Authenticated Received Chain (ARC) Protocol

draft-ietf-dmarc-arc-protocol-01

Abstract

Authenticated Received Chain (ARC) permits an organization which is creating or handling email to indicate its involvement with the handling process. It defines a set of cryptographically signed header fields in a manner analogous to that of DKIM. Assertion of responsibility is validated through a cryptographic signature and by querying the Signer’s domain directly to retrieve the appropriate public key. Changes in the message that might break DKIM can be identified through the ARC set of header fields.

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 June 4, 2017.

Copyright Notice

Copyright (c) 2016 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 .............................................. 3
2. Requirements ............................................. 4
   2.1. Primary Design Criteria .............................. 4
   2.2. Out of Scope ......................................... 4
   2.3. Utility ............................................... 4
3. Terminology ................................................ 5
4. Overview .................................................... 5
5. Definition .................................................. 5
   5.1. Description of the New Header Fields ............... 6
      5.1.1. ARC-Seal ........................................ 6
      5.1.2. ARC-Message-Signature .......................... 9
      5.1.3. ARC-Authentication-Results ........................ 10
   5.2. Constructing the ARC-Seal Set ........................ 11
      5.2.1. Handling Violations in the ARC Sets ............ 12
   5.3. Key Management and Binding .......................... 12
      5.3.1. Namespace ....................................... 12
6. Usage ....................................................... 13
   6.1. Participation ......................................... 13
   6.2. Relationship between DKIM Signatures and ARC Headers .. 13
   6.3. Validating the ARC Set of Header Fields ............. 13
   6.4. ARC SetValidity ...................................... 13
      6.4.1. Assessing Chain Validity Violations ............. 13
      6.4.2. Responding to ARC Validity Violations ........... 14
      6.4.3. Recording the Results of ARC Evaluation .......... 14
      6.4.4. Output Data Points from ARC Evaluation .......... 14
      6.4.5. Reporting ARC Effects for DMARC Local Policy ... 14
7. Privacy Considerations ..................................... 15
8. IANA Considerations ....................................... 15
   8.1. Authentication-Results Method Registry Update ........ 15
   8.2. Definitions of the ARC header fields ............... 15
9. Implementation Status ..................................... 16
   9.1. GMail test reflector ................................ 16
   9.2. AOL test reflector and internal tagging ............. 17
   9.3. dkimpy patch ........................................ 18
   9.4. OpenARC .............................................. 18
   9.5. Mailman addition .................................... 19
1. Introduction

The development of strong domain authentication through Sender Policy Framework (SPF) [RFC7208] and DomainKeys Identified Mail (DKIM) [RFC6376] has led to the implementation of the DMARC framework [RFC7489] which extends the authentication to the author's "From:" (RFC5322.From) field and permits publishing policies for non-compliant messages. Implicit within the DMARC framework is a requirement that any intermediaries between the source system and ultimate receiver system need to preserve the validity of the DKIM signature; however, there are common legitimate email practices which break the DKIM validation ([DMARC-INTEROP]). This specification defines an Authenticated Received Chain (ARC). ARC addresses the problems with the untrustworthiness of the standard Received header field sequence. Through the information tracked in the ARC series of headers, receivers can develop a more nuanced interpretation to guide any local policies related to messages that arrive with broken domain authentication (DMARC).

Forgery of the Received header fields is a common tactic used by bad actors. One of the goals of this specification defines a comparable set of trace header fields which can be relied upon by receivers,
assuming all ADeinistrative Management Domain (ADMD) intermediary
handlers of a message participate in ARC.

The Authentication-Results (A-R) mechanism [RFC7601] permits the
output of an email authentication evaluation process to be
transmitted from the evaluating agent to a consuming agent that uses
the information. On its own, A-R is believable only within a trust
domain. ARC provides a protection mechanism for the data, permitting
the communication to cross trust domain boundaries.

2. Requirements

The specification of the ARC framework is driven by the following
high-level goals, security considerations, and practical operational
requirements.

2.1. Primary Design Criteria

- Provide a verifiable "chain of custody" for email messages;
- Not require changes for originators of email;
- Support the verification of the ARC header field set by each hop
  in the handling chain;
- Work at Internet scale; and
- Provide a trustable mechanism for the communication of
  Authentication-Results across trust boundaries.

2.2. Out of Scope

ARC is not a trust framework. Users of the ARC header fields are
cautioned against making unsubstantiated conclusions when
encountering a "broken" ARC sequence.

2.3. Utility

The ARC-related set of header fields can be used (when validated) to
determine the path that an email message has taken between the
originating system and receiver. Subject to the cautions mentioned
in Section 10, this information can assist in determining any local
policy overrides to for violations of origination domain
authentication policies.
3. Terminology

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

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

Readers are encouraged to be familiar with the contents of [RFC5598], and in particular, the potential roles of intermediaries in the delivery of email.

Syntax descriptions use Augmented BNF (ABNF) [RFC5234].

4. Overview

When an email message is received without a properly validated originating domain, the inability to believe the accuracy of a series of Received header fields prevents receiving systems from having a way to infer anything about the handling of the message by looking at the ADMDs through which the message has traveled.

With ARC, participating ADMDs are able to securely register their handling of an email message. If all mediators ([RFC5598]) participate in the ARC process, receivers will be able to rely upon the chain and make local policy decisions informed by that information.

The ARC set of header fields provides a method by which participating intermediaries can indicate the hand-offs for email messages.

5. Definition

This specification defines three new header fields:

- Header field name: ARC-Seal (abbreviated below as AS)
- Header field name: ARC-Message-Signature (abbreviated below as AMS)
- Header field name: ARC-Authentication-Results (abbreviated below as AAR)

Collectively, these header fields form a connected set of attribution information by which receivers can identify the handling path for a message. As described below, a distinct set of these fields share a common sequence number, identified in an "i=" tag. Such a correlated group of header fields is referred to as an "ARC set".
Specific references to individual header fields use the header field names to distinguish such references.

The ARC sets SHOULD be added at the top of a message header as it transits MTAs that do authentication checks, so some idea of how far away the checks were done can be inferred. They are therefore considered to be a trace field as defined in [RFC5321], and all of the related definitions in that document apply.

Relative ordering of different trace header fields (the ARC sets, DKIM, Received, etc.) is unimportant for this specification. In general, trace header fields, such as ARC, SHOULD be added at the top of the email header fields, but receivers MUST be able to process the header fields from wherever they are found in the message header. Ordering amongst the individual ARC header fields and sets is specified below and MUST be followed for proper canonicalized signing and evaluation.

5.1. Description of the New Header Fields

5.1.1. ARC-Seal

ARC-Seal is a Structured Header Field as defined in Internet Message Format ([RFC5322]). All of the related definitions in that document apply.

The ARC-Seal makes use of Tag=Value Lists as defined in [RFC6376], Section 3.2.

The value of the header field consists of an authentication sequence identifier, and a series of statements and supporting data. The statements indicate relevant data about the signing of the ARC set. The header field can appear more than once in a single message, but each instance MUST have a unique "i=" value.

The ARC-Seal header field includes a digital signature of all preceding ARC message header fields on the message.

5.1.1.1. Tags in the ARC-Seal Header Field Value

The following tags are the only supported tags for an ARC-Seal field. All of them MUST be present. Unknown tags MUST be ignored and do not affect the validity of the header.

- **a** = hash algorithm; syntax is the same as the "a=" tag defined in Section 3.5 of [RFC6376];
o  b = digital signature; syntax is the same as the "b=" tag defined in Section 3.5 of [RFC6376];

o  cv = chain validation status: valid values:
    *  'none' = no pre-existing chain;
    *  'fail' = the chain as received does not validate; or
    *  'pass' = valid chain received.

o  d = domain for key; syntax is the same as the "d=" tag defined in Section 3.5 of [RFC6376];

o  i = "instance" or sequence number; monotonically increasing at each "sealing" entity, beginning with '1', may not exceed '1024'

o  s = selector for key; syntax is the same as the "s=" tag defined in Section 3.5 of [RFC6376];

o  t = timestamp; syntax is the same as the "t=" tag defined in Section 3.5 of [RFC6376].

5.1.1.2. Differences between DKIM-Signature and ARC-Seal

No 'bh' value is defined for ARC-Seal, since only message header fields are ever signed by the ARC-Seal.

ARC-Seal does not use the 'h' tag (the list of signed header fields) that is defined for DKIM-Signatures because the list of applicable header fields is fully determined by the construction rules (see Section 5.1.1.3).

ARC-Seal does not use the 'c' (canonicalization) tag because only 'relaxed' canonicalization [RFC6376] is allowed for ARC-Seal header field canonicalization.

5.1.1.3. Deterministic 'h' Value for ARC-Seal

In this section, the term "scope" is used to indicate those header fields signed by an ARC-Seal header field. A number in parentheses indicates the instance of that field, starting at 1. The suffix "-no-b" is used with an ARC-Seal field to indicate that its "b" field is empty at the time the signature is computed, as described in Section 3.5 of [RFC6376]. "AAR" refers to ARC-Authentication-Results, "AMS" to ARC-Message-Signature, "AS" to ARC-Seal, and "ASB" to an ARC-Seal with an empty "b" tag.
Generally, the scope of an ARC set for a message containing "n" ARC sets is the concatenation of the following, for x (instance number) from 1 to n:

- AAR(x);
- AMS(x);
- ASB(x) if x = n, else AS(x)

Thus for a message with no seals (i.e., upon injection), the scope of the first ARC set is AAR(1):AMS(1):ASB(1). The ARC set thus generated would produce a first ARC-Seal with a "b" value. The next ARC set would include in its signed content the prior scope, so it would have a scope of AAR(1):AMS(1):AS(1):AAR(2):AMS(2):ASB(2).

Note: Typically header field sets appear within the header in descending instance order.

5.1.1.4. Computing the ‘b’ Tag Value for ARC-Seal

The ARC-Seal generation process mirrors the procedure used for DKIM-Signature fields described in Section 5 of [RFC6376] in that it is at first generated with empty "b" field for the purpose of signature generation, and then the "b" value is added just prior to adding the ARC-Seal field to the message.

In particular, signing calculation MUST be done in bottom-up order as specified in Section 5.4.2 of [RFC6376] and as illustrated above Section 5.1.1.3.

5.1.1.5. Determining the ‘cv’ Tag Value for ARC-Seal

In order for a series of ARC sets to be considered valid, the following statements MUST be satisfied:

1. All ARC-Seal header fields MUST validate;
2. All ARC-Seal header fields MUST have a chain value (cv=) status of "pass" (except the first which MUST be "none"); and
3. The newest (highest instance number (i=)) AMS header field MUST validate.
5.1.1.5.1. Pseudocode to Determine Chain Value Status:

In the algorithm below, a "hop" is represented by the ARC set bearing a particular instance number. The number of hops is the same as the highest instance number found in the ARC sets, or 0 (zero) if there are no ARC sets found within the header.

"Success" means that the signature found in the referenced header validates when against the content which was signed.

```plaintext
if (num_hops == 0) {
    return "none"
} else {
    if (validate(latest_hop.AMS) != success) {
        return "fail"
    } else {
        // note that instance is always >= 1 by definition
        for each hop (from highest instance to lowest) {
            if ((hop_num > 1 and hop.ARC-Seal.cv == "pass") or
                (hop_num == 1 and hop.ARC-Seal.cv == "none")) {
                if (validate(hop.ARC-Seal) != success) {
                    return "fail"
                }
            } else {
                return "fail"
            }
        }
        return "pass"
    }
}
```

5.1.2. ARC-Message-Signature

The ARC-Message-Signature header field is a special variant of a DKIM-Signature [RFC6376], using only the relaxed header canonicalization rules specified in [RFC6376].

The ARC-Message-Signature header field can appear multiple times in a single message but each instance MUST have a unique "i=" value.

5.1.2.1. Differences between DKIM-Signature and ARC-Message-Signature
5.1.2.1.1. Header Fields Eligible For ARC-Message-Signature Inclusion

Participants may include any other header fields within the scope of the ARC-Message-Signature signature except that they MUST NOT include ARC-Seal headers fields. In particular, including all DKIM-Signature header fields and all ARC-Authentication-Results header fields is RECOMMENDED. The advice regarding headers to include or avoid for ARC-Message-Signature is otherwise identical to that specified in section 5.4 of [RFC6376].

5.1.2.1.2. "Canonicalization" 'c' Tag Value

The ARC-Message-Signature header field MUST be created using the header and body canonicalization rules mechanisms in Section 3.4 of [RFC6376]. The corresponding "c=" tag value MUST be specified in the AMS header field value.

5.1.2.1.3. "Instance" 'i' Tag Value

Contrary to DKIM, the 'i' tag for ARC-Message-Signature identifies the sequential instance of the field, thus indicating that it is part of a particular ARC set. That is, an ARC-Message-Signature, ARC-Seal, and ARC-Authentication-Results all bearing an "i=" tag with the same value are part of the same ARC set (see Section 5.1.1.1).

5.1.2.1.4. 'v' Tag Value

There is no "v" tag for ARC-Message-Signature.

5.1.2.2. Computing the 'b' Tag Value for ARC-Message-Signature

As with DKIM-Signature and ARC-Seal header fields, the "b" tag of the ARC-Message-Signature is empty until the signature is actually computed, and only then is it added to the header field, before affixing the ARC-Message-Signature to the message.

As with ARC-Seal and DKIM-Signature header fields, the order of header fields signed MUST be done in bottom-up order.

5.1.3. ARC-Authentication-Results

ARC-Authentication-Results is a direct copy of the Authentication-Results header field [RFC7601] created for archival purposes by the each MTA outside of the trust boundary of the originating system which is contributing to the chain of ARC header fields. The corresponding instance ('i=") tag value MUST be prefixed to the Authentication-Results.
The instance identifier MUST be separated from the rest of the Authentication-Results value contents with a semi-colon (';', 0x3b).

The value of the header field (after removing comments) consists of an instance identifier, an authentication identifier, and then a series of statements and supporting data, as described in [RFC7601]. The header field can appear multiple times in a single message but each instance MUST have a unique "i=" value.

5.1.3.1. ‘i’ Tag Value

ARC-Authentication-Results requires inclusion of an "i=" tag before the "authserv-id" which indicates the ARC set to which it belongs as described in the previous section (see Section 5.1.1.1).

The "i=" tag MUST be separated from the rest of the Authentication-Results value contents with a semi-colon (';', 0x3b).

5.2. Constructing the ARC-Seal Set

The ARC-Seal is built in the same fashion as the analogous DKIM-Signature [RFC6376], using the relaxed header canonicalization rules specified in that document but with a strict ordering component for the header fields covered by the cryptographic signature:

1. The ARC sets MUST be ordered in descending instance (i=) order.
2. The referenced ARC-Message-Signatures (matching i= value) MUST immediately follow the ARC-Seal instance which included the reference.
3. The associated ARC-Authentication-Results header field (matching i= value) MUST be the last item in the list for each set of ARC header fields.

Thus, when prefixing ARC header fields to the existing header,

1. the AAR header would be prefixed first; then
2. the AMS would be calculated and prefixed;
3. lastly the AS would be calculated and prefixed.

The ARC-Message-Signature field(s) MUST not include any of the ARC-Seal header field(s) (from prior ARC sets) in their signing scope in order maintain a separation of responsibilities. When adding an ARC-Authentication-Results header field, it MUST be added before computing the ARC-Message-Signature. When "sealing" the message, an
4.5.3.  Key Management and Binding

The public keys for ARC header fields follow the same requirements and semantics as those for DKIM-Signatures, described in Section 3.6 of [RFC6376]. Operators may use distinct selectors for the ARC header fields at their own discretion.

5.3.1.  Namespace

All ARC-related keys are stored in the same namespace as DKIM keys [RFC6376]: "_.domainkey" specifically by adding the "._domainkey" suffix to the name of the key (the "selector"). For example, given
an ARC-Seal (or ARC-Message-Signature) field of a "d=" tag value of "example.com" and an "s=" value of "foo.bar", the DNS query seeking the public key will a query at the name "foo.bar._domainkey.example.com".

6. Usage

For a more thorough treatment of the recommended usage of the ARC header fields for both intermediaries and end receivers, please consult [ARC-USAGE].

6.1. Participation

The inclusion of additional ARC sets is to be done whenever a trust boundary is crossed, and especially when prior DKIM-Signatures might not survive the handling being performed such as some mailing lists that modify the content of messages or some gateway transformations. Note that trust boundaries might or might not exactly correspond with ADMD boundaries.

Each participating ADMD MUST validate the preceding ARC set as a part of asserting their own seal. Even if the set is determined to be invalid, a participating ADMD SHOULD apply their own seal because this can help in analysis of breakage points in the chain.

6.2. Relationship between DKIM Signatures and ARC Headers

ARC-aware DKIM signers do not DKIM-sign any ARC header fields.

6.3. Validating the ARC Set of Header Fields

Determining the validity of a chain of ARC sets is defined above in Section 5.1.1.5. Validation failures MUST be indicated with a "cv=" tag value of 'fail' when attaching a subsequent ARC-Seal header field.

6.4. ARC Set Validity

6.4.1. Assessing Chain Validity Violations

There are a wide variety of ways in which the ARC set of header fields can be broken. 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.

This specification is exclusively focused on well-behaved, participating intermediaries that result in a valid chain of ARC-related header fields. The value of such a well-formed, valid chain
needs to be interpreted with care since malicious content can be easily introduced by otherwise well-intended senders through machine or account compromises. All normal content-based analysis still needs to be performed on any messages bearing a valid chain of ARC header sets.

6.4.2. Responding to ARC Validity Violations

If a receiver determines that the ARC set of header fields has is invalid, the receiver MAY signal the breakage through the extended SMTP response code 5.7.7 [RFC3463] "message integrity failure" [ENHANCED-STATUS] and corresponding SMTP response code.

6.4.3. Recording the Results of ARC Evaluation

Receivers MAY add an "arc=pass" or "arc=fail" method annotation into a locally-affixed Authentication-Results [RFC7601] header field.

6.4.4. Output Data Points from ARC Evaluation

The evaluation of a series of ARC sets results in the following data which MAY be used to inform local-policy decisions:

- A list of the "d=" domains found in the validated (all) ARC-Seal header fields;
- The "d=" domain found in the most recent (highest instance number) AMS header field (since that is the only one necessarily validated)

6.4.5. Reporting ARC Effects for DMARC Local Policy

Receivers SHOULD indicate situations in which ARC evaluation influenced the results of their local policy determination. DMARC reporting of ARC-informed decisions is augmented by adding a local_policy comment explanation as follows:

<policy_evaluated>
  <disposition>delivered</disposition>
  <dkim>fail</dkim>
  <spf>fail</spf>
  <reason>
    <type>local_policy</type>
    <comment>arc=pass ams=d1.example d=d1.example,d2.example</comment>
  </reason>
</policy_evaluated>
7. Privacy Considerations

The ARC-Seal chain provides a verifiable record of the handlers for a message. Anonymous remailers will probably not find this to match their operating goals.

8. IANA Considerations

This specification adds three new header fields as defined below.

8.1. Authentication-Results Method Registry Update

This draft adds one item to the IANA "Email Authentication Methods" registry:

- **Method**: arc
  - **Defined**: [I-D.ARC]
  - **ptype**: header
  - **Property**: chain evaluation result
  - **Value**: chain evaluation result status (see Section 5.1.1.1)
  - **Status**: active
  - **Version**: 1

8.2. Definitions of the ARC header fields

This specification adds three new header fields to the "Permanent Message Header Field Registry", as follows:

- **Header field name**: ARC-Seal
  - **Applicable protocol**: mail
  - **Status**: draft
  - **Author/Change controller**: OAR-Dev Group
  - **Specification document(s)**: [I-D.ARC]
  - **Related information**: [RFC6376]

- **Header field name**: ARC-Message-Signature
Applicable protocol: mail

Status: draft

Author/Change controller: OAR-Dev Group

Specification document(s): [I-D.ARC]

Related information: [RFC6376]

9. Implementation Status

[[ Note to the RFC Editor: Please remove this section before publication along with the reference to [RFC6982]. ]]

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC6982]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC6982], "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".
This information is known to be correct as of the third interoperability test event which was held on 2016-06-17.

9.1. GMail test reflector

Organization: Google

Description: Internal prototype implementation with both debug analysis and validating + sealing pass-through function

Status of Operation: Beta

Coverage: Full spec implemented as of [ARC-DRAFT]

Licensing: Proprietary - Internal only

Implementation Notes: Full functionality was demonstrated during the interop testing on 2016-06-17

In place for reporting usage only as of 2016-11-21 on all GMail flows.

Contact Info: arc-discuss@dmarc.org [1]

9.2. AOL test reflector and internal tagging

Organization: AOL

Description: Internal prototype implementation with both debug analysis and validating + sealing pass-through function

Status of Operation: Alpha

Coverage: ARC chain validity status checking is not operational, but otherwise this system conforms to [ARC-DRAFT]

Licensing: Proprietary - Internal only

Implementation Notes: Full functionality with the exception of chain validity checking was demonstrated during the interop testing on 2016-06-17

Available for production mail via selected account whitelisting for test validation only.

Contact Info: arc-discuss@dmarc.org [2]
9.3. dkimpy patch

Organization: OAR-DEV

Description: Patches to the dkimpy (Python DKIM) package
[https://code.launchpad.net/~dkimpy-hackers/dkimpy/trunk] to add ARC
functionality

Status of Operation: Beta

Coverage: The test suite is incomplete, but the command line
validator was demonstrated to interoperate with the Google and AOL
implementations during the interop on 2016-06-17

Licensing: Open/Other (same as dkimpy package)

Implementation Notes: The patch has been submitted to the dkimpy
maintainer for inclusion into the official package.

Released to the community as of 2016-11-12 at
[https://code.launchpad.net/~dkimpy-hackers/dkimpy/trunk]

Contact Info: arc-discuss@dmarc.org [3]

9.4. OpenARC

Organization: TDP/Murray Kucherawy

Description: Implementation of milter functionality related to the
OpenDKIM and OpenDMARC packages

Status of Operation: Alpha

Coverage: Built to support [ARC-DRAFT]

Licensing: Open/Other (same as OpenDKIM and OpenDMARC packages)

Implementation Notes: The build is FreeBSD oriented and takes some
tweaks to build on RedHat-based Linux platforms. Initial testing
during the interop event on 2016-06-17 showed that it can be
operational, but the documentation regarding configuration settings
is unclear and the generated signature values do not validate when
compared to the Google, AOL or dkimpy implementations.

Contact Info: arc-discuss@dmarc.org [4]
9.5. Mailman addition

Organization: Mailman development team

Description: Integrated ARC capabilities within the Mailman package

Status of Operation: Implementation in progress

Coverage: Unknown

Licensing: Same as mailman package - GPL

Implementation Notes: Incomplete at this time

Contact Info: [https://www.gnu.org/software/mailman/contact.html]

9.6. Copernica/MailerQ web-based validation

Organization: Copernica

Description: Web-based validation of ARC-signed messages

Status of Operation: Beta

Coverage: Built to support [ARC-DRAFT]

Licensing: On-line usage only,

Implementation Notes: Released 2016-10-24

Requires full message content to be pasted into a web form found at [http://arc.mailerq.com/] (warning - https is not supported).

An additional instance of an ARC signature can be added if one is willing to paste a private key into an unsecured web form.

Initial testing shows that results match the other implementations listed in this section.

Contact Info: [https://www.copernica.com/]

10. Security Considerations

The Security Considerations of [RFC6376] and [RFC7601] apply directly to this specification.
Inclusion of ARC sets in the header of emails may cause problems for some older or more constrained MTAs if they are unable to accept the greater size of the header.

Operators who receive a message bearing N ARC sets has to complete N+1 DNS queries to evaluate the chain (barring DNS redirection mechanisms which can increase the lookups for a given target value). This has at least two effects:

1. An attacker can send a message to an ARC participant with a concocted sequence of ARC sets bearing the domains of intended victims, and all of them will be queried by the participant until a failure is discovered.

2. DKIM only does one DNS check per signature, while this one can do many. Absent caching, slow DNS responses can cause SMTP timeouts; this could be exploited as a DoS attack.

10.1. Message Content Suspicion

Recipients are cautioned to treat messages bearing ARC sets with the same suspicion that they apply to all other email messages. This includes appropriate content scanning and other checks for potentially malicious content. The handlers which are identified within the ARC-Seal chain may be used to provide input to local policy engines in cases where the sending system’s DKIM-Signature does not validate.

11. References

11.1. Normative References


11.2. Informative References

[ARC-DRAFT]

[ARC-USAGE]

[DMARC-INTEROP]

[ENHANCED-STATUS]
11.3. URIs

[1] mailto:arc-discuss@dmarc.org
[2] mailto:arc-discuss@dmarc.org
[3] mailto:arc-discuss@dmarc.org
[4] mailto:arc-discuss@dmarc.org
[5] mailto:dmarc@ietf.org
[6] mailto:arc-discuss@dmarc.org

### Appendix A

#### Appendix A - Example Usage (Obsolete but retained for illustrative purposes)

[[ Note: The following examples were mocked up early in the definition process for the spec. They no longer reflect the current definition and need various updates. ]]

A.1. Example 1: Simple mailing list

A.1.1. Here’s the message as it exits the Origin:
Return-Path: <jqd@d1.example>
Received: from [10.10.10.131] (w-x-y-z.dsl.static.isp.com [w.x.y.z])
   (authenticated bits=0)
   by segv.d1.example with ESMTP id t0FN4a8O084569;
   Thu, 14 Jan 2015 15:00:01 -0800 (PST)
   (envelope-from jqd@d1.example)
DKIM-Signature: v=1; a=rsa-sha256; c=relaxed/simple; d=d1.example;
   s=20130426; t=1421363082;
   bh=EoJqaaRvhrngQxmQ3VnRIIMRBgecuKf1pdkxtfGyWaU=;
   h=Message-ID:Date:From:MIME-Version:To:CC:Subject:Content-Type:
      Content-Transfer-Encoding;
   b=HxsvPubDE+R96v9dM9Y7V3dJUXvajd6rvF5ec5BPe/vpVBRJnD4I2weEIyYijrvQw
      bv9uUAlt94kMNQQ+haFo6hiQPnkuDxku5+oxyZWOqtNH7CTMgcBWWTp4QD4Gd3TRJl
      gotsX4RkbNcUhlfncQO+p+CywWjieI8aR6eof6WDQ=
Message-ID: <54B84785.1060301@d1.example>
Date: Thu, 14 Jan 2015 15:00:01 -0800
From: John Q Doe <jqd@d1.example>
To: arc@dmarc.org
Subject: Example 1

Hey gang,
This is a test message.
--J.

A.1.2. Message is then received at example.org

A.1.2.1. Example 1, Step A: Message forwarded to list members

   Processing at example.org:
   o example.org performs authentication checks
   o No previous Auth-Results or ARC-Seal headers are present
   o example.org adds ARC-Author-Results header
   o example.org adds Received: header
   o example.org adds a ARC-Seal header

Here’s the message as it exits example.org:
Hey gang,
This is a test message.
--J.
A.1.3. Example 1: Message received by Recipient

Let’s say that the Recipient is example.com

Processing at example.com:

- example.com performs usual authentication checks
- example.com adds Auth-Results: header, Received header
- Determines that message fails DMARC
- Checks for ARC-Seal: header; finds one
- Validates the signature in the ARC-Seal: header, which covers the ARC-Authentication-Results: header
- example.com can use the ARC-Authentication-Results values or verify the DKIM-Signature from lists.example.org

Here’s what the message looks like at this point:

Return-Path: <jqd@d1.example>
Received: from example.org (example.org [208.69.40.157])
    by clothilde.example.com with ESMTP id
d200mr22663000ykb.93.1421363207
    for <fmartin@example.com>; Thu, 14 Jan 2015 15:02:40 -0800 (PST)
Authentication-Results: clothilde.example.com; spf=fail
    smtp.from=jqd@d1.example; dkim=pass (1024-bit key)
    header.i=@example.org; dmarc=fail; arc=pass
ARC-Seal: i=1; a=rsa-sha256; t=1421363107; s=seal2015; d=example.org; cv=none;
b=pCw3Qxqfs9E1qnyNZ+cTTF3KHgA+jWwZz++Rju0BceSiuwIg0Pkk+3RZH/kaiz61
    TX6RVT64gs417mujiOR5R661ahL1QJZ/YfdZ3NImCU52gFWLUD7L69
    EU8TzypfkUhscqXjoJgDwji6ceBNNOfh3Jy+V8hQ2rkVFCwO0A=
ARC-Message-Signature: i=1; a=rsa-sha256; c=relaxed/relaxed;
    d=example.org; s=clochette; t=1421363105;
b=FjQYm3HhxSutauz4VUc02o55ExATNFl4uBvEoy7k3s=;
    h=List-Id:List-Unsubscribe:List-Archive:List-Post:
    List-Help:List-Subscribe:Reply-To:DKIM-Signature;
    b=Wb4EiVANwAx8obWrRpmlhxmd1vijdv0psIkiaGOOug32ITA4c74/iWv1PXPxF
    1F5vYVF00w5cmKQa824tKkU0E3yinTakkqyly7GJuFCDeSA1FQHhStVV7BzAr3
    A+m4bwa6RlDr3rOPj1i678d2THfztFDyjwiUx5AkJj/M=
Received: from segv.d1.example (sevg.d1.example [72.52.75.15])
    by lists.example.org (8.14.5/8.14.5) with ESMTP id tOEKaNU9010123
    for <arc@example.org>; Thu, 14 Jan 2015 15:01:30 -0800 (PST)
    (envelope-from jqd@d1.example)
ARC-Authentication-Results: i=1; lists.example.org;
Hey gang,
This is a test message.
--J.

A.2. Example 2: Mailing list to forwarded mailbox

A.2.1. Here’s the message as it exits the Origin:
A.2.2. Message is then received at example.org

A.2.2.1. Example 2, Step A: Message forwarded to list members

Processing at example.org:

- example.org performs authentication checks
- example.org applies standard DKIM signature
- No previous Auth-Results or ARC-Seal headers are present
- example.org adds ARC-Author-Results header
- example.org adds usual Received: header
- example.org adds a ARC-Seal header

Here’s the message as it exits Step A:
Hey gang,
This is a test message.
--J.
A.2.2.2. Example 2, Step B: Message from list forwarded

The message is delivered to a mailbox at gmail.com

Processing at gmail.com:

- gmail.com performs usual authentication checks
- gmail.com adds Auth-Results: and Received: header
- Determines that message fails DMARC
- Checks for ARC-Seal: header; finds one
- Validates the signature in the ARC-Seal: header, which covers the ARC-Authentication-Results: header
- Uses the ARC-Auth-Results: values, but:
  - Instead of delivering message, prepares to forward message per user settings
  - Applies usual DKIM signature
- gmail.com adds it’s own ARC-Seal: header, contents of which are
  - version
  - sequence number ("i=2")
  - hash algorithm (SHA256 as example)
  - timestamp ("t=")
  - selector for key ("s=notary01")
  - domain for key ("d=gmail.com")
  - headers included in hash ("h=ARC-Authentication-Results:ARC-Seal")
  - Note: algorithm requires only ARC-Seals with lower sequence # be included, in ascending order
  - signature of the header hash

Here’s what the message looks like at this point:

Return-Path: <jqd@d1.example>
Hey gang,

This is a test message.

--J.

**A.2.3. Example 2: Message received by Recipient**

Let’s say that the Recipient is example.com

Processing at example.com:

- example.com performs usual authentication checks
- example.com adds Auth-Results: header, Received header
- Determines that message fails DMARC
- Checks for ARC-Seal: header; finds two
- Validates the signature in the highest numbered ("i=2") ARC-Seal: header, which covers all previous ARC-Seal: and ARC-Authentication-Results: headers
- Validates the other ARC-Seal header ("i=1"), which covers the ARC-Authentication-Results: header
- example.com uses the ARC-Authentication-Results: values

Here’s what the message looks like at this point:
Authentication-Results: clothilde.example.com; spf=fail
smtp.from=jqd@d1.example; dkim=pass (1024-bit key)
header.i=@gmail.com; dmarc=fail; arc=pass

ARC-Seal: i=2; a=rsa-sha256; t=1421363253;
s=notary01; d=gmail.com; cv=pass;
b=sjHDMri1RZ0Mu5eVEOGscRHWbQHcy971vrduHQ8h+f2CFIrXuIKOE44x3LQwDWR
YbDjf5fcM9MdcIahc+Cp59BQ9Y9DhwMDzwRTn7NVb4kY+t5aVnLoIOaP91F/sut
tx0+RRN=0FCFw==

ARC-Message-Signature: i=2; a=rsa-sha256; c=relaxed/relaxed;
d=gmail.com; s=20120806;

Received: by mail-yk0-f179.google.com with SMTP id 19so2728865ykq.10
for <mailbox@gmail.com>; Thu, 14 Jan 2015 15:02:45 -0800 (PST)
Authentication-Results: i=2; gmail.com; spf=fail
smtp.from=jqd@d1.example; dkim=pass (1024-bit key)
header.i=@example.org; dmarc=fail; arc=pass

ARC-Seal: i=1; a=rsa-sha256; t=1421363107;
s=seal2015; d=example.org; cv=none;
b=pCw3Qxgfs9E1qnyNZ+CTTF3KHgAjJwWzz+Rju0BceSiuwIQpkk+3RZH/kaiz61
TX6VT6E4gs49Sstp41KmujiOR5R6Ql1ahL1QJZ/YFzD3NIMC5U2gFWLUD7l66
EU8TZypfkUhscxQjoJqDwjeiBNOOofh3Jy+V8hQ2rVFCW0b8Ol1ebYV/hIbmfrShLF1E80hMPcMiOqJQ865bOh/kE6N2fqp6aSnLG/3WJ+i31EhXVh1GJfEmE
KdjqiW5cxqgPTM+RbN5ee6Tzg69kr625NTDIAU8p8fQNUfJ349MMA+QwDBJtXw
bOoZyRtb6X6qOyYaszUI8kw==

Received: from segv.d1.example [72.52.75.15] by lists.example.org (8.14.5/8.14.5) with ESMTP id t0EKaNU9010123
for <arc@example.org>; Thu, 14 Jan 2015 15:01:30 -0800 (PST)

ARC-Message-Signature: i=1; a=rsa-sha256; c=relaxed/relaxed;
d=example.org; s=clochette; t=1421363105;
b=FjoYm3HxSTuauzV4UC02o555EzATNF4u4bVbEcX7k3s==;

---

Received: from seqv.d1.example (seqv.d1.example [72.52.75.15])
by lists.example.org (8.14.5/8.14.5) with ESMTP id t0EKaNU9010123
for <arc@example.org>; Thu, 14 Jan 2015 15:01:30 -0800 (PST)
(envelope-from jqd@d1.example)

ARC-Authentication-Results: i=1; lists.example.org;
spf=pass smtp.mfrom=jqd@d1.example;
dkim=pass (1024-bit key) header.i=@example.org; dmarc=pass

Received: from 10.10.10.131 (w-x-y-z.dsl.static.isp.com [w.x.y.z])

---

A.3. Example 3: Mailing list to forwarded mailbox with source

A.3.1. Here’s the message as it exits the Origin:
Hey gang,
This is a test message.
--J.

A.3.2. Message is then received at example.org

A.3.2.1. Example 3, Step A: Message forwarded to list members with source

Processing at example.org:

- example.org performs authentication checks
- example.org applies standard DKIM signature
- Checks for ARC-Seal: header; finds one (i=1)
- Validates the signature in the ARC-Seal (i=1): header, which covers the d1.example ARC-Message-Signature: header
- example.org adds ARC-Auth-Results header
- example.org adds usual Received: header

Message-ID: <54B84785.1060301@d1.example>
o example.org adds a DKIM-Signature

o example.org adds a ARC-Seal header, contents of which are
  * sequence number ("i=2")
  * hash algorithm (SHA256 as example)
  * timestamp ("t=")
  * chain validity ("cv=")
  * selector for key ("s=seal2015")
  * domain for key ("d=example.org")
  * signature ("b=")

Here’s the message as it exits Step A:
Hey gang,
This is a test message.
--J.

A.3.2.2. Example 3, Step B: Message from list forwarded with source

The message is delivered to a mailbox at gmail.com
Processing at gmail.com:

- gmail.com performs usual authentication checks
- gmail.com adds Auth-Results: and Received: header
- Determines that message fails DMARC
- Checks for ARC-Seal: header; finds two
  - Validates the signature in the ARC-Seal (i=2): header, which covers the ARC-Authentication-Results: header
  - Validates the signature in the ARC-Seal (i=1): header, which covers the d1.example ARC-Message-Signature: header
- Uses the ARC-Auth-Results: values, but:
  - Instead of delivering message, prepares to forward message per user settings
  - Applies usual DKIM signature

- gmail.com adds its own ARC-Seal: header, contents of which are
  - version
  - sequence number ("i=2")
  - hash algorithm (SHA256 as example)
  - timestamp ("t=")
  - selector for key ("s=notary01")
  - domain for key ("d=gmail.com")

  * Note: algorithm requires only ARC-Seals with lower sequence # be included, in ascending order
  * signature of the chain

Here’s what the message looks like at this point:

Return-Path: <jqd@d1.example>
ARC-Seal: i=3; a=rsa-sha256; t=1421363253;
s=notary01; d=gmail.com; cv=pass;
b=sjHDrirRZ0Mui5eVEOGscRHwbdQHcy971vrduHQ8h+f2CfIrUXuKO/E4x3LqQwD
WRYbdjf5echM9MdcIahc+cPs598QY999WMDzwTRmM7Nvbc4kY+tSaVnLoIOap91F
/suttxo+RRN0rfCFw==
ARC-Message-Signature: i=3; a=rsa-sha256; c=relaxed/relaxed;
d=gmail.com; s=20120806;
h=mime-version:content-type:x-original-sender
:x-original-authentication-results:precedence:mailing-list
:list-id:list-post:list-help:list-archive:sender
:list-unsubscribe:reply-to;
bh=2+gwz2hUKV2V77bopo02MTrU19/gvhaA4NjiohFo92/Z9/g=
b=pCw3qkgfs9ElqnyNZ+cTTF3KHgAjWwZzz+RJu0BceSiuwIg0Pkk+3RZh/kaiz6
1X6RT64gs49Stp41K7mujo1R5R6Q611ahL1QJ2/YfDZ3NImC52gFWложения7L
69E8UZyvphUhsqc8juOJgDwJiceBNNoh3Jy+V8hQzFVCEwOAb80i1ebVY/h1Bm
fslf1E80hMPCMiJoF9Q6g5Sho/kE6N2g6aSngL/WA3+g3id8EhXhVfC/gC
RFemKdQji5w5cxqdpTfWR+BNr5e6e5t706kr265NTDIAU88fQNuL7fj49MA+QwD
BfJtwbQo2yRtb6X6q0MhYnZwUBkw==
Received: by mail-yk0-f179.google.com with SMTP id 19so2728865ykq.10
for <mailbox@gmail.com>; Thu, 14 Jan 2015 15:02:45 -0800 (PST)
Authentication-Results: i=3; gmail.com; spf=fail
smtp.from=jqd@d1.example; dkim=pass (1024-bit key)
header.i=@example.org; dmarc=fail; arc=pass
ARC-Seal: i=2; a=rsa-sha256; t=1421363107;
s=seal2015; d=example.org; cv=pass;
b=pCw3qkgfs9ElqnyNZ+cTTF3KHgAjWwZzz+RJu0BceSiuwIg0Pkk+3RZh/kaiz61
1X6RT64gs49Stp41K7mujo1R5R6Q611ahL1QJ2/YfDZ3NImC52gFWложения7L
69E8UZyvphUhsqc8juOJgDwJiceBNNoh3Jy+V8hQzFVCEwOAb80i1ebVY/h1Bm
fslf1E80hMPCMiJoF9Q6g5Sho/kE6N2g6aSngL/WA3+g3id8EhXhVfC/gC
RFemKdQji5w5cxqdpTfWR+BNr5e6e5t706kr265NTDIAU88fQNuL7fj49MA+QwD
BfJtwbQo2yRtb6X6q0MhYnZwUBkw==
ARC-Message-Signature: i=2; a=rsa-sha256; c=relaxed/relaxed;
d=example.org; s=clochette; t=1421363105;
bh=FjQYm3HhXStuzauszV4Uc0s55EZATNflU4lvEcy7k3s=
List-Id:List-Unsubscribe:List-Archive:List-Post:
List-Help:List-Subscribe:Reply-To:DKIM-Signature;
b=Wb4EiVANwX8ObwWrRWpmNhxmdTvjdOVpsIkaGOOPg321TAc774/1Vv1PXeF1
F5vVVF0m6cmkOa8z4KkuOEnE3inTakqny77GJuFCDesal1fQHSHS7V7Bzrs3A+
m4bw6IRjDr3rOPJ1678827HTjz7FWyjwIUx5Ajsx/M=
Received: from segv.d1.example (segv.d1.example [72.52.75.15])
by lists.example.org (8.14.5/8.14.5) with ESMTP id t0EKaNU9010123
for <arc@example.org>; Thu, 14 Jan 2015 15:01:30 -0800 (PST)
Authentication-Results: i=2; lists.example.org;
spf=pass smtp.mfrom=jqd@d1.example;
dkim=pass (1024-bit key) header.i=@d1.example; dmarc=pass
ARC-Message-Signature: i=2; a=rsa-sha256; c=relaxed/relaxed;
d=example.org; s=clochette; t=1421363105;
bh=FjQYm3HhXStuzauszV4Uc0s55EZATNflU4lvEcy7k3s=
List-Id:List-Unsubscribe:List-Archive:List-Post:
List-Help:List-Subscribe:Reply-To:DKIM-Signature;
b=Wb4EiVANwX8ObwWrRWpmNhxmdTvjdOVpsIkaGOOPg321TAc774/1Vv1PXeF1
F5vVVF0m6cmkOa8z4KkuOEnE3inTakqny77GJuFCDesal1fQHSHS7V7Bzrs3A+
m4bw6IRjDr3rOPJ1678827HTjz7FWyjwIUx5Ajsx/M=
Received: from segv.d1.example [72.52.75.15]
by lists.example.org (8.14.5/8.14.5) with ESMTP id t0EKaNU9010123
for <arc@example.org>; Thu, 14 Jan 2015 15:01:30 -0800 (PST)
(Envelope from jqd@d1.example)
ARC-Authentication-Results: i=2; lists.example.org;
spf=pass smtp.mfrom=jqd@d1.example;
dkim=pass (1024-bit key) header.i=@d1.example; dmarc=pass
Received: from [10.10.10.131] (w-x-y-z.dsl.static.isp.com [w.x.y.z])
(authenticated bodies=0)
by segv.d1.example with ESMTP id t0FN4a80084569;
Thu, 14 Jan 2015 15:00:01 -0800 (PST)
Hey gang,
This is a test message.
--J.

A.3.3. Example 3: Message received by Recipient

Let’s say that the Recipient is example.com
Processing at example.com:
  o example.com performs usual authentication checks
  o example.com adds Auth-Results: header, Received header
  o Determines that message fails DMARC
  o Checks for ARC-Seal: header; finds three
    o Validates the signature in the highest numbered ("i=2") ARC-Seal: header, which covers all previous ARC-Seal: and ARC-Authentication-Results: headers
    o Validates the other ARC-Seal header ("i=2"), which covers the ARC-Authentication-Results: header
    o Validates the other ARC-Seal header ("i=1"), which covers the d1.example ARC-Message-Signature: header
  o example.com uses the ARC-Authentication-Results: values
Here’s what the message looks like at this point:

Return-Path: <jqd@d1.example>
Received: from mail-ob0-f188.google.com (mail-ob0-f188.google.com [208.69.40.157]) by clothilde.example.com with ESMTP id d200mr22664000ykb.93.1421363268 for <fmartin@example.com>; Thu, 14 Jan 2015 15:03:15 -0800 (PST)
Authentication-Results: clothilde.example.com; spf=fail
smtp.from=jqd@d1.example; dkim=pass (1024-bit key)
header.i=;d=gmail.com; dmarc=fail; arc=pass
ARC-Seal: i=3; a=rsa-sha256; t=1421363253;
s=notary01; d=gmail.com; cc=pass;
b=sJHDMrirZ0Mu5eVEOGscRhWbQHcy971vruDHQ8h+02CFIrXuiK0E44h3LQwDW
RYpDj5f5MC9MdcIahC+CP59BQ9YYD9wMDzwtN7NVb4kY+sTaVnLo10a9P1/f
uttzo+RNRn0FCFw==
ARC-Message-Signature: i=3; a=rsha-sha256; c=relaxed/relaxed;
d@gmail.com; s=20120806;
h=mime-version:content-type:x-original-sender
:x-original-authentication-results:precedence
:mailing-list:list-id:list-post:list-help:list-archive:sender
:list-unsubscribe:reply-to;
bh=2gq2whJIKZVdJpbo2OMTRu19WvhAc4NjjiobEm9Z2/g=;
b=pCw3Qxgfs69ElqnyNZ+cTTF3KHqAjiwWz++Rju0BceSiuw1gOPkk+3RZH/kaiz6
1TX6rVT6E4s9Stp4IKmujOR5R6Q611aUhL1QJZ/YfDZ3ImCu52gFWLUD7L
69h5tZypfrU6scxqjXJo9jyWl1ceBNNO0fh33y+V8hQ3RvFCW0Ab8O1lebYV/hIBm
fhSLF1E80hMPfcMiONfTQB65hho/kE6N2fph6aSngL/WA3+g3i6EihXHViGcj
RFEmKdJqiW5cxdqPTRW+BnR5ee67qz06kr265NTDIAu8p8fQNuLzj499MA+W0
BjTvxbWoQ0yRtb6X6q0MhYszUB8kw==
Received: by mail-yk0-f179.google.com with SMTP id 19so2728865ykq.10
for <mailbox@gmail.com>; Thu, 14 Jan 2015 15:02:45 -0800 (PST)
Authentication-Results: i=3; gmail.com; spf=fail
smtp.from=jqd@d1.example; dkim=pass (1024-bit key)
header.i=;d=gmail.com; dmarc=fail; arc=pass
ARC-Seal: i=2; a=rsha-sha256; t=1421363107;
s=example.org; d=example.org; cc=pass;
b=pCw3Qxgfs69ElqnyNZ+cTTF3KHqAjiwWz++Rju0BceSiuw1gOPkk+3RZH/kaiz6
1TX6rVT6E4s9Stp4IKmujOR5R6Q611aUhL1QJZ/YfDZ3ImCu52gFWLUD7L
69h5tZypfrU6scxqjXJo9jyWl1ceBNNO0fh33y+V8hQ3RvFCW0Ab=
ARC-Message-Signature: i=2; a=rsha-sha256; c=relaxed/relaxed;
d=example.org; s=clochette; t=1421363105;
h=FjQyM3HHXstuauv4Vu02o55EzATNF4l4BuE0y7k3s=;
List-Id:List-Unsubscribe:List-Archive:List-Post:
List-Next:List-Subscibers:Reply-To:DKIM-Signature;
b=WB4EL3LvAwXa80hWwrRWpmWmdIjv0DvopsIqiaGOOug32i1Acc74/iWv1XPxF
F5vYYV0v5mKOA824ttkU00E3yinTaeKqlny7yGjuFCDeSA1QHhStVv7BzAr3A+
m4bwa6RIDgr3rOPjil678dZTHjzFWyjIUx55Ajxj/M=
Received: from segv.d1.example (segv.d1.example [72.52.75.15])
by lists.example.org (8.14.5/8.14.5) with ESMTP id t0EKeA9U010123
for <arc@example.org>; Thu, 14 Jan 2015 15:01:30 -0800 (PST)
(envelope-from jqd@d1.example)
ARC-Authentication-Results: i=2; lists.example.org;
   spf=pass smtp.mfrom=jqd@d1.example;
   dkim=pass (1024-bit key) header.i=@d1.example;
   dmarc=pass
Received: from [10.10.10.131] (w-x-y-z.dsl.static.isp.com [w.x.y.z])
   (authenticated bits=0)
   by segv.d1.example with ESMTP id t0FN4a80084569;
   Thu, 14 Jan 2015 15:00:01 -0800 (PST)
   (envelope-from jqd@d1.example)
ARC-Seal: i=1; a=rsa-sha256; t=1421363107;
   s=origin2015; d=d1.example; cv=none;
   b=pCw3QqxfS9ElgnyN2+cTTF3KHgAjWwZz++Rju0BceSiuwIg0Pkk+3RZH/kaiz61
   TX6RTt6E4gs49Stp41K7mu10R5Q6Q61lahL1QJZ/YfD2NImCUS2gFwUd7L69
   EUBTzpfxUhsqgXjOjTqDwjIcBNN0fh3Jy+V8hQZrVFCw0A=
ARC-Message-Signature: i=1; a=rsa-sha256; c=relaxed/relaxed;
   d=d1.example; s=20130426; t=1421363082;
   bh=EoJqaaRvhrQoxmQVnRIIMRBgecuKf1pdfktfGyWaU=;
   h=MIME-Version:To:CC:Subject:Content-Type:Content-Transfer-Encoding;
   b=HxsvPubDE+E96v9dM9Y7V3dJUXvaJd6rvF5ec5BPe/vpVBRJnD4I2weEIyYijr
   vQwbb9UlAit94kMN0Q+haPo6h1QNkuDxku5+oxyZWoqtNH7CTMgcBWWTp4QD4G
   d3TRJlgotsX4XkNaOshfnoQ0p+CywWjie18aR6ecf6WDQ=
Message-ID: <54B84785.1060301@d1.example>
Date: Thu, 14 Jan 2015 15:00:01 -0800
From: John Q Doe <jqd@d1.example>
To: arc@example.org
Subject: [Lists] Example 1

Hey gang,
This is a test message.
--J.

Appendix B. Acknowledgements

This draft is the work of OAR-Dev Group.

The authors thank all of the OAR-Dev group for the ongoing help and
though-provoking 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.

Grateful appreciation is extended to the people who provided feedback
through the discuss mailing list.
Appendix C. Comments and Feedback

Please address all comments, discussions, and questions to dmarc@ietf.org [5]. Earlier discussions can be found at arc-discuss@dmarc.org [6].

Authors' Addresses

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

Email: kurta@linkedin.com

John Rae-Grant (editor)
Google

Email: johnrg@google.com

Brandon Long (editor)
Google

Email: blong@google.com

J. Trent Adams (editor)
Paypal

Email: trent.adams@paypal.com

Steven Jones (editor)
TDP

Email: smj@crash.com