific article is defined as target). Only collateral damage like arbitrary deletion or spam is possible.

Currently there is no known practicable preimage and second preimage attack against the hash function SHA1. Therefore there is no hurry to replace it. The reasons why this document specifies SHA-256 (aka SHA2-256) are:

- The last draft for the authentication system defined in this document is nearly two decades old. The client side implementations are moving forward extremely slowly too (newsreaders from the last millenium are still in heavy use). What is defined today should be strong enough for at least the next decades.

- The collision resistance of SHA1 is already broken, therefore it is now obsolete for digital signatures as used in TLS. It is intended that an implementation of the authentication system defined in this document can share the same cryptographic library functions that are used for TLS.

- It is intended that the same hash function can be used for <scheme> and (as base) for the HMAC that is suggested in Section 4. See notes below for HMAC-MD5 and HMAC-SHA1.

- The SHA2 family of hash algorithms is widely supported by cryptographic libraries. In contrast, SHA3 is currently not supported by e.g. OpenSSL.

The operation HMAC(mid+sec) as suggested in Section 4 must be able to protect the local secret <sec>. The Message-ID <mid> is public (in the article header). An attacker who wants to steal/use a local secret only need to break this algorithm (regardless of <scheme>), because Cancel-Keys are explicitly published for every request to modify or delete existing articles.

Even if HMAC-MD5 and HMAC-SHA1 are not considered broken today, it is desired to have some more security margin here. Breaking <scheme> only allows to authenticate a single forged modify or delete request. With <sec> in hand it is possible to forge such requests for all articles that contain Cancel-Locks based on Cancel-Keys generated with this <sec> in the past.

8. IANA Considerations

IANA has registered the following header fields in the Permanent Message Header Field Repository, in accordance with the procedures set out in [RFC3864]:

Header field name: Cancel-Lock

Applicable protocol: netnews

Status: standard

Author/change controller: IETF

Specification document(s): This document (Section 2.1)

Header field name: Cancel-Key

Applicable protocol: netnews

Status: standard

Author/change controller: IETF

Specification document(s): This document (Section 2.2)

The Netnews Cancel-Lock hash algorithm registry will be maintained by IANA.

The registry will be available at <https://www.iana.org/assignments/netnews-cancel-lock-parameters/>.

8.1. Algorithm Name Registration Procedure

IANA will register new Cancel-Lock hash algorithm names on a First Come First Served basis, as defined in BCP 26 [RFC5226]. IANA has the right to reject obviously bogus registration requests, but will perform no review of claims made in the registration form.

Registration of a Netnews Cancel-Lock hash algorithm is requested by filling in the following template and sending it via electronic mail to IANA at <iana@iana.org>:

Subject: Registration of Netnews Cancel-Lock hash algorithm X

Netnews Cancel-Lock hash algorithm name:

Security considerations:

Published specification (recommended):

Contact for further information:

Intended usage: (One of COMMON, LIMITED USE, or OBSOLETE)

Owner/Change controller:

Note: (Any other information that the author deems relevant may be added here.)

Authors may seek community review by posting a specification of their proposed algorithm as an Internet-Draft. Netnews Cancel-Lock hash algorithms intended for widespread use should be standardized through the normal IETF process, when appropriate.

8.2. Registration of the Netnews Cancel-Lock hash algorithms

This section gives a formal definition of the Netnews Cancel-Lock hash algorithms as required by Section 8.1 for the IANA registry.

Netnews hash algorithm name: sha1

Security considerations: See Section 7 of this document

Published specification: This document

Contact for further information: Authors of this document

Intended usage: LIMITED USE

Owner/Change controller: IESG <iesg@ietf.org>

Note: This algorithm is intended for backward compatibility

Netnews hash algorithm name: sha256

Security considerations: See Section 7 of this document

Published specification: This document

Contact for further information: Authors of this document

Intended usage: COMMON

Owner/Change controller: IESG <iesg@ietf.org>

Note: This algorithm is mandatory to implement

9. References

9.1. Normative References

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

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3864] Klyne, G., Nottingham, M., and J. Mogul, "Registration Procedures for Message Header Fields", BCP 90, RFC 3864, DOI 10.17487/RFC3864, September 2004, <http://www.rfc-editor.org/info/rfc3864>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, DOI 10.17487/RFC5226, May 2008, <http://www.rfc-editor.org/info/rfc5226>.

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

[RFC5536] Murchison, K., Ed., Lindsey, C., and D. Kohn, "Netnews Article Format", RFC 5536, DOI 10.17487/RFC5536, November 2009, <http://www.rfc-editor.org/info/rfc5536>.

[RFC5537] Allbery, R., Ed. and C. Lindsey, "Netnews Architecture and Protocols", RFC 5537, DOI 10.17487/RFC5537, November 2009, <http://www.rfc-editor.org/info/rfc5537>.

[RFC7405] Kyzivat, P., "Case-Sensitive String Support in ABNF", RFC 7405, DOI 10.17487/RFC7405, December 2014, <http://www.rfc-editor.org/info/rfc7405>.

[SHA] National Institute of Standards and Technology, "Secure Hash Standard (SHS)", FIPS 180-4, DOI 10.6028/FIPS.180-4, August 2015, <http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf>.

9.2. Informative References

[RFC1321] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, DOI 10.17487/RFC1321, April 1992, <http://www.rfc-editor.org/info/rfc1321>.

[RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, DOI 10.17487/RFC2104, February 1997, <http://www.rfc-editor.org/info/rfc2104>.

[RFC3174] Eastlake 3rd, D. and P. Jones, "US Secure Hash Algorithm 1 (SHA1)", RFC 3174, DOI 10.17487/RFC3174, September 2001, <http://www.rfc-editor.org/info/rfc3174>.

[RFC4880] Callas, J., Donnerhacke, L., Finney, H., Shaw, D., and R. Thayer, "OpenPGP Message Format", RFC 4880, DOI 10.17487/RFC4880, November 2007, <http://www.rfc-editor.org/info/rfc4880>.

[RFC6151] Turner, S. and L. Chen, "Updated Security Considerations for the MD5 Message-Digest and the HMAC-MD5 Algorithms", RFC 6151, DOI 10.17487/RFC6151, March 2011, <http://www.rfc-editor.org/info/rfc6151>.

[RFC6234] Eastlake 3rd, D. and T. Hansen, "US Secure Hash Algorithms (SHA and SHA-based HMAC and HKDF)", RFC 6234, DOI 10.17487/RFC6234, May 2011, <http://www.rfc-editor.org/info/rfc6234>.

Appendix A. Acknowledgements

The author acknowledges the original author of the Cancel-Lock authentication system as documented in draft-ietf-usefor-cancel-lock: Simon Lyall. He has written the original draft and former version <https://tools.ietf.org/html/draft-ietf-usefor-cancel-lock-01> and approved the usage of his work for this document. This document is mostly based on his work and was originally intended as revision 02. It must be renamed because the USEFOR IETF WG is now closed.

The author would like to thank the following individuals for contributing their ideas and reviewing this specification: Julien Elie, Richard Kettlewell, Holger Marzen. And Peter Faust, Urs Janssen and Alfred Peters for providing statistic data about the algorithms currently in use.

Appendix B. **Document History (to be removed by RFC Editor before** publication)

B.1. Changes since -01

- Changed wording in Section 7.
- Added example for HMAC calculation in Section 5.
- Changed wording in Section 4.
- Added use cases to Section 3.2.
- Replaced wording "injecting-agent" by "injecting agent".
- Added Definition for "LOWER" in Section 2.
- Added Section 8.2.
- Added Section 8.1.
- Added new entries for header field registry in Section 8.
- Removed recommendation that moderators and injecting agents should add only one Cancel-Lock or Cancel-Key resprectively to the list in Section 3.1, Section 3.2 and Section 3.3.
- Added missing headerfield termination to Section 2.1 and Section 2.2.

- Removed definition for "code-string" from Section 2. Added stricter definition "c-lock-string" to Section 2.1. Added backward compatible definition "c-key-string" to Section 2.2.

- Use different wording in Section 2.2.

- Changed wording to reflect that an injecting agent is allowed to create Cancel-Lock headerfields in Section 2.1.

- Fixed wording and typo in Section 2.

- Added normative reference to RFC7405 because case-sensitivity is used in ABNF.

- Added reference to RFC5536 (Section 2.2) in Section 2.

- Added references to RFC4880 and RFC5537 in Section 1.

- Replaced the wordings "remove" by "cancel" and "replace" by "supersede".

- Modified header and abstract section to no longer list RFC5536 and RFC5537 as updated by this document.

B.2. Changes since -00

- Added additional note that deprecated "scheme" values should be preserved for backward compatibility as long as reasonable.

- Removed deprectated scheme "md5" (not in use anymore).

- Added descriptions how to generate "code-string" to Section 2.1 and Section 2.2.

- Removed length limitiation in ABNF of "scheme".

- Changed copyright notice to use text from TLP section 6.c.iii.

- Removed references from "abstract" section.

- Changed "SHOULD not" into "SHOULD NOT" in Section 6.

- Added line wraps to CLI commands in Section 5.

B.3. Changes since draft-ietf-usefor-cancel-lock-01

- Renamed document because the USEFOR IETF WG is now closed.

- Added more details how to check Cancel-Key header fields in Section 3.4.

- Added more details to Section 7.

- Added updated ABNF for Cancel-Lock and Cancel-Key header fields.

- Deprecated "md5" and "sha1" schemes.

- Added "sha-256" scheme.

- Reworded the abstract section and added references.

- Added note to other authentication systems to Section 1.

- Added command line check examples to Section 5.

B.4. Changes since draft-ietf-usefor-cancel-lock-00

- References to SHA-160 changed to SHA1

- "scheme" is now a case insensitive token and the number "1" has been changed to "sha1".

- Added some examples and fixed the section numbering.

- Updated 2nd paragraph on section 2.2 to make clear what exactly is being hashed and how.

- Changed paragraph 2 of 3.1 to discourage injection agents from adding the header.

- Removed the Clue-string as this complicated the scheme without adding realistic functionality

- Moderators can now add these headers under the same conditions as injection agents.

Author's Address

Michael Baeuerle
STZ Elektronik
Hofener Weg 33C
Remseck, Baden-Wuerttemberg 71686
Germany

Fax: +49 7146 999061
EMail: michael.baeuerle@stz-e.de