

Applications Area Working Group
Internet-Draft
Intended status: Informational
Expires: January 29, 2018

G. Yasutaka
Rakuten, Inc.
T. Akagiri
Regumi, Inc.
D. Kodama
BIGLOBE Inc.
K. Okada
Lepidum Co. Ltd.
July 28, 2017

DMARC verification without record definitions
draft-akagiri-dmarc-virtual-verification-02.txt

Abstract

While DMARC is a powerful architecture to protect email users from malicious email activities, its deployment is still a work in progress. To encourage further adoption of DMARC, in this draft we propose an incremental deployment procedure.

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 January 29, 2018.

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.

Table of Contents

1. Introduction	2
2. Terminology and Definitions	3
3. Overview	3
3.1. Problem Statement	3
3.2. DMARC verification without DMARC records.	5
4. The Virtual DMARC Record	6
5. Use cases	7
6. Security consideration	7
7. Privacy consideration	7
8. Discussion	7
9. Acknowledgements	8
10. Change Log	8
10.1. 01 revision	8
11. References	8
11.1. Normative References	8
11.2. Informative References	9
Authors' Addresses	9

1. Introduction

Currently, there are several email sender validation technologies such as SPF and DKIM, which are independent from each other. There is also a need for email verification frameworks to handle the email authentication results provided by those validation technologies in a unified manner. DMARC is one of the technologies that provides such a capability.

Although DMARC is an effective technology to protect email users from malicious activities such as phishing or malware, at this moment its deployment is a work in progress. A survey[ReturnPath] indicates that by the end of 2015, less than 30% of top global companies have published DMARC records. To validate incoming emails properly on the receiving MTA side, it is desirable that more domains publish DMARC records.

At the same time, the adoption of SPF and DKIM is widespread, and, especially for email receivers, it is valuable to utilize the validation results of those technologies. From the view point of email receivers, the main goals of adopting an email verification framework are:

1. To identify malicious emails and apply policies
2. To verify non-malicious emails as authorized

This draft focuses on the second goal.

In this draft, we propose a default verification for domains that do not declare DMARC resource records. This default verification is called "virtual verification(s)" in this draft. With virtual verifications, receivers are able to validate emails from domains not publishing DMARC records. This approach allows receiving MTAs to validate email senders by utilizing virtual DNS resource records populated with default values, and notify email users of the authenticity of email senders in a manner compatible with DMARC.

2. Terminology and Definitions

- o virtual DMARC record: DMARC record virtually created to verify domains that do not publish DMARC records
- o regular verification: DMARC RFC7489 verification
- o virtual verification: verification leveraging virtual DMARC records to verify emails which would potentially be "dmarc=pass"

3. Overview

3.1. Problem Statement

As described in Section 1, currently the adoption rate of DMARC remains low on the email sender side. This means that operators of receiver MTAs have little incentive to introduce DMARC verifiers into their MTAs. DMARC needs implementation on both sender and receiver sides, i.e. email senders have to publish DMARC TXT records on their DNS servers, while email receivers have to enable DMARC verification on their MTAs. Thus, the deployment of DMARC is a chicken-and-egg problem between email senders and receivers.

There are situations where emails from domains that do not have DMARC records can be treated as "dmarc=pass". In [RFC7489], the conditions for adding "dmarc=none" to Authentication-Results are:

1. No DMARC records are defined for email sender domains.
2. DMARC records are defined with "p=none" and DMARC verification did not pass.

Out of the conditions above, on the first condition there should be plenty of emails which would potentially be marked as "dmarc=pass". Those emails are from domains that have valid SPF records or DKIM signatures and validated as authorized with SPF or DKIM, while RFC5322.From fields of the emails are aligned with SPF or DKIM identifiers.

		SPF or DKIM	
		Align	Not-Align
DMARC Record	Yes	Pass	None(1) (p=none)
	No	None(2)	None(3)

Figure 1: DMARC Authentication-Results

Figure 1 illustrates the relationship between verification conditions and Authentication-Results. There are three conditions where the Authentication-Results are marked as None via DMARC regular verification. The first one is the condition where there is a DMARC record with p tag set to 'none' on a sender domain and the domain is not aligned with the email RFC5322.From field (None(1) in Figure 1).

The other two conditions are where the sender domains do not declare DMARC records on their DNS servers. In the first case, None is set when DKIM or SPF passes and the identifier aligns but no DMARC records are defined on the sender domain (None(2)). In the second case, even if valid DMARC records are declared for the sender domains, regular DMARC verification will fail (None(3)). Regular DMARC verification treats None(2) and None(3) in Figure 1 as the same "None", but the authenticity of received emails is different in each case.

According to an analysis performed by Re:gumi with a Japanese ISP([RegumiDmarcAnalysys]), the proportion of domains which can be verified as legitimate by DMARC rises to 41.9% after including the domains which potentially can be treated as legitimate (None(2)).

In this draft, we propose to explicitly mark emails which are potentially "dmarc=pass", but are not marked as such via regular DMARC verification (None(2)), as "dmarc=pass". This approach is current practice in some email receiver MTAs.

3.2. DMARC verification without DMARC records.

Figure 2 is the state diagram of the DMARC virtual verification.

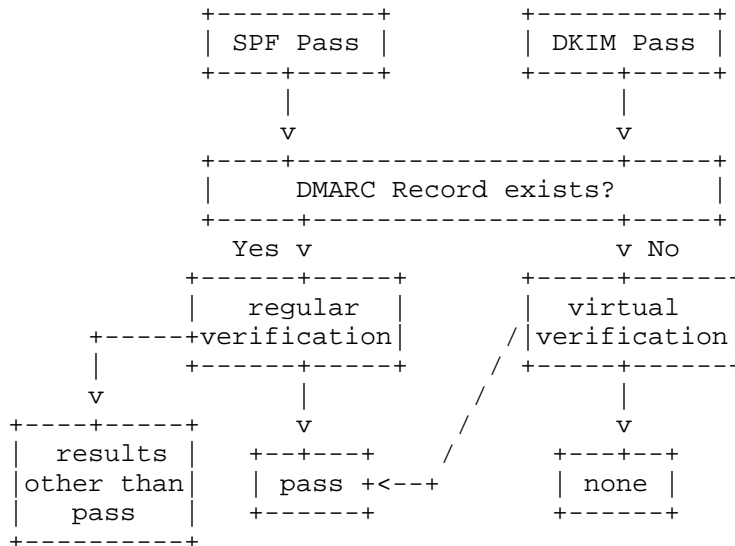


Figure 2: State Diagram

Email receivers SHOULD verify email senders using virtual DMARC records when receivers cannot find DMARC records published by the email senders. The regular DMARC verifier just sets Authentication-Results to "none" when the receiving MTA cannot find DMARC policy records for the sender domain. In contrast, when verifications of SPF or DKIM have passed, the virtual verification validates senders against virtual DMARC records populated with `_default_` values. The `_default_` values of virtual DMARC records are detailed in Section 4.

Virtual DMARC verification processes received emails in the same manner as the regular DMARC verifier does using those default values. The Authentication-Result field is set to "dmarc=pass" for email that passes virtual DMARC verification and to "dmarc=none" for email that does not.

Inserting "dmarc=pass" into Authentication-Results may cause controversy. From the point of view of email operators, it is desirable to differentiate between emails that pass virtual DMARC verifications and those that pass regular DMARC verification. However, for end users without enough expertise, simply utilizing "dmarc=pass" makes it easier to leverage the field.

4. The Virtual DMARC Record

When a email sender does not publish their DMARC record, the virtual DMARC verification validates the sender against a virtual DMARC record populated with pre-determined values. This record is referred to as "virtual DMARC record" in this draft.

The values in the virtual DMARC record are listed below (Table 1).

Tag Name	Virtual DMARC Record Value
v	DMARC1
p	none
adkim	s(strict)
aspf	s(strict)

Table 1: Values for Virtual DMARC records

These values are determined so that reasonable verifications can be performed when the sender has not published their own DMARC records.

- o version(v): DMARC1
 - * To allow the DMARC verifier to process virtual DMARC records properly, the version tag must be set as 'DMARC1'.
- o policy(p): none
 - * Received emails must be delivered to email destinations unless the email sender has published a DMARC record and has set the policy in the record to "quarantine" or "reject" explicitly. Therefore, the policy(p) must be set to "none".
- o adkim, aspf: s(strict)
 - * The default alignment modes for DKIM and SPF are "s(strict)". An example would be when a hosting provider ("example.com") enables DKIM and publishes a DKIM resource record for a customer ("aaa.example.com"). In this case, if the DKIM alignment mode (adkim) in the virtual record is "r(relaxed)", another customer of the hosting provider ("bbb.example.com") can pretend to be the owner of "aaa.example.com". To prevent this, we propose the default alignment modes (adkim and aspf)

to be "s(strict)" in the virtual DMARC record, as opposed to the default value for the regular DMARC record.

5. Use cases

Figuring out whether an email is potentially "dmarc=pass" or not allows email system operators to evaluate the authenticity of the email sender domain. DMARC evaluates the RFC5322.From, while SPF evaluates the RFC5321.Mailfrom and DKIM verifies the signature of email senders. By combining this information, DMARC verification helps to establish domain authenticity which can be used as domain reputation or whitelisting of email senders.

Microsoft Office365 employs the same technique as one mentioned in this draft ([BestGuessPass]). They append "dmarc=bestguesspass" to the Authentication-Results to indicate the authenticity of received emails to receiving MUAs.

6. Security consideration

TBD

7. Privacy consideration

TBD

8. Discussion

- o In this draft, we propose to verify the incoming email messages with the virtual records only in the strict mode. The Office365 verifier, however, verify in the relax mode. Are there any demands for the interface to configure the verification alignment modes and the authentication codes marked in the Authentication-Results?
- o Values "ruf" and "rua" for virtual records need to be decided. There would be differing opinions regarding DMARC reports. One is the opinion that reports without explicitly published DMARC records are harmful, while another one is that without reports, virtual DMARC verification can not be called DMARC. Currently, we are siding with the first opinion in this draft.
- o Name of the verification

9. Acknowledgements

The authors would like to thank engineers at the Japanese ISPs So-net, NIFTY, Biglobe and CNCI especially Mr. Koichi Abe, Mr. Daisuke Kodama and Mr. Nickolay Boyadjiev, for their technical advice and data analysis.

10. Change Log

10.1. 01 revision

- o Removed Roadmap section
- o Added focus of this document to Introduction and Problem Statement
- o Added Use cases section

11. References

11.1. Normative References

- [RFC5321] Klensin, J., "Simple Mail Transfer Protocol", RFC 5321, DOI 10.17487/RFC5321, October 2008, <<http://www.rfc-editor.org/info/rfc5321>>.
- [RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322, DOI 10.17487/RFC5322, October 2008, <<http://www.rfc-editor.org/info/rfc5322>>.
- [RFC5598] Crocker, D., "Internet Mail Architecture", RFC 5598, DOI 10.17487/RFC5598, July 2009, <<http://www.rfc-editor.org/info/rfc5598>>.
- [RFC6376] Crocker, D., Ed., Hansen, T., Ed., and M. Kucherawy, Ed., "DomainKeys Identified Mail (DKIM) Signatures", STD 76, RFC 6376, DOI 10.17487/RFC6376, September 2011, <<http://www.rfc-editor.org/info/rfc6376>>.
- [RFC7208] Kitterman, S., "Sender Policy Framework (SPF) for Authorizing Use of Domains in Email, Version 1", RFC 7208, DOI 10.17487/RFC7208, April 2014, <<http://www.rfc-editor.org/info/rfc7208>>.
- [RFC7489] Kucherawy, M., Ed. and E. Zwicky, Ed., "Domain-based Message Authentication, Reporting, and Conformance (DMARC)", RFC 7489, DOI 10.17487/RFC7489, March 2015, <<http://www.rfc-editor.org/info/rfc7489>>.

[RFC7601] Kucherawy, M., "Message Header Field for Indicating Message Authentication Status", RFC 7601, DOI 10.17487/RFC7601, August 2015, <<http://www.rfc-editor.org/info/rfc7601>>.

11.2. Informative References

[BestGuessPass]
Zink, T., "What is DMARC BestGuessPass in Office 365?", n.d., <<https://blogs.msdn.microsoft.com/tzink/2015/05/06/what-is-dmarc-bestguesspass-in-office-365/>>.

[RegumiDmarcAnalysys]
"Re:gumi Dmarc Analysis", n.d., <Under Construction>.

[ReturnPath]
Return Path, "DMARC Intelligence Report", February 2016, <<https://returnpath.com/downloads/dmarc-intelligence-report/>>.

Authors' Addresses

Genki Yasutaka
Rakuten, Inc.

Email: genki.yasutaka@rakuten.com

Takehito Akagiri
Regumi, Inc.

Email: akagiri@regumi.net

Daisuke Kodama
BIGLOBE Inc.

Email: d-kodama@biglobe.co.jp

Kouji Okada
Lepidum Co. Ltd.

Email: okd@lepidum.co.jp

DMARC Working Group
Internet-Draft
Intended status: Experimental
Expires: March 14, 2019

K. Andersen
LinkedIn
S. Blank, Ed.
ValiMail
J. Levine, Ed.
Taughannock Networks
September 10, 2018

Using Multiple Signing Algorithms with the ARC (Authenticated Received
Chain) Protocol
draft-ietf-dmarc-arc-multi-02

Abstract

The Authenticated Received Chain (ARC) protocol creates a mechanism whereby a series of handlers of an email message can conduct authentication of the email message as it passes among them on the way to its destination.

Initial development of ARC has been done with a single allowed signing algorithm, but RFC 8463 has expanded the supported algorithms. This specification defines how to extend ARC for multiple signing algorithms.

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 <https://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 March 14, 2019.

Copyright Notice

Copyright (c) 2018 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 (<https://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.

Table of Contents

1. Introduction	2
2. Overview	3
3. Definitions and Terminology	3
4. Supporting Alternate Signing Algorithms	3
5. General Approach	3
5.1. Signers	3
5.2. Validators	4
6. Phases of Algorithm Evolution	4
6.1. Introductory Period	4
6.2. Co-Existence Period	4
6.3. Deprecation Period	4
6.4. Obsolescence Period	4
7. Privacy Considerations	4
8. IANA Considerations	5
9. Security Considerations	5
10. References	5
10.1. Normative References	5
10.2. Informative References	5
Appendix A. Acknowledgements	6
Appendix B. Comments and Feedback	6
Authors' Addresses	6

1. Introduction

The Authenticated Received Chain (ARC) protocol adds a traceable chain of signatures that cover the handling of an email message through a chain of intermediary handlers.

Initial development of ARC has been done with a single allowed signing algorithm, but RFC 8463 expanded the supported algorithms. This specification defines how to extend ARC for multiple signing algorithms.

2. Overview

In order to phase in new signing algorithms, this specification identifies how signers and validators process ARC sets found in email messages.

3. Definitions and Terminology

This section defines terms used in the rest of the document.

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

Because many of the core concepts and definitions are found in [RFC5598], readers should be familiar with the contents of [RFC5598], and in particular, the potential roles of intermediaries in the delivery of email and the problems [RFC7960] created by the initial DMARC [RFC7489] .

4. Supporting Alternate Signing Algorithms

To enable multiple algorithms, all of the statements in the ARC spec which refer to "exactly one set of ARC headers per instance" need to be understood as "at least one set per instance and no more than one set per instance per algorithm".

5. General Approach

5.1. Signers

There is a separate independent signing chain for each signing algorithm. Hence, when creating an ARC signature, a signer MUST include only other signatures that use the same algorithm as the signature being created.

When signing a message with no previous ARC signatures, signers MUST sign using all supported algorithms.

A signer MUST continue the longest ARC chain(s) in a message with all algorithms that it supports. That is, if at least one of the longest chains uses an algorithm that a signer supports, the signer continues the chain(s). If none of the longest chains in a message use an algorithm supported by a signer, the signer MUST NOT extend any chains, even if a shorter chain does use a supported algorithm.

5.2. Validators

A validator MUST use the longest ARC chain(s) on the message. If a validator cannot interpret the signing algorithm on any of the longest chains, validation fails, even if a shorter chain does use a supported algorithm.

If there is more than one longest chain, the overall result reported can be that of any of the validations. The result used when extending an ARC chain MUST be the result from validating that chain.

6. Phases of Algorithm Evolution

6.1. Introductory Period

Intermediaries MUST be able to validate ARC chains built with either algorithm but MAY create ARC sets with either (or both) algorithm.

The introductory period should be at least six (6) months.

6.2. Co-Existence Period

Intermediaries MUST be able to validate ARC chains build with either algorithm and MUST create ARC sets with both algorithms. Chains ending with either algorithm may be used for the result.

6.3. Deprecation Period

ARC sets built with algorithms that are being deprecated MAY be considered valid within an ARC chain, however, intermediaries MUST NOT create additional sets with the deprecated algorithm.

The deprecation period should be at least two (2) years.

6.4. Obsolescence Period

ARC sets built with algorithms that are obsolete MUST NOT be considered valid within an ARC chain. Intermediaries MUST NOT create any sets with any obsoleted algorithm.

7. Privacy Considerations

No unique privacy considerations are introduced by this specification beyond those of the base [ARC-DRAFT] protocol.

8. IANA Considerations

No new IANA considerations are introduced by this specification.

9. Security Considerations

No new security considerations are introduced by this specification beyond those of the base [ARC-DRAFT] protocol.

10. References

10.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC5598] Crocker, D., "Internet Mail Architecture", RFC 5598, DOI 10.17487/RFC5598, July 2009, <<https://www.rfc-editor.org/info/rfc5598>>.

10.2. Informative References

- [ARC-DRAFT] Andersen, K., Long, B., and S. Jones, "Authenticated Received Chain (ARC) Protocol (I-D-16)", n.d., <<https://tools.ietf.org/html/draft-ietf-dmarc-arc-protocol-16>>.
- [RFC7489] Kucherawy, M., Ed. and E. Zwicky, Ed., "Domain-based Message Authentication, Reporting, and Conformance (DMARC)", RFC 7489, DOI 10.17487/RFC7489, March 2015, <<https://www.rfc-editor.org/info/rfc7489>>.
- [RFC7960] Martin, F., Ed., Lear, E., Ed., Draegen, Ed., T., Zwicky, E., Ed., and K. Andersen, Ed., "Interoperability Issues between Domain-based Message Authentication, Reporting, and Conformance (DMARC) and Indirect Email Flows", RFC 7960, DOI 10.17487/RFC7960, September 2016, <<https://www.rfc-editor.org/info/rfc7960>>.

10.3. URIs

- [1] <mailto:dmarc@ietf.org>

Appendix A. Acknowledgements

This draft is the work of DMARC Working Group.

Grateful appreciation is extended to the people who provided feedback through the discuss mailing list.

Appendix B. Comments and Feedback

Please address all comments, discussions, and questions to dmarc@ietf.org [1].

Authors' Addresses

Kurt Andersen
LinkedIn
1000 West Maude Ave
Sunnyvale, California 94085
US

Email: kurta@linkedin.com

Seth Blank (editor)
ValiMail
Montgomery
San Francisco, California
US

Email: seth@valimail.com

John Levine (editor)
Taughannock Networks
PO Box 727
Trumansburg, New York
US

Email: standards@taugh.com

DMARC Working Group
Internet-Draft
Intended status: Informational
Expires: October 25, 2018

S. Jones
DMARC.org
K. Andersen
LinkedIn
J. Rae-Grant
Google
T. Adams, Ed.
Paypal
April 23, 2018

Recommended Usage of the Authenticated Received Chain (ARC)
draft-ietf-dmarc-arc-usage-05

Abstract

The Authentication Received Chain (ARC) provides a means to preserve email authentication results and verify the identity of email message handlers, each of which participates by inserting certain header fields before passing the message on. But the specification does not indicate how intermediaries and receivers should interpret or utilize ARC. This document will provide guidance in these areas.

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 <https://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 October 25, 2018.

Copyright Notice

Copyright (c) 2018 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 (<https://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.

Table of Contents

1.	Introduction	3
2.	How does ARC work?	3
3.	Guidance for Receivers/Validators	4
3.1.	What is the significance of an intact ARC chain?	4
3.2.	What exactly is an "intact" ARC chain?	4
3.3.	What is the significance of an invalid ("broken") ARC chain?	5
3.4.	What does the absence of an ARC chain in a message mean?	5
3.5.	What reasonable conclusions can you draw based upon seeing lots of mail with ARC chains?	6
3.6.	What if none of the intermediaries have been seen previously?	6
3.7.	What about ARC chains where some intermediaries are known and others are not?	6
3.8.	What should message handlers do when they detect malicious content in messages where ARC is present?	7
3.9.	What feedback does a sender or domain owner get about ARC when it is applied to their messages?	7
3.10.	What prevents a malicious actor from removing the ARC header fields, altering the content, and creating a new ARC chain?	7
4.	Guidance for Intermediaries	8
4.1.	What is an Intermediary under ARC?	8
4.2.	What are the minimum requirements for an ARC Intermediary?	8
4.2.1.	More specifically a participating ARC intermediary must do the following:	8
4.3.	Should every MTA be an ARC participant?	9
4.4.	What should an intermediary do in the case of an invalid or "broken" ARC chain?	9
4.5.	What should I do in the case where there is no ARC chain present in a message?	9
4.6.	How could ARC affect my reputation as an intermediary?	9
4.7.	What can I do to influence my reputation as an intermediary?	10
5.	Guidance for Originators	10
5.1.	Where can I find out more information?	10
5.2.	How/where can I test interoperabililty for my implementation?	10

5.3. How can ARC impact my email?	10
5.4. How can ARC impact my reputation as a message sender? . .	11
5.5. Can I tell intermediaries not to use ARC?	11
6. References	11
6.1. Normative References	11
6.2. Informative References	12
6.3. URIs	12
Appendix A. GLOSSARY	12
Appendix B. References	15
Appendix C. Acknowledgements	15
Appendix D. Comments and Feedback	15
Authors' Addresses	15

1. Introduction

[ARC] is intended to be used primarily by intermediaries, or message handlers - those parties who may forward or resend messages, with or without alterations, such that they will no longer pass the SPF, DKIM, and/or [RFC7489] authentication mechanisms. In such cases ARC may provide the final message recipient with useful information about the original sender.

2. How does ARC work?

Consider a mailing list as an example, where the message submitter's domain publishes a DMARC policy other than "p=none". The message is received, a prefix is added to the RFC5322.Subject header field, some text is appended to the message body, and the message is sent to list members with the original RFC5322.From address intact. In this case SPF may pass because the mailing list operator uses their own domain in the RFC5321.MailFrom header field, but this domain will not match the RFC5322.From address, thus the DMARC SPF result cannot be a "pass." Any DKIM signature from the message submitter's domain will be broken as the message body has been altered (and if included in the signature, the RFC5322.Subject header field). Again, the DMARC DKIM result cannot be a "pass." And if the mailing list operator inserted an Authentication-Results header field it was most likely stripped and/or replaced by the next message receiver.

If the mailing list implemented ARC, it would record the contents of the Authentication-Results header field in the ARC-Authentication-Results header field. It would then create an ARC-Message-Signature header field, which includes a cryptographic signature of the message itself, and then an ARC-Seal header field, which includes a cryptographic signature of a few key message header fields - including the other ARC header fields.

Any subsequent system participating in ARC that was not performing final delivery of the message within its ADMD boundaries would also generate and insert ARC header fields whose signatures cover all ARC header fields inserted into the message by previous message handlers. Thus the information from any previous ARC participants, including the ARC-Authentication-Results header field from the mailing list operator, would be signed at each ADMD that handled the message.

When the message reaches the final receiving system, the SPF and DKIM results will not satisfy the DMARC policy for the message author's domain. However if the receiving system implements ARC then it can check for and validate an ARC chain and verify that the contents of the ARC-Authentication-Results header field were conveyed intact from the mailing list operator. At that point the receiving system might choose to use those authentication results in the decision of whether or not to deliver the message, even though it failed to pass the usual authentication checks.

3. Guidance for Receivers/Validators

3.1. What is the significance of an intact ARC chain?

An intact ARC chain conveys authentication results like SPF and DKIM as observed by the first ARC participant. In cases where the message no longer produces passing results for DKIM, SPF, or DMARC but an intact ARC chain is present, the message receiver may choose to use the contents of the ARC-Authentication-Results header field in determining how to handle the message.

3.2. What exactly is an "intact" ARC chain?

Note that not all ADMDs will implement ARC, and receivers will see messages where one or more non-participating ADMDs handled a message before, after, or in between participating ADMDs.

An intact ARC chain is one where the ARC header fields that are present can be validated, and in particular the ARC-Message-Signature header field from the last ARC participant can still be validated. This shows that, whether another ADMD handled the message after the last ARC participant or not, the portions of the message covered by that signature were not altered. If any non-participating ADMDs handled the message between ARC intermediaries but did not alter the message in a way that invalidated the most recent ARC-Message-Signature present at that time, the chain would still be considered intact by the next ARC participant, and recorded as such in the ARC-Seal header field they insert.

Message receivers may make local policy decisions about whether to use the contents of the ARC-Authentication-Results header field in cases where a message no longer passes DKIM, DMARC, and/or SPF checks. Whether an ARC chain is intact can be used to inform that local policy decision.

So for example one message receiver may decide that, for messages with an intact ARC chain where a DMARC evaluation does not pass, but the ARC-Authentication-Results header field indicates a DKIM pass was reported by the first ARC intermediary that matches the domain in the RFC5322.From header field, it will override a DMARC "p=reject" policy. Another message receiver may decide to do so for intact ARC chains where the ARC-Authentication-Results header field indicates an SPF pass. A third message receiver may use very different criteria, according to their requirements, while a fourth may choose not to take ARC information into account at all.

3.3. What is the significance of an invalid ("broken") ARC chain?

An ARC chain is broken if the signatures in the ARC-Seal header fields cannot be verified or if the most recent AMS can not be verified. For example the remote server delivering the message to the local ADMD is not reflected in any ARC header fields, perhaps because they have not implemented ARC, but they modified the message such that ARC and DKIM signatures already in the message were invalidated.

In case of a broken ARC chain, the message should be treated the same as if there was no ARC chain at all. For example, a message that fails under DMARC and has an invalid ARC chain would be subject to that DMARC policy, which may cause it to be quarantined or rejected.

Email transit can produce broken signatures for a wide variety of benign reasons. This includes possibly breaking one or more ARC signatures. Therefore, receivers need to be wary of ascribing motive to such breakage although patterns of common behaviour may provide some basis for adjusting local policy decisions.

3.4. What does the absence of an ARC chain in a message mean?

The absence of an ARC chain means nothing. ARC is intended to allow a participating message handler to preserve certain authentication results when a message is being forwarded and/or modified such that the final recipient can evaluate this information. If they are absent, there is nothing extra that ARC requires the final recipient to do.

- 3.5. What reasonable conclusions can you draw based upon seeing lots of mail with ARC chains?

With sufficient history, ARC can be used to augment DMARC authentication policy (i.e. a message could fail DMARC, but validated ARC information and therefore could be considered as validly authenticated as reported by the first ARC participant).

If the validator does content analysis and reputation tracking, the ARC participants in a message can be credited or discredited for good or bad content. By analyzing different ARC chains involved in "bad" messages, a validator might identify malicious participating intermediaries.

With a valid chain and good reputations for all ARC participants, receivers may choose to apply a "local policy override" to the DMARC policy assertion for the domain authentication evaluation, depending on the ARC-Authentication-Results header field value. Normal content analysis should never be skipped.

- 3.6. What if none of the intermediaries have been seen previously?

This has no impact on the operation of ARC, as ARC is not a reputation system. ARC conveys the results of other authentication mechanisms such that the participating message handlers can be positively identified. Final message recipients may or may not choose to examine these results when messages fail other authentication checks. They are more likely to override, say, a failing DMARC result in the presence of an intact ARC chain where the participating ARC message handlers have been observed to not convey "bad" content in the past, and the initial ARC participant indicates the message they received had passed authentication checks.

- 3.7. What about ARC chains where some intermediaries are known and others are not?

Validators may choose to build reputation models for ARC message handlers they have observed. Generally speaking it is more feasible to accrue positive reputation to intermediaries when they consistently send messages that are evaluated positively in terms of content and ARC chains. When messages are received with ARC chains that are not intact, it is very difficult identify which intermediaries may have manipulated the message or injected bad content.

- 3.8. What should message handlers do when they detect malicious content in messages where ARC is present?

Message handlers should do what they normally do when they detect malicious content in a message - hopefully that means quarantining or discarding the message. ARC information should never make malicious content acceptable.

In such cases it is difficult to determine where the malicious content may have been injected. What ARC can do in such cases is verify that a given intermediary or message handler did in fact handle the message as indicated in the header fields. In such cases a message recipient who maintains a reputation system about email senders may wish to incorporate this information as an additional factor in the score for the intermediaries and sender in question. However reputation systems are very complex, and usually unique to those organizations operating them, and therefore beyond the scope of this document.

- 3.9. What feedback does a sender or domain owner get about ARC when it is applied to their messages?

ARC itself does not include any mechanism for feedback or reporting. It does however recommend that message receiving systems that use ARC to augment their delivery decisions, who use DMARC and decide to deliver a message because of ARC information, should include a notation to that effect in their normal DMARC reports. These notations would be easily identifiable by report processors, so that senders and domain owners can see where ARC is being used to augment the deliverability of their messages.

- 3.10. What prevents a malicious actor from removing the ARC header fields, altering the content, and creating a new ARC chain?

ARC does not prevent a malicious actor from doing this. Nor does it prevent a malicious actor from removing all but the first ADMD's ARC header fields and altering the message, eliminating intervening participants from the ARC chain. Or similar variations.

A valid ARC chain does not provide any automatic benefit. With an intact ARC chain, the final message recipient may choose to use the contents of the ARC-Authentication-Results header field in determining how to handle the message. The decision to use the ARC-Authentication-Results header field is dependent on evaluation of those ARC intermediaries.

In the first case, the bad actor has succeeded in manipulating the message but they have attached a verifiable signature identifying

themselves. While not an ideal situation, it is something they are already able to do without ARC involved, but now a signature linked to the domain responsible for the manipulation is present.

Additionally in the second case it is possible some negative reputational impact might accrue to the first ARC participant left in place until more messages reveal the pattern of activity by the bad actor. But again, a bad actor can similarly manipulate a sequence of RFC5322.Received header fields today without ARC, but with ARC that bad actor has verifiably identified themselves.

4. Guidance for Intermediaries

4.1. What is an Intermediary under ARC?

In the context of ARC, an Intermediary is typically an Administrative Management Domain [RFC5598] that is receiving a message, potentially manipulating or altering it, and then passing it on to another ADMD for delivery. Common examples of Intermediaries are mailing lists, alumni or professional email address providers that forward messages such as universities or professional organizations, et cetera.

4.2. What are the minimum requirements for an ARC Intermediary?

A participating ARC intermediary must validate the ARC chain on a message it receives, if one is present. It then attaches its own ARC seal and signature, including an indication if the chain failed to validate upon receipt.

4.2.1. More specifically a participating ARC intermediary must do the following:

1. Validate that the ARC chain, if one is already present in the message, is intact and well-formed.
2. Validate that the most recent sender matches the last entry in the ARC chain (if present).
3. Validate that the most recent sender's DKIM signature is attached, and matches the reference to it in the ARC chain (if present).
4. Generate a new ARC Signature and add it to the message according to the ARC specification.
5. Generate a new ARC Seal and add it to the message according to the ARC specification.

4.3. Should every MTA be an ARC participant?

Generally speaking, ARC is designed to operate at the ADMD level. When a message is first received by an ADMD, the traditional authentication results should be captured and preserved - this could be the common case of creating an Authentication-Results header field. But when it is determined that the message is being sent on outside of that ADMD, that is when the ADMD should add itself to the ARC chain - before sending the message outside of the ADMD.

Some organizations may operate multiple ADMDs, with more or less independence between them. While they should make a determination based on their specific circumstances, it may be useful and appropriate to have one or both ADMDs be ARC participants.

4.4. What should an intermediary do in the case of an invalid or "broken" ARC chain?

In general terms, a participating ARC intermediary will note that an ARC chain was present and invalid, or broken, when it attaches its own ARC seal and signature. However the fact that the ARC chain was invalid should have no impact on whether and how the message is delivered.

4.5. What should I do in the case where there is no ARC chain present in a message?

A participating ARC intermediary receiving a message with no ARC chain, and which will be delivered outside its ADMD, should start an ARC chain according to the ARC specification. This will include capturing the normal email authentication results for the intermediary (SPF, DKIM, DMARC, etc), which will be conveyed as part of the ARC chain.

4.6. How could ARC affect my reputation as an intermediary?

Message receivers often operate reputation systems, which build a behavioral profile of various message handlers and intermediaries. The presence or absence of ARC is yet another data point that may be used as an input to such reputation systems. Messages deemed to have good content may provide a positive signal for the intermediaries that handled it, while messages with bad content may provide a negative signal for the those intermediaries. Intact and valid ARC elements may amplify or attenuate such signals, depending on the circumstances.

Reputation systems are complex and usually specific to a given message receiver, and a meaningful discussion of such a broad topic is beyond the scope of this document.

4.7. What can I do to influence my reputation as an intermediary?

Today it is extremely simple for a malicious actor to construct a message that includes your identity as an intermediary, even though you never handled the message. It is possible that an intermediary implementing ARC on all traffic it handles might receive some reputational benefit by making it easier to detect when their involvement in conveying bad traffic has been "forged."

As mentioned previously reputation systems are very complex and usually specific to a given message receiver, and a meaningful discussion of such a broad topic is beyond the scope of this document.

5. Guidance for Originators

5.1. Where can I find out more information?

Please join the arc-discuss list at arc-discuss@dmARC.org [1][mailto:arc-discuss@dmARC.org].

To discuss the IETF spec itself, please join the dmARC working group at [<https://datatracker.ietf.org/wg/dmARC>].

5.2. How/where can I test interoperability for my implementation?

The arc-discuss list is the best place to stay in touch with work in progress.

5.3. How can ARC impact my email?

Prior to ARC, certain DMARC policies on a domain would cause messages using those domains in the RFC5322.From field, and which pass through certain kinds of intermediaries (mailing lists, forwarding services), to fail authentication checks at the message receiver. As a result these messages might not be delivered to the intended recipient.

ARC seeks to provide these so-called "indirect mailflows" with a means to preserve email authentication results as recorded by participating intermediaries. Message receivers may accept validated ARC information to supplement the information that DMARC provides, potentially deciding to deliver the message even though a DMARC check did not pass.

The net result for domain owners and senders is that ARC may allow messages routed through participating ARC intermediaries to be delivered, even though those messages would not have been delivered in the absence of ARC.

5.4. How can ARC impact my reputation as a message sender?

Message receivers often operate reputation systems, which build a behavioral profile of various message senders (and perhaps intermediaries). The presence or absence of ARC is yet another data point that may be used as an input to such reputation systems. Messages deemed to have good content may provide a positive signal for the sending domain and the intermediaries that handled it, while messages with bad content may provide a negative signal for the sending domain and the intermediaries that handled it. Intact and valid ARC elements may amplify or attenuate such signals, depending on the circumstances.

Reputation systems are complex and usually specific to a given message receiver, and a meaningful discussion of such a broad topic is beyond the scope of this document.

5.5. Can I tell intermediaries not to use ARC?

At present there is no way for a message sender to request that intermediaries not employ ARC.

6. References

6.1. Normative References

- [RFC5321] Klensin, J., "Simple Mail Transfer Protocol", RFC 5321, DOI 10.17487/RFC5321, October 2008, <<https://www.rfc-editor.org/info/rfc5321>>.
- [RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322, DOI 10.17487/RFC5322, October 2008, <<https://www.rfc-editor.org/info/rfc5322>>.
- [RFC5598] Crocker, D., "Internet Mail Architecture", RFC 5598, DOI 10.17487/RFC5598, July 2009, <<https://www.rfc-editor.org/info/rfc5598>>.
- [RFC6377] Kucherawy, M., "DomainKeys Identified Mail (DKIM) and Mailing Lists", BCP 167, RFC 6377, DOI 10.17487/RFC6377, September 2011, <<https://www.rfc-editor.org/info/rfc6377>>.

- [RFC7601] Kucherawy, M., "Message Header Field for Indicating Message Authentication Status", RFC 7601, DOI 10.17487/RFC7601, August 2015, <<https://www.rfc-editor.org/info/rfc7601>>.

6.2. Informative References

- [ARC] Andersen, K., Rae-Grant, J., Long, B., and S. Jones, "Authenticated Received Chain (ARC) Protocol", December 2017, <<https://tools.ietf.org/html/draft-ietf-dmarc-arc-protocol-10>>.
- [ENHANCED-STATUS] "IANA SMTP Enhanced Status Codes", n.d., <<http://www.iana.org/assignments/smtp-enhanced-status-codes/smtp-enhanced-status-codes.xhtml>>.
- [OAR] Chew, M. and M. Kucherawy, "Original-Authentication-Results Header Field", February 2012, <<https://tools.ietf.org/html/draft-kucherawy-original-authres-00>>.
- [RFC7489] Kucherawy, M., Ed. and E. Zwicky, Ed., "Domain-based Message Authentication, Reporting, and Conformance (DMARC)", RFC 7489, DOI 10.17487/RFC7489, March 2015, <<https://www.rfc-editor.org/info/rfc7489>>.
- [RFC7960] Martin, F., Ed., Lear, E., Ed., Draegen, Ed., T., Zwicky, E., Ed., and K. Andersen, Ed., "Interoperability Issues between Domain-based Message Authentication, Reporting, and Conformance (DMARC) and Indirect Email Flows", RFC 7960, DOI 10.17487/RFC7960, September 2016, <<https://www.rfc-editor.org/info/rfc7960>>.

6.3. URIs

- [1] <mailto:arc-discuss@dmARC.org>

Appendix A. GLOSSARY

ADMD Administrative Management Domain as used in [RFC5598] and similar references refers to a single entity operating one or more computers within one or more domain names under said entity's control. One example might be a small company with a single server, handling email for that company's domain. Another example might be a large university, operating many servers that fulfill different roles, all handling email for several different domains representing parts of the university.

ARC ARC is an acronym: Authentication Results Chain - see also [ARC]

ARC-Seal An [RFC5322] message header field formed in compliance with the ARC specification. It includes certain content from all prior ARC participants, if there are any.

ARC-Message-Signature (also abbreviated as "AMS") An [RFC5322] message header field formed in compliance with the [ARC] specification. It includes certain content about the message as it was received and manipulated by the intermediary who inserted it.

ARC-Authentication-Results (also abbreviated as "AAR") An [RFC5322] message header field formed in compliance with the [ARC] specification. It includes certain content about the message as it was received by the intermediary.

Authentication Results Chain (ARC) A system that allows a Message Receiver to identify Intermediaries or Message Handlers who have conveyed a particular message. For more information see the Abstract of this document, or refer to [ARC].

Domain Naming System Block List (DNSBL) This is a system widely used in email filtering services whereby information about the past activity of a set of hosts or domains indicates that messages should not be accepted from them, or at least should be subject to greater scrutiny before being accepted. Common examples would be SpamCop, Spamhaus.org, SORBS, etc.

Email Service Provider (ESP) An Email Service Provider is typically a vendor or partner firm that sends mail on behalf of another company. They may use email addresses in Internet domains belonging to the client or partner firm in various [RFC5321] fields or [RFC5322] message header fields of the messages they send on their behalf.

Intermediary In the context of [ARC], an Intermediary is typically an Administrative Management Domain (per [RFC5598]) that is receiving a message, potentially manipulating or altering it, and then passing it on to another ADMD for delivery. Also see [RFC7960] for more information and discussion. Common examples of Intermediaries are mailing lists, alumni or professional email address providers like universities or professional organizations, et cetera.

Mail/Message Transfer Agent (MTA) This refers to software that sends and receives email messages across a network with other MTAs.

Often run on dedicated servers, common examples are Exim, Microsoft Exchange, Postfix, and Sendmail.

Mailflow A group of messages that share features in common. Typical examples would be all messages sent by a given Message Sender to a Message Receiver, related to a particular announcement, a given mailing list, et cetera.

Malicious Actor A Malicious Actor is a party, often an Intermediary, that will take actions that seek to exploit or defraud the ultimate recipient of the message, or subvert the network controls and infrastructure of the Message Receiver. Typical examples would be a spammer who forges content or attributes of a message in order to evade anti-spam measures, or an entity that adds an attachment containing a virus to a message.

Message Handler A Message Handler is another name for an Intermediary.

Message Receiver In the transmission of an email message from one ADMD to another, this is the organization receiving the message on behalf of the intended recipient or end user. The Message Receiver may do this because the intended recipient is an employee or member of the organization, or because the end user utilizes email services provided by the Message Receiver (Comcast, GMail, Yahoo, QQ, et cetera).

Message Sender In the transmission of an email message from one ADMD to another, this is the organization sending the message on behalf of the Originator or end user.

Originator This refers to the author of a given email message. In different contexts it may refer to the end-user writing the message, or the ADMD providing email services to that end-user.

Reputation In the larger context of email hygiene - blocking spam and malicious messages - reputation generally refers to a wide variety of techniques and mechanisms whereby a message receiver uses the past actions of a sending host or domain to influence the handling of messages received from them in the future. One of the classic examples would be a Spamhaus-style DNSBL, where individual IP addresses will be blocked from sending messages because they've been identified as being bad actors. Very large message receivers may build and maintain their own reputation systems of this kind, whereas other organizations might choose to use commercial products or free services.

Reputation Service Provider A Reputation Service Provider would be a source of reputation information about a message sender. In this context, the DNSBL services offered by Spamhaus would allow them to be referred to as an RPS. Many spam and virus filtering vendors incorporate similar functionality into their services.

Request For Comment (RFC) RFCs are memoranda that "contain technical and organizational notes about the Internet." Created and managed by the Internet Engineering Task Force (IETF), they are de facto standards for various methods of communicating or collaborating over the Internet.

RFC5321 - Simple Mail Transfer Protocol This document describes the protocol used to transfer email messages between Message Transfer Agents (MTA) over a network. Link: [RFC5321]

RFC5322 - Internet Message Format This document describes the format of Internet email messages, including both the header fields within the message and various types of content within the message body. Link: [RFC5322]

Validator A Message Receiver that attempts to validate the ARC chain in a message.

Appendix B. References

Appendix C. Acknowledgements

This draft is the work of OAR-Dev Group.

The authors thanks the entire OAR-Dev group for the ongoing help, innumerable diagrams and discussions from all the participants, especially: Alex Brotman, Brandon Long, Dave Crocker, Elizabeth Zwicky, Franck Martin, Greg Colburn, J. Trent Adams, John Rae-Grant, Mike Hammer, Mike Jones, Steve Jones, Terry Zink, Tim Draegen.

Appendix D. Comments and Feedback

Please address all comments, discussions, and questions to the dmarc working group at [<https://datatracker.ietf.org/wg/dmarc>].

Authors' Addresses

Steven Jones
DMARC.org

Email: smj@crash.com

Kurt Andersen
LinkedIn
2029 Stierlin Ct.
Mountain View, California 94043
USA

Email: kurta@linkedin.com

John Rae-Grant
Google

Email: johnrg@google.com

J. Trent Adams (editor)
Paypal

Email: trent.adams@paypal.com

DMARC Working Group
Internet-Draft
Obsoletes: 7601 (if approved)
Intended status: Standards Track
Expires: February 23, 2019

M. Kucherawy
August 22, 2018

Message Header Field for Indicating Message Authentication Status
draft-ietf-dmarc-rfc7601bis-03

Abstract

This document specifies a message header field called Authentication-Results for use with electronic mail messages to indicate the results of message authentication efforts. Any receiver-side software, such as mail filters or Mail User Agents (MUAs), can use this header field to relay that information in a convenient and meaningful way to users or to make sorting and filtering decisions.

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 February 23, 2019.

Copyright Notice

Copyright (c) 2018 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.

Table of Contents

1.	Introduction	4
1.1.	Purpose	5
1.2.	Trust Boundary	6
1.3.	Processing Scope	6
1.4.	Requirements	6
1.5.	Definitions	7
1.5.1.	Key Words	7
1.5.2.	Internationalized Email	7
1.5.3.	Security	7
1.5.4.	Email Architecture	8
1.5.5.	Other Terms	9
1.6.	Trust Environment	9
2.	Definition and Format of the Header Field	9
2.1.	General Description	9
2.2.	Formal Definition	10
2.3.	Property Types (ptypes) and Properties	13
2.4.	The "policy" ptype	14
2.5.	Authentication Identifier Field	14
2.6.	Version Tokens	16
2.7.	Defined Methods and Result Values	16
2.7.1.	DKIM and DomainKeys	16
2.7.2.	SPF and Sender ID	18
2.7.3.	"iprev"	19
2.7.4.	SMTP AUTH	20
2.7.5.	Other Registered Codes	21
2.7.6.	Extension Methods	21
2.7.7.	Extension Result Codes	22
3.	The "iprev" Authentication Method	23
4.	Adding the Header Field to a Message	24
4.1.	Header Field Position and Interpretation	25
4.2.	Local Policy Enforcement	26
5.	Removing Existing Header Fields	27
6.	IANA Considerations	28
6.1.	The Authentication-Results Header Field	28
6.2.	"Email Authentication Methods" Registry Description	28
6.3.	"Email Authentication Methods" Registry Update	28
6.3.1.	'header.a' for DKIM	29
6.3.2.	'header.s' for DKIM	29
6.4.	"Email Authentication Property Types" Registry	29
6.5.	"Email Authentication Result Names" Description	29
6.6.	"Email Authentication Result Names" Update	30
7.	Security Considerations	30

7.1.	Forged Header Fields	30
7.2.	Misleading Results	32
7.3.	Header Field Position	32
7.4.	Reverse IP Query Denial-of-Service Attacks	32
7.5.	Mitigation of Backscatter	32
7.6.	Internal MTA Lists	32
7.7.	Attacks against Authentication Methods	33
7.8.	Intentionally Malformed Header Fields	33
7.9.	Compromised Internal Hosts	33
7.10.	Encapsulated Instances	33
7.11.	Reverse Mapping	34
8.	References	34
8.1.	Normative References	34
8.2.	Informative References	36
Appendix A.	Legacy MUAs	38
Appendix B.	Authentication-Results Examples	38
B.1.	Trivial Case; Header Field Not Present	39
B.2.	Nearly Trivial Case; Service Provided, but No Authentication Done	39
B.3.	Service Provided, Authentication Done	40
B.4.	Service Provided, Several Authentications Done, Single MTA	41
B.5.	Service Provided, Several Authentications Done, Different MTAs	42
B.6.	Service Provided, Multi-tiered Authentication Done	44
B.7.	Comment-Heavy Example	45
Appendix C.	Operational Considerations about Message Authentication	46
Appendix D.	Changes Since RFC7601	47
Appendix E.	Acknowledgments	47
Author's Address	48

1. Introduction

This document describes a header field called Authentication-Results for electronic mail messages that presents the results of a message authentication effort in a machine-readable format. The intent of the header field is to create a place to collect such data when message authentication mechanisms are in use so that a Mail User Agent (MUA) and downstream filters can make filtering decisions and/or provide a recommendation to the user as to the validity of the message's origin and possibly the safety and integrity of its content.

End users are not expected to be direct consumers of this header field. This header field is intended for consumption by programs that will then use such data or render it in a human-usable form.

This document specifies the format of this header field and discusses the implications of its presence or absence. However, it does not discuss how the data contained in the header field ought to be used, such as what filtering decisions are appropriate or how an MUA might render those results, as these are local policy and/or user interface design questions that are not appropriate for this document.

At the time of publication of this document, the following are published email authentication methods:

- o Author Domain Signing Practices ([ADSP]) (Historic)
- o SMTP Service Extension for Authentication ([AUTH])
- o DomainKeys Identified Mail Signatures ([DKIM])
- o Domain-based Message Authentication, Reporting and Conformance ([DMARC])
- o Sender Policy Framework ([SPF])
- o reverse IP address name validation ("iprev", defined in Section 3)
- o Require-Recipient-Valid-Since Header Field and SMTP Service Extension ([RRVS])
- o S/MIME Signature Verification ([SMIME-REG])
- o Vouch By Reference ([VBR])
- o DomainKeys ([DOMAINKEYS]) (Historic)

- o Sender ID ([SENDERID]) (Experimental)

There exist registries for tokens used within this header field that refer to the specifications listed above. Section 6 describes the registries and their contents and specifies the process by which entries are added or updated. It also updates the existing contents to match the current states of these specifications.

The goal of this work is to give current and future authentication schemes a common framework within which to deliver their results to downstream agents and discourage the creation of unique header fields for each.

Although SPF defined a header field called "Received-SPF" and the historic DomainKeys defined one called "DomainKey-Status" for this purpose, those header fields are specific to the conveyance of their respective results only and thus are insufficient to satisfy the requirements enumerated below. In addition, many SPF implementations have adopted the header field specified here at least as an option, and DomainKeys has been obsoleted by DKIM.

1.1. Purpose

The header field defined in this document is expected to serve several purposes:

1. Convey the results of various message authentication checks, which are applied by upstream filters and Mail Transfer Agents (MTAs) and then passed to MUAs and downstream filters within the same "trust domain". Such agents might wish to render those results to end users or to use those data to apply more or less stringent content checks based on authentication results;
2. Provide a common location within a message for this data;
3. Create an extensible framework for reporting new authentication methods as they emerge.

In particular, the mere presence of this header field does not mean its contents are valid. Rather, the header field is reporting assertions made by one or more authentication schemes applied somewhere upstream. For an MUA or downstream filter to treat the assertions as actually valid, there must be an assessment of the trust relationship among such agents, the validating MTA, and the mechanism for conveying the information.

1.2. Trust Boundary

This document makes several references to the "trust boundary" of an administrative management domain (ADMD). Given the diversity among existing mail environments, a precise definition of this term isn't possible.

Simply put, a transfer from the producer of the header field to the consumer must occur within a context that permits the consumer to treat assertions by the producer as being reliable and accurate (trustworthy). How this trust is obtained is outside the scope of this document. It is entirely a local matter.

Thus, this document defines a "trust boundary" as the delineation between "external" and "internal" entities. Services that are internal -- within the trust boundary -- are provided by the ADMD's infrastructure for its users. Those that are external are outside of the authority of the ADMD. By this definition, hosts that are within a trust boundary are subject to the ADMD's authority and policies, independent of their physical placement or their physical operation. For example, a host within a trust boundary might actually be operated by a remote service provider and reside physically within its data center.

It is possible for a message to be evaluated inside a trust boundary but then depart and re-enter the trust boundary. An example might be a forwarded message such as a message/rfc822 attachment (see Multipurpose Internet Mail Extensions [MIME]) or one that is part of a multipart/digest. The details reported by this field cannot be trusted in that case. Thus, this field found within one of those media types is typically ignored.

1.3. Processing Scope

The content of this header field is meant to convey to message consumers that authentication work on the message was already done within its trust boundary, and those results are being presented. It is not intended to provide message parameters to consumers so that they can perform authentication protocols on their own.

1.4. Requirements

This document establishes no new requirements on existing protocols or servers.

In particular, this document establishes no requirement on MTAs to reject or filter arriving messages that do not pass authentication checks. The data conveyed by the specified header field's contents

are for the information of MUAs and filters and are to be used at their discretion.

1.5. Definitions

This section defines various terms used throughout this document.

1.5.1. Key Words

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

1.5.2. Internationalized Email

In this document, there are references to messages formatted to support Email Address Internationalization (EAI). Reference material for this can be found in [RFC6530], [RFC6531], and [RFC6532]. Generally speaking, these documents allow UTF-8 in most places that free-form text can be found and U-labels where domain names can be used, and this document extends Authentication-Results accordingly.

1.5.3. Security

"Guidelines for Writing RFC Text on Security Considerations" ([SECURITY]) discusses authentication and authorization and the conflation of the two concepts. The use of those terms within the context of recent message security work has given rise to slightly different definitions, and this document reflects those current usages, as follows:

- o "Authorization" is the establishment of permission to use a resource or represent an identity. In this context, authorization indicates that a message from a particular ADMD arrived via a route the ADMD has explicitly approved.
- o "Authentication" is the assertion of validity of a piece of data about a message (such as the sender's identity) or the message in its entirety.

As examples: SPF and Sender ID are authorization mechanisms in that they express a result that shows whether or not the ADMD that apparently sent the message has explicitly authorized the connecting Simple Mail Transfer Protocol ([SMTP]) client to relay messages on its behalf, but they do not actually validate any other property of the message itself. By contrast, DKIM is agnostic as to the routing of a message but uses cryptographic signatures to authenticate agents, assign (some) responsibility for the message (which implies

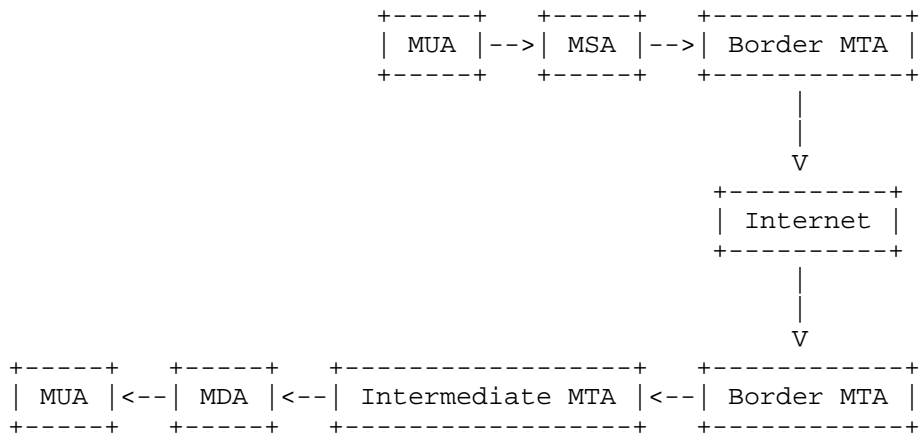
authorization), and ensure that the listed portions of the message were not modified in transit. Since the signatures are not tied to SMTP connections, they can be added by either the ADMD of origin, intermediate ADMDs (such as a mailing list server), other handling agents, or any combination.

Rather than create a separate header field for each class of solution, this specification groups them both into a single header field.

1.5.4. Email Architecture

- o A "border MTA" is an MTA that acts as a gateway between the general Internet and the users within an organizational boundary. (See also Section 1.2.)
- o A "delivery MTA" (or Mail Delivery Agent or MDA) is an MTA that actually enacts delivery of a message to a user's inbox or other final delivery.
- o An "intermediate MTA" is any MTA that is not a delivery MTA and is also not the first MTA to handle the message.

The following diagram illustrates the flow of mail among these defined components. See Internet Mail Architecture [EMAIL-ARCH] for further discussion on general email system architecture, which includes detailed descriptions of these components, and Appendix C of this document for discussion about the common aspects of email authentication in current environments.



Generally, it is assumed that the work of applying message authentication schemes takes place at a border MTA or a delivery MTA.

This specification is written with that assumption in mind. However, there are some sites at which the entire mail infrastructure consists of a single host. In such cases, such terms as "border MTA" and "delivery MTA" might well apply to the same machine or even the very same agent. It is also possible that some message authentication tests could take place on an intermediate MTA. Although this document doesn't specifically describe such cases, they are not meant to be excluded.

1.5.5. Other Terms

In this document, the term "producer" refers to any component that adds this header field to messages it is handling, and "consumer" refers to any component that identifies, extracts, and parses the header field to use as part of a handling decision.

1.6. Trust Environment

This header field permits one or more message validation mechanisms to communicate output to one or more separate assessment mechanisms. These mechanisms operate within a unified trust boundary that defines an Administrative Management Domain (ADMD). An ADMD contains one or more entities that perform validation and generate the header field and one or more that consume it for some type of assessment. The field often contains no integrity or validation mechanism of its own, so its presence must be trusted implicitly. Hence, valid use of the header field requires removing any occurrences of it that claim to be associated with the ADMD when the message enters the ADMD. This ensures that later occurrences have been added within the trust boundary of the ADMD.

The authserv-id token defined in Section 2.2 can be used to reference an entire ADMD or a specific validation engine within an ADMD. Although the labeling scheme is left as an operational choice, some guidance for selecting a token is provided in later sections of this document.

2. Definition and Format of the Header Field

This section gives a general overview of the format of the header field being defined and then provides a formal specification.

2.1. General Description

The header field specified here is called Authentication-Results. It is a Structured Header Field as defined in Internet Message Format ([MAIL]), and thus all of the related definitions in that document

apply.

This header field is added at the top of the message as it transits MTAs that do authentication checks, so some idea of how far away the checks were done can be inferred. It is therefore considered to be a trace field as defined in [MAIL], and thus all of the related definitions in that document apply.

The value of the header field (after removing comments) consists of an authentication identifier, an optional version, and then a series of statements and supporting data. The statements are of the form "method=result" and indicate which authentication method(s) were applied and their respective results. For each such statement, the supporting data can include a "reason" string and one or more "property=value" statements indicating which message properties were evaluated to reach that conclusion.

The header field can appear more than once in a single message, more than one result can be represented in a single header field, or a combination of these can be applied.

2.2. Formal Definition

Formally, the header field is specified as follows using Augmented Backus-Naur Form ([ABNF]):

```
authres-header-field = "Authentication-Results:" authres-payload
authres-payload = [CFWS] authserv-id
                  [ CFWS authres-version ]
                  ( no-result / 1*resinfo ) [CFWS] CRLF
authserv-id = value
              ; see below for a description of this element
authres-version = 1*DIGIT [CFWS]
                  ; indicates which version of this specification is in use;
                  ; this specification is version "1", and the absence of a
                  ; version implies this version of the specification
no-result = [CFWS] ";" [CFWS] "none"
            ; the special case of "none" is used to indicate that no
            ; message authentication was performed
resinfo = [CFWS] ";" methodspec [ CFWS reasonspec ]
          *( CFWS propspec )
methodspec = [CFWS] method [CFWS] "=" [CFWS] result
```

```
        ; indicates which authentication method was evaluated
        ; and what its output was

reasonspec = "reason" [CFWS] "=" [CFWS] value
        ; a free-form comment on the reason the given result
        ; was returned

propspec = ptype [CFWS] "." [CFWS] property [CFWS] "=" pvalue
        ; an indication of which properties of the message
        ; were evaluated by the authentication scheme being
        ; applied to yield the reported result

method = Keyword [ [CFWS] "/" [CFWS] method-version ]
        ; a method indicates which method's result is
        ; represented by "result", and is one of the methods
        ; explicitly defined as valid in this document
        ; or is an extension method as defined below

method-version = 1*DIGIT [CFWS]
        ; indicates which version of the method specification is
        ; in use, corresponding to the matching entry in the IANA
        ; "Email Authentication Methods" registry; a value of "1"
        ; is assumed if this version string is absent

result = Keyword
        ; indicates the results of the attempt to authenticate
        ; the message; see below for details

ptype = Keyword
        ; indicates whether the property being evaluated was
        ; a parameter to an [SMTP] command, was a value taken
        ; from a message header field, was some property of
        ; the message body, or was some other property evaluated by
        ; the receiving MTA; expected to be one of the "property
        ; types" explicitly defined as valid, or an extension
        ; ptype, as defined below

property = special-smtp-verb / Keyword
        ; indicates more specifically than "ptype" what the
        ; source of the evaluated property is; the exact meaning
        ; is specific to the method whose result is being reported
        ; and is defined more clearly below

special-smtp-verb = "mailfrom" / "rcptto"
        ; special cases of [SMTP] commands that are made up
        ; of multiple words

pvalue = [CFWS] ( value / [ [ local-part ] "@" ] domain-name )
```

[CFWS]
; the value extracted from the message property defined
; by the "ptype.property" construction

"local-part" is defined in Section 3.4.1 of [MAIL], as modified by [RFC6531].

"CFWS" is defined in Section 3.2.2 of [MAIL].

"domain-name" is as defined in Section 3.5 of [DKIM] where the "d=" tag is defined, with "sub-domain" as modified by [RFC6531].

"Keyword" is defined in Section 4.1.2 of [SMTP]. It is further constrained by the necessity of being registered in the IANA registry relevant to the context in which it is used. See Section 2.7, Section 2.3, and Section 6.

"value" is as defined in Section 5.1 of [MIME].

See Section 2.5 for a description of the authserv-id element.

If the value portion of a "pvalue" construction identifies something intended to be an email identity, then it MUST use the right hand portion of that ABNF definition.

The list of commands eligible for use with the "smtp" ptype can be found in Section 4.1 of [SMTP].

The "prospec" may be omitted if, for example, the method was unable to extract any properties to do its evaluation yet still has a result to report. It may also be omitted if the agent generating this result wishes not to reveal such properties to downstream agents.

Where an SMTP command name is being reported as a "property", the agent generating the header field represents that command by converting it to lowercase and dropping any spaces (e.g., "MAIL FROM" becomes "mailfrom", "RCPT TO" becomes "rcptto", etc.).

A "ptype" value of "policy" indicates a policy decision about the message not specific to a property of the message that could be extracted. See Section 2.4 for details.

Examples of complete messages using this header field can be found in Appendix B.

2.3. Property Types (ptypes) and Properties

The "ptype" in the ABNF above indicates the general type of property being described by the result being reported, upon which the reported result was based. Coupled with the "property", which is more specific, they indicate from where the reported data were extracted. This can include part of the message header or body, some part of the SMTP session, a secondary output of an authentication method (apart from its pure result), or some other aspect of the message's handling.

Combinations of ptypes and properties are registered and described in the "Email Authentication Methods" registry, coupled with the authentication methods with which they are used. This is further described in Section 6.

Legal values of "ptype" are as defined in the IANA "Email Authentication Property Types" registry, created by [RFC7410]. The initial values and what they typically indicate are as follows, based on [RFC7001]:

body: Information that was extracted from the body of the message.

This might be an arbitrary string of bytes, a hash of a string of bytes, a Uniform Resource Identifier, or some other content of interest. The "property" is an indication of where within the message body the extracted content was found, and can indicate an offset, identify a MIME part, etc.

header: Indicates information that was extracted from the header of the message. This might be the value of a header field or some portion of a header field. The "property" gives a more precise indication of the place in the header from which the extraction took place.

policy: A local policy mechanism was applied that augments or overrides the result returned by the authentication mechanism. (See Section 2.4.)

smtp: Indicates information that was extracted from an SMTP command that was used to relay the message. The "property" indicates which SMTP command included the extracted content as a parameter.

Results reported using unknown ptypes MUST NOT be used in making handling decisions. They can be safely ignored by consumers.

Entries in the "Email Authentication Methods" registry can define properties that deviate from these definitions when appropriate. Such deviations need to be clear in the registry and/or in the

defining document. See Section 2.7.1 for an example.

2.4. The "policy" ptype

A special ptype value of "policy" is also defined. This ptype is provided to indicate that some local policy mechanism was applied that augments or even replaces (i.e., overrides) the result returned by the authentication mechanism. The property and value in this case identify the local policy that was applied and the result it returned.

For example, a DKIM signature is not required to include the Subject header field in the set of fields that are signed. An ADMD receiving such a message might decide that such a signature is unacceptable, even if it passes, because the content of the Subject header field could be altered post-signing without invalidating the signature. Such an ADMD could replace the DKIM "pass" result with a "policy" result and then also include the following in the corresponding Authentication-Result field:

```
... dkim=policy policy.dkim-rules=unsigned-subject ...
```

In this case, the property is "dkim-rules", indicating some local check by that name took place and that check returned a result of "unsigned-subject". These are arbitrary names selected by (and presumably used within) the ADMD making use of them, so they are not normally registered with IANA or otherwise specified apart from setting syntax restrictions that allow for easy parsing within the rest of the header field.

This ptype existed in the original specification for this header field, but without a complete description or example of intended use. As a result, it has not seen any practical use to date that matches its intended purpose. These added details are provided to guide implementers toward proper use.

2.5. Authentication Identifier Field

Every Authentication-Results header field has an authentication service identifier field (authserv-id above). Specifically, this is any string intended to identify the authentication service within the ADMD that conducted authentication checks on the message. This identifier is intended to be machine-readable and not necessarily meaningful to users.

Note that in an EAI-formatted message, this identifier may be expressed in UTF-8.

Since agents consuming this field will use this identifier to determine whether its contents are of interest (and are safe to use), the uniqueness of the identifier **MUST** be guaranteed by the ADMD that generates it and **MUST** pertain to that ADMD. MUAs or downstream filters **SHOULD** use this identifier to determine whether or not the data contained in an Authentication-Results header field ought to be used or ignored.

For simplicity and scalability, the authentication service identifier **SHOULD** be a common token used throughout the ADMD. Common practice is to use the DNS domain name used by or within that ADMD, sometimes called the "organizational domain", but this is not strictly necessary.

For tracing and debugging purposes, the authentication identifier can instead be the specific hostname of the MTA performing the authentication check whose result is being reported. Moreover, some implementations define a substructure to the identifier; such structures are outside of the scope of this specification.

Note, however, that using a local, relative identifier like a flat hostname, rather than a hierarchical and globally unique ADMD identifier like a DNS domain name, makes configuration more difficult for large sites. The hierarchical identifier permits aggregating related, trusted systems together under a single, parent identifier, which in turn permits assessing the trust relationship with a single reference. The alternative is a flat namespace requiring individually listing each trusted system. Since consumers will use the identifier to determine whether to use the contents of the header field:

- o Changes to the identifier impose a large, centralized administrative burden.
- o Ongoing administrative changes require constantly updating this centralized table, making it difficult to ensure that an MUA or downstream filter will have access to accurate information for assessing the usability of the header field's content. In particular, consumers of the header field will need to know not only the current identifier(s) in use but previous ones as well to account for delivery latency or later re-assessment of the header field's contents.

Examples of valid authentication identifiers are "example.com", "mail.example.org", "msl.newyork.example.com", and "example-auth".

2.6. Version Tokens

The grammar above provides for the optional inclusion of versions on both the header field itself (attached to the authserv-id token) and on each of the methods being reported. The method version refers to the method itself, which is specified in the documents describing those methods, while the authserv-id version refers to this document and thus the syntax of this header field.

The purpose of including these is to avoid misinterpretation of the results. That is, if a parser finds a version after an authserv-id that it does not explicitly know, it can immediately discontinue trying to parse since what follows might not be in an expected format. For a method version, the parser SHOULD ignore a method result if the version is not supported in case the semantics of the result have a different meaning than what is expected. For example, if a hypothetical DKIM version 2 yielded a "pass" result for different reasons than version 1 does, a consumer of this field might not want to use the altered semantics. Allowing versions in the syntax is a way to indicate this and let the consumer of the header field decide.

2.7. Defined Methods and Result Values

Each individual authentication method returns one of a set of specific result values. The subsections below provide references to the documents defining the authentication methods specifically supported by this document, and their corresponding result values. Verifiers SHOULD use these values as described below. New methods not specified in this document, but intended to be supported by the header field defined here, MUST include a similar result table either in their defining documents or in supplementary ones.

2.7.1. DKIM and DomainKeys

DKIM is represented by the "dkim" method and is defined in [DKIM]. DomainKeys is defined in [DOMAINKEYS] and is represented by the "domainkeys" method.

Section 3.8 of [DOMAINKEYS] enumerates some possible results of a DomainKeys evaluation. Those results are not used when generating this header field; rather, the results returned are listed below.

A signature is "acceptable to the ADMD" if it passes local policy checks (or there are no specific local policy checks). For example, an ADMD policy might require that the signature(s) on the message be added using the DNS domain present in the From header field of the message, thus making third-party signatures unacceptable even if they

verify.

Both DKIM and DomainKeys use the same result set, as follows:

none: The message was not signed.

pass: The message was signed, the signature or signatures were acceptable to the ADMD, and the signature(s) passed verification tests.

fail: The message was signed and the signature or signatures were acceptable to the ADMD, but they failed the verification test(s).

policy: The message was signed, but some aspect of the signature or signatures was not acceptable to the ADMD.

neutral: The message was signed, but the signature or signatures contained syntax errors or were not otherwise able to be processed. This result is also used for other failures not covered elsewhere in this list.

temperror: The message could not be verified due to some error that is likely transient in nature, such as a temporary inability to retrieve a public key. A later attempt may produce a final result.

permerror: The message could not be verified due to some error that is unrecoverable, such as a required header field being absent. A later attempt is unlikely to produce a final result.

DKIM results are reported using a ptype of "header". The property, however, represents one of the tags found in the DKIM-Signature header field rather than a distinct header field. For example, the ptype-property combination "header.d" refers to the content of the "d" (signing domain) tag from within the signature header field, and not a distinct header field called "d".

Note that in an EAI-formatted message, the values of the "d" and "i" properties can be expressed in UTF-8.

In addition to previous registrations, this document registers the DKIM tag "a" (cryptographic algorithm used to sign the message) as a reportable property. This can be used to aid receivers during post-verification processing. In particular, [RFC8301] obsoleted use of the "rsa-sha1" algorithm in DKIM, so it is important to be able to distinguish such signatures from those using preferred algorithms.

The ability to report different DKIM results for a message with

multiple signatures is described in [RFC6008].

[DKIM] advises that if a message fails verification, it is to be treated as an unsigned message. A report of "fail" here permits the receiver of the report to decide how to handle the failure. A report of "neutral" or "none" preempts that choice, ensuring the message will be treated as if it had not been signed.

Section 3.1 of [DOMAINKEYS] describes a process by which the sending address of the message is determined. DomainKeys results are thus reported along with the signing domain name, the sending address of the message, and the name of the header field from which the latter was extracted. This means that a DomainKeys result includes a ptype-property combination of "header.d", plus one of "header.from" and "header.sender". The sending address extracted from the header is included with any [MAIL]-style comments removed; moreover, the local-part of the address and the "@" character are removed if it has not been authenticated in some way.

2.7.2. SPF and Sender ID

SPF and Sender ID use the "spf" and "sender-id" method names, respectively. The result values for SPF are defined in Section 2.6 of [SPF], and those definitions are included here by reference:

Code	Meaning
none	[RFC7208], Section 2.6.1
pass	[RFC7208], Section 2.6.3
fail	[RFC7208], Section 2.6.4
softfail	[RFC7208], Section 2.6.5
policy	RFC 7601, Section 2.4
neutral	[RFC7208], Section 2.6.2
temperror	[RFC7208], Section 2.6.6
permerror	[RFC7208], Section 2.6.7

These result codes are used in the context of this specification to reflect the result returned by the component conducting SPF evaluation.

For SPF, the ptype used is "smtp", and the property is either "mailfrom" or "helo", since those values are the ones SPF can evaluate. (If the SMTP client issued the EHLO command instead of HELO, the property used is "helo".)

Note that in an EAI-formatted message, the "mailfrom" value can be expressed in UTF-8.

The "sender-id" method is described in [SENDERID]. For this method, the ptype used is "header" and the property will be the name of the header field from which the Purported Responsible Address (see [PRA]) was extracted -- namely, one of "Resent-Sender", "Resent-From", "Sender", or "From".

The results for Sender ID are listed and described in Section 4.2 of [SENDERID], but for the purposes of this specification, the SPF definitions enumerated above are used instead. Also, [SENDERID] specifies result codes that use mixed case, but they are typically used all lowercase in this context.

For both methods, an additional result of "policy" is defined, which means the client was authorized to inject or relay mail on behalf of the sender's DNS domain according to the authentication method's algorithm, but local policy dictates that the result is unacceptable. For example, "policy" might be used if SPF returns a "pass" result, but a local policy check matches the sending DNS domain to one found in an explicit list of unacceptable DNS domains (e.g., spammers).

If the retrieved sender policies used to evaluate SPF and Sender ID do not contain explicit provisions for authenticating the local-part (see Section 3.4.1 of [MAIL]) of an address, the "pvalue" reported along with results for these mechanisms SHOULD NOT include the local-part or the following "@" character.

2.7.3. "iprev"

The result values used by the "iprev" method, defined in Section 3, are as follows:

pass: The DNS evaluation succeeded, i.e., the "reverse" and "forward" lookup results were returned and were in agreement.

fail: The DNS evaluation failed. In particular, the "reverse" and "forward" lookups each produced results, but they were not in agreement, or the "forward" query completed but produced no result, e.g., a DNS RCODE of 3, commonly known as NXDOMAIN, or an RCODE of 0 (NOERROR) in a reply containing no answers, was returned.

temperror: The DNS evaluation could not be completed due to some error that is likely transient in nature, such as a temporary DNS error, e.g., a DNS RCODE of 2, commonly known as SERVFAIL, or other error condition resulted. A later attempt may produce a final result.

permerror: The DNS evaluation could not be completed because no PTR data are published for the connecting IP address, e.g., a DNS RCODE of 3, commonly known as NXDOMAIN, or an RCODE of 0 (NOERROR) in a reply containing no answers, was returned. This prevented completion of the evaluation. A later attempt is unlikely to produce a final result.

There is no "none" for this method since any TCP connection delivering email has an IP address associated with it, so some kind of evaluation will always be possible.

The result is reported using a ptype of "policy" (as this is not part of any established protocol) and a property of "iprev".

For discussion of the format of DNS replies, see "Domain Names - Implementation and Specification" ([DNS]).

2.7.4. SMTP AUTH

SMTP AUTH (defined in [AUTH]) is represented by the "auth" method. Its result values are as follows:

none: SMTP authentication was not attempted.

pass: The SMTP client authenticated to the server reporting the result using the protocol described in [AUTH].

fail: The SMTP client attempted to authenticate to the server using the protocol described in [AUTH] but was not successful (such as providing a valid identity but an incorrect password).

temperror: The SMTP client attempted to authenticate using the protocol described in [AUTH] but was not able to complete the attempt due to some error that is likely transient in nature, such as a temporary directory service lookup error. A later attempt may produce a final result.

permerror: The SMTP client attempted to authenticate using the protocol described in [AUTH] but was not able to complete the attempt due to some error that is likely not transient in nature, such as a permanent directory service lookup error. A later attempt is not likely to produce a final result.

The result of AUTH is reported using a ptype of "smtp" and a property of either:

- o "auth", in which case the value is the authorization identity generated by the exchange initiated by the AUTH command; or
- o "mailfrom", in which case the value is the mailbox identified by the AUTH parameter used with the MAIL FROM command. Note that in an EAI-formatted message, these values can be expressed in UTF-8.

If both identities are available, both can be reported. For example, consider this command issued by a client that has completed session authentication with the AUTH command resulting in an authorized identity of "client@c.example":

```
MAIL FROM:<alice@a.example> AUTH=<bob@b.example>
```

This could result in a "resinfo" construction like so:

```
; auth=pass smtp.auth=client@c.example smtp.mailfrom=bob@b.example
```

Note that in all cases other than "pass", the message was sent by an unauthenticated client. All non-"pass" cases SHOULD thus be treated as equivalent with respect to this method.

2.7.5. Other Registered Codes

Result codes were also registered in other RFCs as follows:

- o Vouch By Reference (in [AR-VBR], represented by "vbr");
- o Authorized Third-Party Signatures (in [ATPS], represented by "dkim-atps");
- o Author Domain Signing Practices (in [ADSP], represented by "dkim-adsp");
- o Require-Recipient-Valid-Since (in [RRVS], represented by "rrvs");
- o S/MIME (in [SMIME-REG], represented by "smime").

Note that "vbr.mv" and "vbr.md", which are already registered, are permitted to be UTF-8 in an EAI-formatted message.

2.7.6. Extension Methods

Additional authentication method identifiers (extension methods) may be defined in the future by later revisions or extensions to this

specification. These method identifiers are registered with the Internet Assigned Numbers Authority (IANA) and, preferably, published in an RFC. See Section 6 for further details.

Extension methods can be defined for the following reasons:

1. To allow additional information from new authentication systems to be communicated to MUAs or downstream filters. The names of such identifiers ought to reflect the name of the method being defined but ought not be needlessly long.
2. To allow the creation of "sub-identifiers" that indicate different levels of authentication and differentiate between their relative strengths, e.g., "auth1-weak" and "auth1-strong".

Authentication method implementers are encouraged to provide adequate information, via message header field comments if necessary, to allow an MUA developer to understand or relay ancillary details of authentication results. For example, if it might be of interest to relay what data was used to perform an evaluation, such information could be relayed as a comment in the header field, such as:

```
Authentication-Results: example.com;  
                        foo=pass bar.baz=blob (2 of 3 tests OK)
```

Experimental method identifiers **MUST** only be used within ADMs that have explicitly consented to use them. These method identifiers and the parameters associated with them are not documented in RFCs. Therefore, they are subject to change at any time and not suitable for production use. Any MTA, MUA, or downstream filter intended for production use **SHOULD** ignore or delete any Authentication-Results header field that includes an experimental (unknown) method identifier.

2.7.7. Extension Result Codes

Additional result codes (extension results) might be defined in the future by later revisions or extensions to this specification. Non-experimental result codes **MUST** be registered with the Internet Assigned Numbers Authority (IANA) and preferably published in an RFC. See Section 6 for further details.

Experimental results **MUST** only be used within ADMs that have explicitly consented to use them. These results and the parameters associated with them are not formally documented. Therefore, they are subject to change at any time and not suitable for production use. Any MTA, MUA, or downstream filter intended for production use **SHOULD** ignore or delete any Authentication-Results header field that

includes an extension result.

3. The "iprev" Authentication Method

This section defines an additional authentication method called "iprev".

"iprev" is an attempt to verify that a client appears to be valid based on some DNS queries, which is to say that the IP address is explicitly associated with a domain name. Upon receiving a session initiation of some kind from a client, the IP address of the client peer is queried for matching names (i.e., a number-to-name translation, also known as a "reverse lookup" or a "PTR" record query). Once that result is acquired, a lookup of each of the names (i.e., a name-to-number translation, or an "A" or "AAAA" record query) thus retrieved is done. The response to this second check will typically result in at least one mapping back to the client's IP address.

Expressed as an algorithm: If the client peer's IP address is I, the list of names to which I maps (after a "PTR" query) is the set N, and the union of IP addresses to which each member of N maps (after corresponding "A" and "AAAA" queries) is L, then this test is successful if I is an element of L.

Often an MTA receiving a connection that fails this test will simply reject the connection using the enhanced status code defined in [AUTH-ESC]. If an operator instead wishes to make this information available to downstream agents as a factor in handling decisions, it records a result in accordance with Section 2.7.3.

The response to a PTR query could contain multiple names. To prevent heavy DNS loads, agents performing these queries MUST be implemented such that the number of names evaluated by generation of corresponding A or AAAA queries is limited so as not to be unduly taxing to the DNS infrastructure, though it MAY be configurable by an administrator. As an example, Section 4.6.4 of [SPF] chose a limit of 10 for its implementation of this algorithm.

"DNS Extensions to Support IP Version 6" ([DNS-IP6]) discusses the query formats for the IPv6 case.

There is some contention regarding the wisdom and reliability of this test. For example, in some regions, it can be difficult for this test ever to pass because the practice of arranging to match the forward and reverse DNS is infrequently observed. Therefore, the precise implementation details of how a verifier performs an "iprev"

test are not specified here. The verifier MAY report a successful or failed "iprev" test at its discretion having done some kind of check of the validity of the connection's identity using DNS. It is incumbent upon an agent making use of the reported "iprev" result to understand what exactly that particular verifier is attempting to report.

Extensive discussion of reverse DNS mapping and its implications can be found in "Considerations for the use of DNS Reverse Mapping" ([DNSOP-REVERSE]). In particular, it recommends that applications avoid using this test as a means of authentication or security. Its presence in this document is not an endorsement but is merely acknowledgment that the method remains common and provides the means to relay the results of that test.

4. Adding the Header Field to a Message

This specification makes no attempt to evaluate the relative strengths of various message authentication methods that may become available. The methods listed are an order-independent set; their sequence does not indicate relative strength or importance of one method over another. Instead, the MUA or downstream filter consuming this header field is to interpret the result of each method based on its own knowledge of what that method evaluates.

Each "method" MUST refer to an authentication method declared in the IANA registry or an extension method as described in Section 2.7.6, and each "result" MUST refer to a result code declared in the IANA registry or an extension result code as defined in Section 2.7.7. See Section 6 for further information about the registered methods and result codes.

An MTA compliant with this specification adds this header field (after performing one or more message authentication tests) to indicate which MTA or ADMD performed the test, which test was applied, and what the result was. If an MTA applies more than one such test, it adds this header field either once per test or once indicating all of the results. An MTA MUST NOT add a result to an existing header field.

An MTA MAY add this header field containing only the authentication identifier portion and the "none" token (see Section 2.2) to indicate explicitly that no message authentication schemes were applied prior to delivery of this message.

An MTA adding this header field has to take steps to identify it as legitimate to the MUAs or downstream filters that will ultimately

consume its content. One process to do so is described in Section 5. Further measures may be necessary in some environments. Some possible solutions are enumerated in Section 7.1. This document does not mandate any specific solution to this issue as each environment has its own facilities and limitations.

Most known message authentication methods focus on a particular identifier to evaluate. SPF and Sender ID differ in that they can yield a result based on more than one identifier; specifically, SPF can evaluate the RFC5321.HELO parameter or the RFC5321.MailFrom parameter, and Sender ID can evaluate the RFC5321.MailFrom parameter or the Purported Responsible Address (PRA) identity. When generating this field to report those results, only the parameter that yielded the result is included.

For MTAs that add this header field, adding header fields in order (at the top), per Section 3.6 of [MAIL], is particularly important. Moreover, this header field SHOULD be inserted above any other trace header fields such MTAs might prepend. This placement allows easy detection of header fields that can be trusted.

End users making direct use of this header field might inadvertently trust information that has not been properly vetted. If, for example, a basic SPF result were to be relayed that claims an authenticated addr-spec, the local-part of that addr-spec has actually not been authenticated. Thus, an MTA adding this header field SHOULD NOT include any data that has not been authenticated by the method(s) being applied. Moreover, MUAs SHOULD NOT render to users such information if it is presented by a method known not to authenticate it.

4.1. Header Field Position and Interpretation

In order to ensure non-ambiguous results and avoid the impact of false header fields, MUAs and downstream filters SHOULD NOT interpret this header field unless specifically configured to do so by the user or administrator. That is, this interpretation should not be "on by default". Naturally then, users or administrators ought not activate such a feature unless (1) they are certain the header field will be validly added by an agent within the ADMD that accepts the mail that is ultimately read by the MUA, and (2) instances of the header field that appear to originate within the ADMD but are actually added by foreign MTAs will be removed before delivery.

Furthermore, MUAs and downstream filters SHOULD NOT interpret this header field unless the authentication service identifier of the header field is used within the ADMD as configured by the user or administrator.

MUAs and downstream filters MUST ignore any result reported using a "result" not specified in the IANA "Result Code" registry or a "ptype" not listed in the "Email Authentication Property Types" registry for such values as defined in Section 6. Moreover, such agents MUST ignore a result indicated for any "method" they do not specifically support.

An MUA SHOULD NOT reveal these results to end users, absent careful human factors design considerations and testing, for the presentation of trust-related materials. For example, an attacker could register example.com (note the digit "1" (one)) and send signed mail to intended victims; a verifier would detect that the signature was valid and report a "pass" even though it's clear the DNS domain name was intended to mislead. See Section 7.2 for further discussion.

As stated in Section 2.1, this header field MUST be treated as though it were a trace header field as defined in Section 3.6.7 of [MAIL] and hence MUST NOT be reordered and MUST be prepended to the message, so that there is generally some indication upon delivery of where in the chain of handling MTAs the message authentication was done.

Note that there are a few message handlers that are only capable of appending new header fields to a message. Strictly speaking, these handlers are not compliant with this specification. They can still add the header field to carry authentication details, but any signal about where in the handling chain the work was done may be lost. Consumers SHOULD be designed such that this can be tolerated, especially from a producer known to have this limitation.

MUAs SHOULD ignore instances of this header field discovered within message/rfc822 MIME attachments.

Further discussion of these topics can be found in Section 7 below.

4.2. Local Policy Enforcement

Some sites have a local policy that considers any particular authentication policy's non-recoverable failure results (typically "fail" or similar) as justification for rejecting the message. In such cases, the border MTA SHOULD issue an SMTP rejection response to the message, rather than adding this header field and allowing the message to proceed toward delivery. This is more desirable than allowing the message to reach an internal host's MTA or spam filter, thus possibly generating a local rejection such as a Delivery Status Notification (DSN) [DSN] to a forged originator. Such generated rejections are colloquially known as "backscatter".

The same MAY also be done for local policy decisions overriding the

results of the authentication methods (e.g., the "policy" result codes described in Section 2.7).

Such rejections at the SMTP protocol level are not possible if local policy is enforced at the MUA and not the MTA.

5. Removing Existing Header Fields

To mitigate the impact of forged header fields, any MTA conforming to this specification MUST delete any discovered instance of this header field that claims, by virtue of its authentication service identifier, to have been added within its trust boundary but that did not come directly from another trusted MTA. For example, an MTA for example.com receiving a message MUST delete or otherwise obscure any instance of this header field bearing an authentication service identifier indicating that the header field was added within example.com prior to adding its own header fields. This could mean each MTA will have to be equipped with a list of internal MTAs known to be compliant (and hence trustworthy).

For messages that are EAI-formatted messages, this test is done after converting A-labels into U-labels.

For simplicity and maximum security, a border MTA could remove all instances of this header field on mail crossing into its trust boundary. However, this may conflict with the desire to access authentication results performed by trusted external service providers. It may also invalidate signed messages whose signatures cover external instances of this header field. A more robust border MTA could allow a specific list of authenticating MTAs whose information is to be admitted, removing the header field originating from all others.

As stated in Section 1.2, a formal definition of "trust boundary" is deliberately not made here. It is entirely possible that a border MTA for example.com will explicitly trust authentication results asserted by upstream host example.net even though they exist in completely disjoint administrative boundaries. In that case, the border MTA MAY elect not to delete those results; moreover, the upstream host doing some authentication work could apply a signing technology such as [DKIM] on its own results to assure downstream hosts of their authenticity. An example of this is provided in Appendix B.

Similarly, in the case of messages signed using [DKIM] or other message-signing methods that sign header fields, this removal action could invalidate one or more signatures on the message if they

covered the header field to be removed. This behavior can be desirable since there's little value in validating the signature on a message with forged header fields. However, signing agents MAY therefore elect to omit these header fields from signing to avoid this situation.

An MTA SHOULD remove any instance of this header field bearing a version (express or implied) that it does not support. However, an MTA MUST remove such a header field if the [SMTP] connection relaying the message is not from a trusted internal MTA. This means the MTA needs to be able to understand versions of this header field at least as late as the ones understood by the MUAs or other consumers within its ADMD.

6. IANA Considerations

IANA has registered the defined header field and created registries as described below. These registry actions were originally defined by [RFC5451] and updated by [RFC6577] and [RFC7001]. The created registries were further updated in [RFC7601] to make them more complete.

Each is listed below, though generally they are not changed by this document.

6.1. The Authentication-Results Header Field

The Authentication-Results header field was added to the IANA "Permanent Message Header Field Names" registry, per the procedure found in [IANA-HEADERS]. That entry will be updated to reference this document. The following is the registration template:

```
Header field name: Authentication-Results
Applicable protocol: mail ([MAIL])
Status: Standard
Author/Change controller: IETF
Specification document(s): [this document]
Related information: none
```

6.2. "Email Authentication Methods" Registry Description

No changes are made to the description of this registry.

6.3. "Email Authentication Methods" Registry Update

The following two entries are added.

6.3.1. 'header.a' for DKIM

Method: dkim

Definition: [this document]

ptype: header

property: a

Description: value of signature "a" tag

Status: active

Version: 1

6.3.2. 'header.s' for DKIM

"header.s" for DKIM:

Method: dkim

Definition: [this document]

ptype: header

property: s

Description: value of signature "s" tag

Status: active

Version: 1

6.4. "Email Authentication Property Types" Registry

[RFC7410] created the "Email Authentication Property Types" registry.

No changes are made to the description of this registry. However, it should be noted that Section 2.3 contains slightly different language than prior versions of this document, allowing a broader space from which to extract meaningful identifiers and report them through this mechanism.

6.5. "Email Authentication Result Names" Description

No changes are made to the description of this registry.

6.6. "Email Authentication Result Names" Update

No changes are made to entries in this registry.

7. Security Considerations

The following security considerations apply when adding or processing the Authentication-Results header field:

7.1. Forged Header Fields

An MUA or filter that accesses a mailbox whose messages are handled by a non-conformant MTA, and understands Authentication-Results header fields, could potentially make false conclusions based on forged header fields. A malicious user or agent could forge a header field using the DNS domain of a receiving ADMD as the authserv-id token in the value of the header field and, with the rest of the value, claim that the message was properly authenticated. The non-conformant MTA would fail to strip the forged header field, and the MUA could inappropriately trust it.

For this reason, it is best not to have processing of the Authentication-Results header field enabled by default; instead, it should be ignored, at least for the purposes of enacting filtering decisions, unless specifically enabled by the user or administrator after verifying that the border MTA is compliant. It is acceptable to have an MUA aware of this specification but have an explicit list of hostnames whose Authentication-Results header fields are trustworthy; however, this list should initially be empty.

Proposed alternative solutions to this problem were made some time ago and are listed below. To date, they have not been developed due to lack of demand but are documented here should the information be useful at some point in the future:

1. Possibly the simplest is a digital signature protecting the header field, such as using [DKIM], that can be verified by an MUA by using a posted public key. Although one of the main purposes of this document is to relieve the burden of doing message authentication work at the MUA, this only requires that the MUA learn a single authentication scheme even if a number of them are in use at the border MTA. Note that [DKIM] requires that the From header field be signed, although in this application, the signing agent (a trusted MTA) likely cannot authenticate that value, so the fact that it is signed should be ignored. Where the authserv-id is the ADMD's domain name, the authserv-id matching this valid internal signature's "d=" DKIM

value is sufficient.

2. Another would be a means to interrogate the MTA that added the header field to see if it is actually providing any message authentication services and saw the message in question, but this isn't especially palatable given the work required to craft and implement such a scheme.
3. Yet another might be a method to interrogate the internal MTAs that apparently handled the message (based on Received header fields) to determine whether any of them conform to Section 5 of this memo. This, too, has potentially high barriers to entry.
4. Extensions to [IMAP], [SMTP], and [POP3] could be defined to allow an MUA or filtering agent to acquire the authserv-id in use within an ADMD, thus allowing it to identify which Authentication-Results header fields it can trust.
5. On the presumption that internal MTAs are fully compliant with Section 3.6 of [MAIL] and the compliant internal MTAs are using their own hostnames or the ADMD's DNS domain name as the authserv-id token, the header field proposed here should always appear above a Received header added by a trusted MTA. This can be used as a test for header field validity.

Support for some of these is being considered for future work.

In any case, a mechanism needs to exist for an MUA or filter to verify that the host that appears to have added the header field (a) actually did so and (b) is legitimately adding that header field for this delivery. Given the variety of messaging environments deployed today, consensus appears to be that specifying a particular mechanism for doing so is not appropriate for this document.

Mitigation of the forged header field attack can also be accomplished by moving the authentication results data into metadata associated with the message. In particular, an [SMTP] extension could be established to communicate authentication results from the border MTA to intermediate and delivery MTAs; the latter of these could arrange to store the authentication results as metadata retrieved and rendered along with the message by an [IMAP] client aware of a similar extension in that protocol. The delivery MTA would be told to trust data via this extension only from MTAs it trusts, and border MTAs would not accept data via this extension from any source. There is no vector in such an arrangement for forgery of authentication data by an outside agent.

7.2. Misleading Results

Until some form of service for querying the reputation of a sending agent is widely deployed, the existence of this header field indicating a "pass" does not render the message trustworthy. It is possible for an arriving piece of spam or other undesirable mail to pass checks by several of the methods enumerated above (e.g., a piece of spam signed using [DKIM] by the originator of the spam, which might be a spammer or a compromised system). In particular, this issue is not resolved by forged header field removal discussed above.

Hence, MUAs and downstream filters must take some care with use of this header even after possibly malicious headers are scrubbed.

7.3. Header Field Position

Despite the requirements of [MAIL], header fields can sometimes be reordered en route by intermediate MTAs. The goal of requiring header field addition only at the top of a message is an acknowledgment that some MTAs do reorder header fields, but most do not. Thus, in the general case, there will be some indication of which MTAs (if any) handled the message after the addition of the header field defined here.

7.4. Reverse IP Query Denial-of-Service Attacks

Section 4.6.4 of [SPF] describes a DNS-based denial-of-service attack for verifiers that attempt DNS-based identity verification of arriving client connections. A verifier wishing to do this check and report this information needs to take care not to go to unbounded lengths to resolve "A" and "PTR" queries. MUAs or other filters making use of an "iprev" result specified by this document need to be aware of the algorithm used by the verifier reporting the result and, especially, its limitations.

7.5. Mitigation of Backscatter

Failing to follow the instructions of Section 4.2 can result in a denial-of-service attack caused by the generation of [DSN] messages (or equivalent) to addresses that did not send the messages being rejected.

7.6. Internal MTA Lists

Section 5 describes a procedure for scrubbing header fields that may contain forged authentication results about a message. A compliant installation will have to include, at each MTA, a list of other MTAs known to be compliant and trustworthy. Failing to keep this list

current as internal infrastructure changes may expose an ADMD to attack.

7.7. Attacks against Authentication Methods

If an attack becomes known against an authentication method, clearly then the agent verifying that method can be fooled into thinking an inauthentic message is authentic, and thus the value of this header field can be misleading. It follows that any attack against the authentication methods supported by this document is also a security consideration here.

7.8. Intentionally Malformed Header Fields

It is possible for an attacker to add an Authentication-Results header field that is extraordinarily large or otherwise malformed in an attempt to discover or exploit weaknesses in header field parsing code. Implementers must thoroughly verify all such header fields received from MTAs and be robust against intentionally as well as unintentionally malformed header fields.

7.9. Compromised Internal Hosts

An internal MUA or MTA that has been compromised could generate mail with a forged From header field and a forged Authentication-Results header field that endorses it. Although it is clearly a larger concern to have compromised internal machines than it is to prove the value of this header field, this risk can be mitigated by arranging that internal MTAs will remove this header field if it claims to have been added by a trusted border MTA (as described above), yet the [SMTP] connection is not coming from an internal machine known to be running an authorized MTA. However, in such a configuration, legitimate MTAs will have to add this header field when legitimate internal-only messages are generated. This is also covered in Section 5.

7.10. Encapsulated Instances

MIME messages can contain attachments of type "message/rfc822", which contain other messages. Such an encapsulated message can also contain an Authentication-Results header field. Although the processing of these is outside of the intended scope of this document (see Section 1.3), some early guidance to MUA developers is appropriate here.

Since MTAs are unlikely to strip Authentication-Results header fields after mailbox delivery, MUAs are advised in Section 4.1 to ignore such instances within MIME attachments. Moreover, when extracting a

message digest to separate mail store messages or other media, such header fields should be removed so that they will never be interpreted improperly by MUAs that might later consume them.

7.11. Reverse Mapping

Although Section 3 of this memo includes explicit support for the "iprev" method, its value as an authentication mechanism is limited. Implementers of both this specification and agents that use the data it relays are encouraged to become familiar with the issues raised by [DNSOP-REVERSE] when deciding whether or not to include support for "iprev".

8. References

8.1. Normative References

- [ABNF] 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>>.
- [DKIM] Crocker, D., Ed., Hansen, T., Ed., and M. Kucherawy, Ed., "DomainKeys Identified Mail (DKIM) Signatures", STD 76, RFC 6376, DOI 10.17487/RFC6376, September 2011, <<http://www.rfc-editor.org/info/rfc6376>>.
- [IANA-HEADERS] 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>>.
- [KEYWORDS] 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>>.
- [MAIL] Resnick, P., Ed., "Internet Message Format", RFC 5322, DOI 10.17487/RFC5322, October 2008, <<http://www.rfc-editor.org/info/rfc5322>>.
- [MIME] 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>>.

- [RFC5451] Kucherawy, M., "Message Header Field for Indicating Message Authentication Status", RFC 5451, DOI 10.17487/RFC5451, April 2009, <<https://www.rfc-editor.org/info/rfc5451>>.
- [RFC6008] Kucherawy, M., "Authentication-Results Registration for Differentiating among Cryptographic Results", RFC 6008, DOI 10.17487/RFC6008, September 2010, <<https://www.rfc-editor.org/info/rfc6008>>.
- [RFC6530] Klensin, J. and Y. Ko, "Overview and Framework for Internationalized Email", RFC 6530, DOI 10.17487/RFC6530, February 2012, <<https://www.rfc-editor.org/info/rfc6530>>.
- [RFC6531] Yao, J. and W. Mao, "SMTP Extension for Internationalized Email", RFC 6531, DOI 10.17487/RFC6531, February 2012, <<https://www.rfc-editor.org/info/rfc6531>>.
- [RFC6532] Yang, A., Steele, S., and N. Freed, "Internationalized Email Headers", RFC 6532, DOI 10.17487/RFC6532, February 2012, <<https://www.rfc-editor.org/info/rfc6532>>.
- [RFC6577] Kucherawy, M., "Authentication-Results Registration Update for Sender Policy Framework (SPF) Results", RFC 6577, DOI 10.17487/RFC6577, March 2012, <<https://www.rfc-editor.org/info/rfc6577>>.
- [RFC7001] Kucherawy, M., "Message Header Field for Indicating Message Authentication Status", RFC 7001, DOI 10.17487/RFC7001, September 2013, <<https://www.rfc-editor.org/info/rfc7001>>.
- [RFC7601] Kucherawy, M., "Message Header Field for Indicating Message Authentication Status", RFC 7601, DOI 10.17487/RFC7601, August 2015, <<https://www.rfc-editor.org/info/rfc7601>>.
- [RFC8301] Kitterman, S., "Cryptographic Algorithm and Key Usage Update to DomainKeys Identified Mail (DKIM)", RFC 8301, DOI 10.17487/RFC8301, January 2018, <<https://www.rfc-editor.org/info/rfc8301>>.
- [SMTP] Klensin, J., "Simple Mail Transfer Protocol", RFC 5321, DOI 10.17487/RFC5321, October 2008, <<http://www.rfc-editor.org/info/rfc5321>>.

8.2. Informative References

- [ADSP] Allman, E., Fenton, J., Delany, M., and J. Levine, "DomainKeys Identified Mail (DKIM) Author Domain Signing Practices (ADSP)", RFC 5617, DOI 10.17487/RFC5617, August 2009, <<http://www.rfc-editor.org/info/rfc5617>>.
- [AR-VBR] Kucherawy, M., "Authentication-Results Registration for Vouch by Reference Results", RFC 6212, DOI 10.17487/RFC6212, April 2011, <<http://www.rfc-editor.org/info/rfc6212>>.
- [ATPS] Kucherawy, M., "DomainKeys Identified Mail (DKIM) Authorized Third-Party Signatures", RFC 6541, DOI 10.17487/RFC6541, February 2012, <<http://www.rfc-editor.org/info/rfc6541>>.
- [AUTH] Siemborski, R., Ed. and A. Melnikov, Ed., "SMTP Service Extension for Authentication", RFC 4954, DOI 10.17487/RFC4954, July 2007, <<http://www.rfc-editor.org/info/rfc4954>>.
- [AUTH-ESC] Kucherawy, M., "Email Authentication Status Codes", RFC 7372, DOI 10.17487/RFC7372, September 2014, <<http://www.rfc-editor.org/info/rfc7372>>.
- [DMARC] Kucherawy, M., Ed. and E. Zwicky, Ed., "Domain-based Message Authentication, Reporting, and Conformance (DMARC)", RFC 7489, DOI 10.17487/RFC7489, March 2015, <<http://www.rfc-editor.org/info/rfc7489>>.
- [DNS] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <<http://www.rfc-editor.org/info/rfc1035>>.
- [DNS-IP6] Thomson, S., Huitema, C., Ksinant, V., and M. Souissi, "DNS Extensions to Support IP Version 6", RFC 3596, DOI 10.17487/RFC3596, October 2003, <<http://www.rfc-editor.org/info/rfc3596>>.
- [DNSOP-REVERSE] Senie, D. and A. Sullivan, "Considerations for the use of DNS Reverse Mapping", Work in Progress, draft-ietf-dnsop-reverse-mapping-considerations-06, March 2008.
- [DOMAINKEYS] Delany, M., "Domain-Based Email Authentication Using

Public Keys Advertised in the DNS (DomainKeys)", RFC 4870, DOI 10.17487/RFC4870, May 2007, <<http://www.rfc-editor.org/info/rfc4870>>.

[DSN] Moore, K. and G. Vaudreuil, "An Extensible Message Format for Delivery Status Notifications", RFC 3464, DOI 10.17487/RFC3464, January 2003, <<http://www.rfc-editor.org/info/rfc3464>>.

[EMAIL-ARCH] Crocker, D., "Internet Mail Architecture", RFC 5598, DOI 10.17487/RFC5598, July 2009, <<http://www.rfc-editor.org/info/rfc5598>>.

[IMAP] Crispin, M., "INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1", RFC 3501, DOI 10.17487/RFC3501, March 2003, <<http://www.rfc-editor.org/info/rfc3501>>.

[POP3] Myers, J. and M. Rose, "Post Office Protocol - Version 3", STD 53, RFC 1939, DOI 10.17487/RFC1939, May 1996, <<http://www.rfc-editor.org/info/rfc1939>>.

[PRA] Lyon, J., "Purported Responsible Address in E-Mail Messages", RFC 4407, DOI 10.17487/RFC4407, April 2006, <<http://www.rfc-editor.org/info/rfc4407>>.

[RFC7410] Kucherawy, M., "A Property Types Registry for the Authentication-Results Header Field", RFC 7410, DOI 10.17487/RFC7410, December 2014, <<http://www.rfc-editor.org/info/rfc7410>>.

[RRVS] Mills, W. and M. Kucherawy, "The Require-Recipient-Valid-Since Header Field and SMTP Service Extension", RFC 7293, DOI 10.17487/RFC7293, July 2014, <<http://www.rfc-editor.org/info/rfc7293>>.

[SECURITY] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, DOI 10.17487/RFC3552, July 2003, <<http://www.rfc-editor.org/info/rfc3552>>.

[SENDERID] Lyon, J. and M. Wong, "Sender ID: Authenticating E-Mail", RFC 4406, DOI 10.17487/RFC4406, April 2006, <<http://www.rfc-editor.org/info/rfc4406>>.

[SMIME-REG]

Melnikov, A., "Authentication-Results Registration for S/MIME Signature Verification", RFC 7281, DOI 10.17487/RFC7281, June 2014, <<http://www.rfc-editor.org/info/rfc7281>>.

[SPF] Kitterman, S., "Sender Policy Framework (SPF) for Authorizing Use of Domains in Email, Version 1", RFC 7208, DOI 10.17487/RFC7208, April 2014, <<http://www.rfc-editor.org/info/rfc7208>>.

[VBR] Hoffman, P., Levine, J., and A. Hathcock, "Vouch By Reference", RFC 5518, DOI 10.17487/RFC5518, April 2009, <<http://www.rfc-editor.org/info/rfc5518>>.

Appendix A. Legacy MUAs

Implementers of this protocol should be aware that many MUAs are unlikely to be retrofitted to support the new header field and its semantics. In the interests of convenience and quicker adoption, a delivery MTA might want to consider adding things that are processed by existing MUAs in addition to the Authentication-Results header field. One suggestion is to include a Priority header field, on messages that don't already have such a header field, containing a value that reflects the strength of the authentication that was accomplished, e.g., "low" for weak or no authentication, "normal" or "high" for good or strong authentication.

Some modern MUAs can already filter based on the content of this header field. However, there is keen interest in having MUAs make some kind of graphical representation of this header field's meaning to end users. Until this capability is added (i.e., while this specification and its successors continue to be adopted), other interim means of conveying authentication results may be necessary.

Appendix B. Authentication-Results Examples

This section presents some examples of the use of this header field to indicate authentication results.

B.1. Trivial Case; Header Field Not Present

The trivial case:

```
Received: from mail-router.example.com
         (mail-router.example.com [192.0.2.1])
         by server.example.org (8.11.6/8.11.6)
         with ESMTTP id g1G0r1kA003489;
         Fri, Feb 15 2002 17:19:07 -0800
From: sender@example.com
Date: Fri, Feb 15 2002 16:54:30 -0800
To: receiver@example.org
Message-Id: <12345.abc@example.com>
Subject: here's a sample
```

Hello! Goodbye!

Example 1: Trivial Case

The Authentication-Results header field is completely absent. The MUA may make no conclusion about the validity of the message. This could be the case because the message authentication services were not available at the time of delivery, or no service is provided, or the MTA is not in compliance with this specification.

B.2. Nearly Trivial Case; Service Provided, but No Authentication Done

A message that was delivered by an MTA that conforms to this specification but provides no actual message authentication service:

```
Authentication-Results: example.org 1; none
Received: from mail-router.example.com
         (mail-router.example.com [192.0.2.1])
         by server.example.org (8.11.6/8.11.6)
         with ESMTTP id g1G0r1kA003489;
         Fri, Feb 15 2002 17:19:07 -0800
From: sender@example.com
Date: Fri, Feb 15 2002 16:54:30 -0800
To: receiver@example.org
Message-Id: <12345.abc@example.com>
Subject: here's a sample
```

Hello! Goodbye!

Example 2: Header Present but No Authentication Done

The Authentication-Results header field is present, showing that the delivering MTA conforms to this specification. It used its DNS

domain name as the authserv-id. The presence of "none" (and the absence of any method or result tokens) indicates that no message authentication was done. The version number of the specification to which the field's content conforms is explicitly provided.

B.3. Service Provided, Authentication Done

A message that was delivered by an MTA that conforms to this specification and applied some message authentication:

```
Authentication-Results: example.com;
                        spf=pass smtp.mailfrom=example.net
Received: from dialup-1-2-3-4.example.net
        (dialup-1-2-3-4.example.net [192.0.2.200])
        by mail-router.example.com (8.11.6/8.11.6)
        with ESMTTP id g1G0r1kA003489;
        Fri, Feb 15 2002 17:19:07 -0800
From: sender@example.net
Date: Fri, Feb 15 2002 16:54:30 -0800
To: receiver@example.com
Message-Id: <12345.abc@example.net>
Subject: here's a sample
```

Hello! Goodbye!

Example 3: Header Reporting Results

The Authentication-Results header field is present, indicating that the border MTA conforms to this specification. The authserv-id is once again the DNS domain name. Furthermore, the message was authenticated by that MTA via the method specified in [SPF]. Note that since that method cannot authenticate the local-part, it has been omitted from the result's value. The MUA could extract and relay this extra information if desired.

B.4. Service Provided, Several Authentications Done, Single MTA

A message that was relayed inbound via a single MTA that conforms to this specification and applied three different message authentication checks:

```
Authentication-Results: example.com;
    auth=pass (cram-md5) smtp.auth=sender@example.net;
    spf=pass smtp.mailfrom=example.net
Authentication-Results: example.com;
    sender-id=pass header.from=example.net
Received: from dialup-1-2-3-4.example.net (8.11.6/8.11.6)
    (dialup-1-2-3-4.example.net [192.0.2.200])
    by mail-router.example.com (8.11.6/8.11.6)
    with ESMTPA id g1G0r1kA003489;
    Fri, Feb 15 2002 17:19:07 -0800
Date: Fri, Feb 15 2002 16:54:30 -0800
To: receiver@example.com
From: sender@example.net
Message-Id: <12345.abc@example.net>
Subject: here's a sample
```

Hello! Goodbye!

Example 4: Headers Reporting Results from One MTA

The Authentication-Results header field is present, indicating that the delivering MTA conforms to this specification. Once again, the receiving DNS domain name is used as the authserv-id. Furthermore, the sender authenticated herself/himself to the MTA via a method specified in [AUTH], and both SPF and Sender ID checks were done and passed. The MUA could extract and relay this extra information if desired.

Two Authentication-Results header fields are not required since the same host did all of the checking. The authenticating agent could have consolidated all the results into one header field.

This example illustrates a scenario in which a remote user on a dial-up connection (example.net) sends mail to a border MTA (example.com) using SMTP authentication to prove identity. The dial-up provider has been explicitly authorized to relay mail as example.net, producing "pass" results from the SPF and Sender ID checks.

B.5. Service Provided, Several Authentications Done, Different MTAs

A message that was relayed inbound by two different MTAs that conform to this specification and applied multiple message authentication checks:

```
Authentication-Results: example.com;
    sender-id=fail header.from=example.com;
    dkim=pass (good signature) header.d=example.com
Received: from mail-router.example.com
    (mail-router.example.com [192.0.2.1])
    by auth-checker.example.com (8.11.6/8.11.6)
    with ESMTTP id i7PK0sH7021929;
    Fri, Feb 15 2002 17:19:22 -0800
DKIM-Signature: v=1; a=rsa-sha256; s=gatsby; d=example.com;
    t=1188964191; c=simple/simple; h=From:Date:To:Subject:
    Message-Id:Authentication-Results;
    bh=sEuZGD/pSr7ANysbY3jtdaQ3Xv9xPQtS0m70;
    b=EToRSuvUfQVP3Bkz ... rTB0t0gYnBVCM=
Authentication-Results: example.com;
    auth=pass (cram-md5) smtp.auth=sender@example.com;
    spf=fail smtp.mailfrom=example.com
Received: from dialup-1-2-3-4.example.net
    (dialup-1-2-3-4.example.net [192.0.2.200])
    by mail-router.example.com (8.11.6/8.11.6)
    with ESMTTPA id g1G0r1kA003489;
    Fri, Feb 15 2002 17:19:07 -0800
From: sender@example.com
Date: Fri, Feb 15 2002 16:54:30 -0800
To: receiver@example.com
Message-Id: <12345.abc@example.com>
Subject: here's a sample
```

Hello! Goodbye!

Example 5: Headers Reporting Results from Multiple MTAs

The Authentication-Results header field is present, indicating conformance to this specification. Once again, the authserv-id used is the recipient's DNS domain name. The header field is present twice because two different MTAs in the chain of delivery did authentication tests. The first MTA, mail-router.example.com, reports that SMTP AUTH and SPF were both used and that the former passed while the latter failed. In the SMTP AUTH case, additional information is provided in the comment field, which the MUA can choose to render if desired.

The second MTA, auth-checker.example.com, reports that it did a

Sender ID test (which failed) and a DKIM test (which passed). Again, additional data about one of the tests is provided as a comment, which the MUA may choose to render. Also noteworthy here is the fact that there is a DKIM signature added by example.com that assured the integrity of the lower Authentication-Results field.

Since different hosts did the two sets of authentication checks, the header fields cannot be consolidated in this example.

This example illustrates more typical transmission of mail into example.com from a user on a dial-up connection example.net. The user appears to be legitimate as he/she had a valid password allowing authentication at the border MTA using SMTP AUTH. The SPF test failed since example.com has not granted example.net's dial-up network authority to relay mail on its behalf. The Sender ID test failed since example.com has not granted mail-router.example.com authority to relay mail on its behalf. However, the DKIM test passed because the sending user had a private key matching one of example.com's published public keys and mail-router.example.com used it to sign the message.

B.6. Service Provided, Multi-tiered Authentication Done

A message that had authentication done at various stages, one of which was outside the receiving ADMD:

```
Authentication-Results: example.com;
    dkim=pass reason="good signature"
    header.i=@mail-router.example.net;
    dkim=fail reason="bad signature"
    header.i=@newyork.example.com
Received: from mail-router.example.net
    (mail-router.example.net [192.0.2.250])
    by chicago.example.com (8.11.6/8.11.6)
    for <recipient@chicago.example.com>
    with ESMTTP id i7PK0sH7021929;
    Fri, Feb 15 2002 17:19:22 -0800
DKIM-Signature: v=1; a=rsa-sha256; s=furble;
    d=mail-router.example.net; t=1188964198; c=relaxed/simple;
    h=From:Date:To:Message-Id:Subject:Authentication-Results;
    bh=ftA9J6GtX8OpwUECzHnCkRzKwluk6FNiLfJl5NmV49E=;
    b=oINEO8hgn/gnunsg ... 9n9ODSNFSDij3=
Authentication-Results: example.net;
    dkim=pass (good signature) header.i=@newyork.example.com
Received: from smtp.newyork.example.com
    (smtp.newyork.example.com [192.0.2.220])
    by mail-router.example.net (8.11.6/8.11.6)
    with ESMTTP id g1G0r1kA003489;
    Fri, Feb 15 2002 17:19:07 -0800
DKIM-Signature: v=1; a=rsa-sha256; s=gatsby;
    d=newyork.example.com;
    t=1188964191; c=simple/simple;
    h=From:Date:To:Message-Id:Subject;
    bh=sEu28nfs9fuZGD/pSr7ANysbY3jtdaQ3Xv9xPQtS0m7=;
    b=EToRSuvUfQVP3Bkz ... rTB0t0gYnBVCM=
From: sender@newyork.example.com
Date: Fri, Feb 15 2002 16:54:30 -0800
To: meetings@example.net
Message-Id: <12345.abc@newyork.example.com>
Subject: here's a sample
```

Example 6: Headers Reporting Results from Multiple MTAs in Different ADMDs

In this example, we see multi-tiered authentication with an extended trust boundary.

The message was sent from someone at example.com's New York office (newyork.example.com) to a mailing list managed at an intermediary.

The message was signed at the origin using DKIM.

The message was sent to a mailing list service provider called example.net, which is used by example.com. There, meetings@example.net is expanded to a long list of recipients, one of whom is at the Chicago office. In this example, we will assume that the trust boundary for chicago.example.com includes the mailing list server at example.net.

The mailing list server there first authenticated the message and affixed an Authentication-Results header field indicating such using its DNS domain name for the authserv-id. It then altered the message by affixing some footer text to the body, including some administrivia such as unsubscription instructions. Finally, the mailing list server affixes a second DKIM signature and begins distribution of the message.

The border MTA for chicago.example.com explicitly trusts results from mail-router.example.net, so that header field is not removed. It performs evaluation of both signatures and determines that the first (most recent) is a "pass" but, because of the aforementioned modifications, the second is a "fail". However, the first signature included the Authentication-Results header added at mail-router.example.net that validated the second signature. Thus, indirectly, it can be determined that the authentications claimed by both signatures are indeed valid.

Note that two styles of presenting metadata about the result are in use here. In one case, the "reason=" clause is present, which is intended for easy extraction by parsers; in the other case, the CFWS production of the ABNF is used to include such data as a header field comment. The latter can be harder for parsers to extract given the varied supported syntaxes of mail header fields.

B.7. Comment-Heavy Example

The formal syntax permits comments within the content in a number of places. For the sake of illustration, this example is also legal:

```
Authentication-Results: foo.example.net (foobar) 1 (baz);
    dkim (Because I like it) / 1 (One yay) = (wait for it) fail
    policy (A dot can go here) . (like that) expired
    (this surprised me) = (as I wasn't expecting it) 1362471462
```

Example 7: A Very Comment-Heavy but Perfectly Legal Example

Appendix C. Operational Considerations about Message Authentication

This protocol is predicated on the idea that authentication (and presumably in the future, reputation) work is typically done by border MTAs rather than MUAs or intermediate MTAs; the latter merely make use of the results determined by the former. Certainly this is not mandatory for participation in electronic mail or message authentication, but this protocol and its deployment to date are based on that model. The assumption satisfies several common ADMD requirements:

1. Service operators prefer to resolve the handling of problem messages as close to the border of the ADMD as possible. This enables, for example, rejection of messages at the SMTP level rather than generating a DSN internally. Thus, doing any of the authentication or reputation work exclusively at the MUA or intermediate MTA renders this desire unattainable.
2. Border MTAs are more likely to have direct access to external sources of authentication or reputation information since modern MUAs are more likely to be heavily firewalled. Thus, some MUAs might not even be able to complete the task of performing authentication or reputation evaluations without complex proxy configurations or similar burdens.
3. MUAs rely upon the upstream MTAs within their trust boundaries to make correct (as much as is possible) evaluations about the message's envelope, header, and content. Thus, MUAs don't need to know how to do the work that upstream MTAs do; they only need the results of that work.
4. Evaluations about the quality of a message, from simple token matching (e.g., a list of preferred DNS domains) to cryptanalysis (e.g., public/private key work), do have a cost and thus need to be minimized. To that end, performing those tests at the border MTA is far preferred to doing that work at each MUA that handles a message. If an ADMD's environment adheres to common messaging protocols, a reputation query or an authentication check performed by a border MTA would return the same result as the same query performed by an MUA. By contrast, in an environment where the MUA does the work, a message arriving for multiple recipients would thus cause authentication or reputation evaluation to be done more than once for the same message (i.e., at each MUA), causing needless amplification of resource use and creating a possible denial-of-service attack vector.
5. Minimizing change is good. As new authentication and reputation methods emerge, the list of methods supported by this header

field would presumably be extended. If MUAs simply consume the contents of this header field rather than actually attempt to do authentication and/or reputation work, then MUAs only need to learn to parse this header field once; emergence of new methods requires only a configuration change at the MUAs and software changes at the MTAs (which are presumably fewer in number). When choosing to implement these functions in MTAs vs. MUAs, the issues of individual flexibility, infrastructure inertia, and scale of effort must be considered. It is typically easier to change a single MUA than an MTA because the modification affects fewer users and can be pursued with less care. However, changing many MUAs is more effort than changing a smaller number of MTAs.

6. For decisions affecting message delivery and display, assessment based on authentication and reputation is best performed close to the time of message transit, as a message makes its journey toward a user's inbox, not afterwards. DKIM keys and IP address reputations, etc., can change over time or even become invalid, and users can take a long time to read a message once delivered. The value of this work thus degrades, perhaps quickly, once the delivery process has completed. This seriously diminishes the value of this work when done elsewhere than at MTAs.

Many operational choices are possible within an ADMD, including the venue for performing authentication and/or reputation assessment. The current specification does not dictate any of those choices. Rather, it facilitates those cases in which information produced by one stage of analysis needs to be transported with the message to the next stage.

Appendix D. Changes Since RFC7601

- o Added IANA registration for DKIM "a" and "s" properties.
- o Include EAI guidance.
- o Adjust some ABNF tokens and names for easier inclusion by other documents.

Appendix E. Acknowledgments

The author wishes to acknowledge the following individuals for their review and constructive criticism of this document: Seth Blank, Tim Draegen, John Levine, Scott Kitterman, and Alessandro Vesely.

Author's Address

Murray S. Kucherawy
270 Upland Drive
San Francisco, CA 94127
United States

Email: superuser@gmail.com

