DMARC Working Group
Internet-Draft
Intended status: Experimental
Expires: January 18, 2019

K. Andersen
LinkedIn
B. Long, Ed.
Google
S. Blank, Ed.
Valimail
M. Kucherawy, Ed.
TDP
T. Draegen, Ed.
dmarcian

July 17, 2018

Authenticated Received Chain (ARC) Protocol
draft-ietf-dmarc-arc-protocol-16

Abstract

The Authenticated Received Chain (ARC) protocol allows Internet Mail Handlers to attach assertions of message authentication state to individual messages. As messages traverse ARC-enabled Internet Mail Handlers, additional ARC assertions can be attached to messages to form ordered sets of ARC assertions that represent authentication state along each step of message handling paths.

ARC-enabled Internet Mail Handlers can process sets of ARC assertions to inform message disposition decisions, to identify Internet Mail Handlers that might break existing authentication mechanisms, and to convey original authentication state across trust boundaries.

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 January 18, 2019.

Table of Contents

1. Introduction

The utility of widely deployed email authentication technologies such as Sender Policy Framework (SPF) [RFC7208] and DomainKeys Identified Mail (DKIM) [RFC6376] is impacted by the processing of Internet Mail by intermediate handlers. This impact is thoroughly documented in the defining documents for SPF and DKIM and further discussed in [RFC6377] and [RFC7960].

The utility of technologies that build upon SPF and DKIM (such as DMARC [RFC7489]) is similarly impacted by intermediate handlers. The disruption of authentication mechanisms for legitimate messages by intermediate handlers can impact all aspects of Internet Mail - message authors, message recipients, and even the intermediary handler itself.

Authenticated Received Chain (ARC) creates a mechanism for individual Internet Mail Handlers to add their authentication processing results to a message's ordered set of processing results. ARC encapsulates processing results in a DKIM signature derivative to grant other handlers the ability to verify the authenticity of the individual processing results as well as the aggregate set and sequence of results.

Ordered sets of processing results can be used by ARC-enabled Internet Mail Handlers to inform message handling disposition, to identify where alteration of message content might have occurred, and to provide additional trace information for use in understanding message handling paths.

2. General Concepts

ARC is loosely based on concepts from evidence collection. Evidence is usually collected, labeled, stored, and transported in specific ways to preserve the state of evidence and to document all processing steps.

2.1. Evidence

In ARC's situation, the "evidence" is a message's authentication state at any point along the delivery path between origination and final delivery. Authentication state can change when intermediate handlers modify message content (headers and/or body content), route messages through unforeseen paths, or change envelope information.

The authentication state of a message is determined upon receipt of a message and documented in the Authentication-Results header field(s). ARC extends this mechanism to survive transit through intermediary ADMDs.

2.2. Custody

"Custody" refers to when an ARC-enabled Internet Mail Handler processes a message. When a handler takes custody of a message, the handler becomes a Custodian and attaches their own evidence (authentication state upon receipt) to the message. Evidence is added in such a way so that future handlers can verify the authenticity of both evidence and custody.

2.3. Chain of Custody

The "chain of custody" of ARC is the entire set of evidence and custody that travels with a message.

2.4. Validation of Chain of Custody

Any ARC-enabled Internet Mail Handler can validate the entire set of evidence and custody to yield a valid Chain of Custody. If the evidence-supplying Custodians can be trusted, then the validated Chain of Custody describes the (possibly changing) authentication state as the message traveled through various Custodians.

Even though a message's authentication state might have changed, the validated chain of custody can be used to determine if the changes (and the Custodians responsible for the changes) can be tolerated.

3. Terminology and Definitions

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

Readers should to be familiar with the contents, core concepts, and definitions found in [RFC5598]. The potential roles of intermediaries in the delivery of email are directly relevant.

Language, syntax (including some ABNF constructs), and concepts are imported from DKIM [RFC6376]. Specific references to DKIM are made throughout this document. The following terms are imported from [RFC5598]:

- ADministrative Management Domain (ADMD), Section 2.3
- Message Transfer Agents (MTA), Section 4.3.2
- Message Submission Agent (MSA), Section 4.3.1
- Message Delivery Agent (MDA), Section 4.3.3

Internet Mail Handlers process and deliver messages across the Internet and include MSAs, MTAs, MDAs, gateways, and mailing lists.

Syntax descriptions use Augmented BNF (ABNF) [RFC5234] and [RFC7405].

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here. These words may also appear in this document in lower case as plain English words, absent their normative meanings.

3.1. ARC Set

Section 4.1 introduces three (3) ARC header fields. Together, the 3 header fields compose a single "ARC Set". An ARC Set provides the means for an Internet Mail Handler to attach authentication state to a message in a manner that can be verified by future handlers. A single message can contain multiple ARC Sets.

In General Concept terms, an ARC Set represents Evidence and Custody.

3.2. Authenticated Received Chain (ARC)

The complete sequence of ARC Sets attached to a message is called the Authenticated Received Chain. An Authenticated Received Chain is a recording of individual authentication states as a message traverses through ARC-participating ADMDs.

The first attachment of an ARC Set to a message causes an Authenticated Received Chain to be created. Additional attachments of ARC Sets cause the Authenticated Received Chain to be extended.

In General Concept terms, an Authenticated Received Chain represents Chain of Custody.

3.3. Sealer

A Sealer is an Internet Mail Handler that attaches a complete and valid ARC Set to a message.

In General Concept terms, a Sealer adds Evidence and proof of Custody to the Chain of Custody.

3.4. Validator

A Validator is an ARC-enabled Internet Mail Handler that evaluates an Authenticated Received Chain for validity and content. The process of evaluation of the individual ARC Sets that compose an Authenticated Received Chain is described in Section 5.2.

In General Concept terms, a Validator inspects the Chain of Custody to determine the content and validity of individual Evidence supplied by Custodians.

3.5. Imported ABNF Tokens

The following ABNF tokens are imported:

- tag-list ([RFC6376] section 3.2)
- authres-payload ([I-D-7601bis] section 2.2)
- cfws ([RFC5322] section 3.2.2)

3.6. Common ABNF Tokens

The following ABNF tokens are used elsewhere in this document:

```
position = 1*2DIGIT ; 1 - 50
instance = [CFWS] %s"i" [CFWS] "=" [CFWS] position [CFWS] ";"
chain-status = ("none" / "fail" / "pass")
seal-cv-tag = %s"cv" [CFWS] "=" [CFWS] chain-status
```

4. Protocol Elements

4.1. ARC Headers

ARC introduces three new header fields. Syntax for new header fields borrows heavily from existing specifications. This document only

describes where ARC-specific changes in syntax and semantics differ from existing specifications.

4.1.1. ARC-Authentication-Results (AAR)

The ARC-Authentication-Results (AAR) header field records the message authentication state as processed by an ARC-participating ADMD at message arrival time.

In General Concept terms, the AAR header field is where Evidence is recorded by a Custodian.

The AAR header field is similar in syntax and semantics to an Authentication-Results field [I-D-7601bis], with two (2) differences:

- the name of the header field itself;
- the presence of the "instance tag". Additional information on the "instance tag" can be found in Section 4.2.1.

The formal ABNF for the AAR header field is:

```
arc-info = instance [CFWS] ";" authres-payload
arc-authres-header = "ARC-Authentication-Results:" [CFWS] arc-info
```

Because there is only one AAR allowed per ARC set, the AAR MUST contain all authentication results from within the participating ADMD, regardless of how many Authentication-Results headers are attached to the message.

4.1.2. ARC-Message-Signature (AMS)

The ARC-Message-Signature (AMS) header field allows an ARC-participating ADMD to convey some responsibility (custodianship) for a message and possible message modifications to future ARC-participating Custodians.

In General Concept terms, the AMS header field identifies a Custodian.

The AMS header field is similar in syntax and semantics to a DKIM-Signature field [RFC6376], with three (3) differences:

- the name of the header field itself;
- no version tag ("v") is defined for the AMS header field. As required for undefined tags (in [RFC6376]), if seen, a version tag MUST be ignored;

- the presence of the "instance tag". Additional information on the "instance tag" can be found in Section 4.2.1. The instance tag replaces the DKIM "AUID" tag;

- when building the header field list to be signed, ARC-related headers MUST be submitted to the hash function in increasing instance order.

ARC places no requirements on the selectors and/or domains used for the AMS header field signatures.

The formal ABNF for the AMS header field is:

```
arc-ams-info = instance [CFWS] ";" tag-list
arc-message-signature = "ARC-Message-Signature:" [CFWS] arc-ams-info
```

To avoid unwanted invalidation of AMS signatures:

- AMS header fields are added by ARC-participating ADMDs as messages exit the ADMD. AMS header fields SHOULD be attached so that any modifications made by the ADMD are included in the signature of the AMS header field.

- Authentication-Results header fields MUST NOT be included in AMS signatures as they are likely to be deleted by downstream ADMDs (per [I-D-7601bis] Section 5).

- ARC-related header fields (ARC-Authentication-Results, ARC-Message-Signature, ARC-Seal) MUST NOT be included in the list of header fields covered by the signature of the AMS header field.

To preserve the ability to verify the integrity of a message, the signature of the AMS header field SHOULD include any DKIM-Signature header fields already present in the message.

4.1.3. ARC-Seal (AS)

The ARC-Seal (AS) header field is the mechanism by which ARC-participating ADMDs can verify the integrity of AAR header fields and corresponding AMS header fields.

In General Concept terms, the AS header field is how Custodians bind Evidence into a Chain of Custody so that Validators can inspect individual Evidence and Custodians.

The AS header field is similar in syntax and semantics to DKIM-Signatures [RFC6376], with the following differences:

- the presence of the "instance tag". Additional information on the "instance tag" can be found in Section 4.2.1.

- the signature of the AS header field does not cover the body of the message and therefore there is no 'bh' tag. The signature of the AS header field only covers specific header fields as defined in Section 5.1.1.

- no body canonicalization is performed as the AS signature does not cover the body of a message.

- only "relaxed" header canonicalization ([RFC6376] section 3.4.2) is used.

- the only supported tags are "i" (from Section 4.2.1 of this document), and "a", "b", "d", "s", "t" from [RFC6376] Section 3.5. Note especially that the DKIM "h" tag is NOT allowed and if found, MUST result in a cv status of "fail" (for more information see Section 5.1.1);

- an additional tag, "cv" ("seal-cv-tag" in the ARC-Seal ABNF definition) is used to communicate Chain Validation Status to subsequent ADMDs.

ARC places no requirements on the selectors and/or domains used for the AS header field signatures.

The formal ABNF for the AS header field is:

```
arc-as-info = instance [CFWS] ";" tag-list
arc-seal = "ARC-Seal:" [CFWS] arc-as-info
```

4.2. ARC Set

An "ARC Set" is a single collection of three ARC Headers (AAR, AMS, and AS). ARC Headers of an ARC Set share the same "instance" value.

By adding all ARC Headers to a message, an ARC Sealer adds an ARC Set to a message. A description of how Sealers add an ARC Set to a message is found in Section 5.1.

4.2.1. Instance Tags

Instance tags describe which ARC Headers belong to an ARC Set. Each ARC Header of an ARC Set shares the same instance tag value.

Instance tag values are integers that begin at 1 and are incremented by each addition of an ARC Set. Through the incremental values of

instance tags, an ARC Validator can determine the order in which ARC Sets were added to a message.

Instance tag values can range from 1-50 (inclusive).

INFORMATIONAL: The upper limit of 50 was picked based on some initial observations reported by early working group members with a safety margin multiple added on top to support the vast majority of all intermediary mail flows.

Valid ARC Sets MUST have exactly one instance of each ARC Header field (AAR, AMS, and AS) for a given instance value and signing algorithm.

INFORMATIONAL: Initial development of ARC is only being done with a single allowed signing algorithm, but parallel work in the DCRUP working group is expanding that. For handling multiple signing algorithms, see [ARC-MULTI].

4.3. Authenticated Received Chain

An Authenticated Received Chain is an ordered collection of ARC Sets. As ARC Sets are enumerated sets of ARC Headers, an Authenticated Received Chain represents the output of message authentication state along the handling path of ARC-enabled processors.

Results of message authentication processing along each step of the ARC-enabled handling path is present in an Authenticated Received Chain in the form of AAR header fields. The ability to verify the identity of message handlers and the integrity of message content is provided by AMS header fields. AS header fields allow messages handlers to validate the assertions, order and sequence of the Authenticated Received Chain itself.

In General Concept terms, an Authenticated Received Chain represents a message's Chain of Custody. Validators can consult a message's Chain of Custody to gain insight regarding each Custodian of a message and the Evidence collected by each Custodian.

4.4. Chain Validation Status

The state of the Authenticated Received Chain at a specific processing step is called the "Chain Validation Status". Chain Validation Status information is communicated in several ways:

- the AS header field in the "cv" tag, and

- as part of Authentication-Results and AAR headers.

Chain Validation Status has one of three possible values:

- none: There was no Authenticated Received Chain on the message when it arrived for validation. Typically this occurs when a message is received directly from a message's original Message Transfer Agent (MTA) or Message Submission Agent (MSA), or from an upstream Internet Mail Handler that is not participating in ARC handling.

- fail: The message contains an Authenticated Received Chain whose validation failed.

- pass: The message contains an Authenticated Received Chain whose validation succeeded.

5. Protocol Actions

ARC-enabled Internet Mail Handlers generally act as both ARC Sealers (when sending messages) and ARC Validators (when receiving messages).

5.1. Sealer Actions

To "seal" a message, an ARC Sealer adds an ARC Set (the three ARC header fields AAR, AMS, and AS) to a message. All ARC header fields in an ARC Set share the same instance tag value.

To perform Sealing (aka to build and attach a new ARC Set), the following actions must be taken by an ARC Sealer when presented with a message:

1. All message modifications (including adding DKIM-Signatures) MUST be performed before Sealing.

2. Calculate the instance value: if the message contains an Authenticated Received Chain, the instance value is 1 more than the highest instance number found in the Authenticated Received Chain. If no Authenticated Received Chain exists, the instance value is 1.

3. Using the calculated instance value, generate and attach to the message in the following order:

4. An ARC-Authentication-Results header field as defined in Section 4.1.1.

5. An ARC-Message-Signature header field as defined in Section 4.1.2.

6. An ARC-Seal header field using the AS definition found in Section 4.1.3, the method described in Section 5.1.1, and the Chain Validation Status as determined during ARC validation.

5.1.1. Header Fields To Include In ARC-Seal Signatures

The signature of an AS header field signs a specific canonicalized form of the ARC Set header values. The ARC set header values are supplied to the hash function in increasing instance order, starting at 1, and include the ARC Set being added at the time of Sealing the message.

Within an ARC Set, header fields are supplied to the hash function in the following order:

1. ARC-Authentication-Results

2. ARC-Message-Signature

3. ARC-Seal

The ARC-Seal is generated in a manner similar to when DKIM-Signatures are added to messages ([RFC6376], section 3.7).

Note that when an Authenticated Received Chain has failed validation, the signing scope for the ARC-Seal is modified (see Section 5.1.2).

5.1.2. Marking and Sealing "cv=fail" (Invalid) Chains

In the case of a failed Authenticated Received Chain, the header fields included in the signature scope of the AS header field b= value MUST only include the ARC Set headers created by the MTA which detected the malformed chain, as if this newest ARC Set was the only set present.

INFORMATIONAL: This approach is mandated to handle the case of a malformed or otherwise invalid Authenticated Received Chain. There is no way to generate a deterministic set of AS header fields (Section 5.1.1) in most cases of invalid chains.

5.1.3. Only One Authenticated Received Chain Per Message

A message can have only one Authenticated Received Chain on it at a time. Once broken, the chain cannot be continued, as the chain of custody is no longer valid and responsibility for the message has been lost. For further discussion of this topic and the designed restriction which prevents chain continuation or re-establishment, see [ARC-USAGE].

5.1.4. Broad Ability to Seal

ARC is not solely intended for perimeter MTAs. Any mediator ([RFC5598], section 5) that modifies a message may Seal its own changes. For additional information, see Section 7.1.

5.1.5. Sealing is Always Safe

The utility of an Authenticated Received Chain is limited to very specific cases. Authenticated Received Chains are designed to provide additional information to an Internet Mail Handler when evaluating messages for delivery in the context of authentication failures. Specifically:

- Properly adding an ARC Set to a message does not damage or invalidate an existing Authenticated Received Chain.
- Sealing an Authenticated Received Chain when a message has not been modified does not negatively affect the chain.
- Validating a message exposes no new threat vectors (see Section 9).
- An ADMD may choose to Seal all inbound messages whether or not a message has been modified or will be retransmitted.

5.1.6. Signing vs Sealing

Signing is the process of affixing a digital signature to a message as a header, such as when a DKIM-Signature (as in [RFC6376] section 2.1), or an AMS or AS is added. Sealing is when an ADMD affixes a complete and valid ARC Set to a message creating or continuing an Authenticated Received Chain.

5.2. Validator Actions

A validator performs the following steps, in sequence, to process an Authenticated Received Chain. Canonicalization, hash functions, and signature validation methods are imported from [RFC6376] section 5.

1. Collect all ARC Sets currently attached to the message. If there are none, the Chain Validation Status is "none" and the algorithm stops here. The maximum number of ARC Sets that can be attached to a message is 50. If more than the maximum number exist the Chain Validation Status is "fail" and the algorithm stops here. In the following algorithm, the maximum ARC instance value is referred to as "N".

2. If the Chain Validation Status of the highest instance value ARC Set is "fail", then the Chain Validation status is "fail" and the algorithm stops here.

3. Validate the structure of the Authenticated Received Chain. A valid ARC has the following conditions:

    1. Each ARC Set MUST contain exactly one each of the three ARC header fields (AAR, AMS, and AS).

    2. The instance values of the ARC Sets MUST form a continuous sequence from 1..N with no gaps or repetition.

    3. The "cv" value for all ARC-Seal header fields must be non-failing. For instance values > 1, the value must be "pass". For instance value = 1, the value must be "none".

    * If any of these conditions are not met, the Chain Validation Status is "fail" and the algorithm stops here.

4. Validate the AMS with the greatest instance value (most recent). If validation fails, then the Chain Validation Status is "fail" and the algorithm stops here.

5. _OPTIONAL:_ Determine the "oldest-pass" value from the ARC Set by validating each prior AMS beginning with the N-1 and proceeding in decreasing order to the AMS with the instance value of 1:

6. If an AMS fails to validate (for instance value "M"), then set the oldest-pass value to the lowest AMS instance value which passed (M+1) and go to the next step (there is no need to check any other (older) AMS headers). This does not affect the validity of the Authenticated Received Chain.

7. If all AMS headers verify, set the oldest-pass value to zero (0).

8. Validate each AS beginning with the greatest instance value and proceeding in decreasing order to the AS with the instance value of 1. If any AS fails to validate, the Chain Validation Status is "fail" and the algorithm stops here.

9. If the algorithm reaches this step, then the Chain Validation Status is "pass", and the algorithm is complete.

The end result of this Validation algorithm is added into the Authentication-Results header for the ADMD.

As with a failing DKIM signature ([RFC6376] section 6.3), a message with a failing Authenticated Received Chain MUST be treated the same as a message with no Authenticated Received Chain.

INFORMATIONAL: Recipients of an invalid or failing Authenticated Received Chain can use that information as part of a wider handling context. ARC adoption cannot be assumed by intermediaries; many intermediaries will continue to modify messages without adding ARC Seals.

5.2.1. All Failures Are Permanent

Authenticated Received Chains represent the traversal of messages through one or more intermediaries. All errors, including DNS failures, become unrecoverable and are considered permanent.

Any error Validating an Authenticated Received Chain results in a failed Chain Validation Status. For further discussion of this topic and the design restriction which prevents chain continuation or re-establishment, see [ARC-USAGE].

5.2.2. Responding to ARC Validation Failures During the SMTP Transaction

If an ARC Validator determines that the incoming message fails authentication checks (potentially including ARC validation), the Validator MAY signal the breakage through the extended SMTP response code 5.7.7 [RFC3463] "message integrity failure" [ENHANCED-STATUS] and corresponding SMTP response code.

5.3. Result of Validation

An Authenticated Received Chain with a Chain Validation Status of "pass" allows Internet Mail Handlers to ascertain:

- all ARC-participating ADMDs that claim responsibility for handling (and possibly modifying) the message in transit;
- the authentication state of the message as perceived by each ADMD (from Authentication-Results header fields).

Given this information, handlers can inform local policy decisions regarding disposition of messages that experience authentication failure due to intermediate processing.

6. Communication of Validation Results

Chain Validation Status (described in Section 4.4) is communicated via Authentication-Results (and AAR) headers using the auth method "arc". This auth method is described in Section 10.1.

If necessary data is available, the ptypes and properties defined in Section 10.2 SHOULD be recorded in an Authentication-Results header field:

- smtp.client-ip - The connecting client IP address from which the message is received.

- header.oldest-pass - The instance number of the oldest AMS that still validates, or 0 if all pass.

Upon Sealing of a message, this Authentication-Results information along with all other Authentications-Results added by the ADMD will be recorded into the AAR as defined in section Section 4.1.1.

In General Concept terms, the information recorded in the ARC-Authentication-Results header field is the Evidence that gets attached to a message.

7. Use Cases

This section explores several messaging handling use cases that are addressed by ARC.

7.1. Communicate Authentication Results Across Trust Boundaries

When an intermediary ADMD adds an ARC Set to a message's Authenticated Received Chain (or creates the initial ARC Set), the ADMD communicates authentication state to the next ADMD in the message handling path.

If ARC-enabled ADMDs are trusted, Authenticated Received Chains can be used to bridge administrative boundaries.

7.1.1. Message Scanning Services

Message services are available to perform anti-spam, anti-malware, and anti-phishing scanning. Such services typically remove malicious content, replace HTTP links in messages with sanitized links, and/or attach footers to messages advertising the abilities of the message scanning service. These modifications almost always break signature-based authentication (such as DKIM).

Scanning services typically require clients to point MX records of an Internet domain to the scanning service. Messages destined for the Internet domain are initially delivered to the scanning service. Once scanning is performed, messages are then routed to the client's own mail handling infrastructure. Re-routing messages in this way almost always breaks path-based authentication (such as SPF).

Message scanning services can attach Authenticated Received Chains to messages to communicate authentication results into client ADMDs. Clients can then benefit from the message scanning service while processing messages as if the client's infrastructure were the original destination of the Internet domain's MX record.

7.1.2. Multi-tier MTA Processing

Large message processing infrastructure is often divided into several processing tiers that can break authentication information between tiers. For example, a large site may maintain a cluster of MTAs dedicated to connection handling and enforcement of IP-based reputation filtering. A secondary cluster of MTAs may be dedicated and optimized for content-based processing of messages.

Authenticated Received Chains can be used to communicated authentication state between processing tiers.

7.1.3. Mailing Lists

Mailing lists resend posted messages to subscribers. A full description of authentication-related mailing list issues can be found in [RFC7960] Section 3.2.3.

Mailing list services can implement ARC to convey the original authentication state of posted messages sent to the list's subscriber base. The ADMDs of the mailing list subscribers can then use the Authenticated Received Chain to determine the authentication state of the original message before mailing list handling.

7.2. Inform Message Disposition Decisions

ARC functionality allows Internet Mail Handlers to reliably identify intermediary ADMDs and for ADMDs to expose authentication state that can survive additional intermediary handling.

Intermediaries often break authentication through content modification, interfere with path-based authentication (such as SPF), and strip authentication results (if an MTA removes Authentication-Results headers).

Authenticated Received Chains allow ARC Validators to:

1. identify ARC-enabled ADMDs that break authentication while processing messages;

2. gain extended visibility into the authentication-preserving abilities of ADMDs that relay messages into ARC-enabled ADMDs.

Through the collection of ARC-related data, an ADMD can identify handling paths that have broken authentication.

An Authenticated Received Chain allows an Internet Mail Handler to potentially base decisions of message disposition on authentication state provided by different ADMDs.

7.2.1. DMARC Local Policy Overrides

DMARC introduces a policy model where Domain Owners can request email receivers to reject or quarantine messages that fail DMARC alignment. Interoperability issues between DMARC and indirect email flows are documented in [RFC7960].

Authenticated Received Chains allow DMARC processors to consider authentication states provided by other ADMDs. As a matter of local policy, a DMARC processor may choose to accept the authentication state provided by an Authenticated Received Chain when determining if a message is DMARC compliant.

When an Authenticated Received Chain is used to determine message disposition, the DMARC processor can communicate this local policy decision to Domain Owners as described in Section 7.2.2.

7.2.2. DMARC Reporting

DMARC-enabled receivers indicate when ARC Validation influences DMARC-related local policy decisions. DMARC reporting of ARC-influenced decisions is accomplished by adding a local_policy comment containing a list of data discovered during ARC Validation, which at a minimum includes:

- the Chain Validation Status,

- the domain and selector for each AS,

- the originating IP address from the first ARC Set:

EXAMPLE:

```
<policy_evaluated>
  <disposition>none</disposition>
  <dkim>fail</dkim>
  <spf>fail</spf>
  <reason>
   <type>local_policy</type>
   <comment>arc=pass ams[2].d=d2.example ams[2].s=s1
     as[2].d=d2.example as[2].s=s2 as[1].d=d1.example
     as[1].s=s3 client-ip[1]=10.10.10.13</comment>
  </reason>
</policy_evaluated>
```

In the above example DMARC XML reporting fragment, data relating to specific validated ARC Sets are enumerated using array syntax (eg, "ams[2]" means AMS header field with instance value of 2). d2.example is the Sealing domain for ARC Set #2 (i=2) and d1.example is the Sealing domain for ARC Set #1 (i=1).

Depending on the reporting practices of intermediate message handlers, Domain Owners may receive multiple DMARC reports for a single message. DMARC report processors should be aware of this behaviour and make the necessary accommodations.

8. Privacy Considerations

The Authenticated Received Chain provides a verifiable record of the handlers for a message. This record may include Personally Identifiable Information such as IP address and domain names. Such information is also including in existing header fields such as the "Received" header field.

9. Security Considerations

The Security Considerations of [RFC6376] and [I-D-7601bis] apply directly to this specification.

As with other domain authentication technologies (such as SPF, DKIM, and DMARC), ARC makes no claims about the semantic content of messages.

9.1. Increased Header Size

Inclusion of Authenticated Received Chains into messages may cause issues for older or constrained MTAs due to increased total header size. Large header blocks, in general, may cause failures to deliver

or other outage scenarios for such MTAs. ARC itself would not cause problems.

9.2. DNS Operations

The validation of an Authenticated Received Chain composed of N ARC Sets can require up to 2*N DNS queries (not including any DNS redirection mechanisms which can increase the total number of queries). This leads to two considerations:

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. The difficulty of forging the signature values should limit the extent of this load to domains under control of the attacker. Query traffic pattern analysis may expose information about downstream validating ADMD infrastructure.

2. DKIM only performs one DNS query per signature, while ARC can introduce many (per chain). Absent caching, slow DNS responses can cause SMTP timeouts; and backlogged delivery queues on Validating systems. This could be exploited as a DoS attack.

9.3. Message Content Suspicion

Recipients are cautioned to treat messages bearing Authenticated Received Chains with the same suspicion applied to all other messages. This includes appropriate content scanning and other checks for potentially malicious content.

Just as passing message authentication is not an indication of message safety, forwarding that information through the mechanism of ARC is also not an indication of message safety. Even if all ARC-enabled ADMDs are trusted, ADMDs may have become compromised, may miss unsafe content, or may not properly authenticate messages.

9.4. Message Sealer Suspicion

Recipients are cautioned to treat every Sealer of the ARC Chain with suspicion. Just as with a validated DKIM signature, responsibility for message handling is attributed to the signing domain, but whether or not that signer is a malicious actor is out of scope of the authentication mechanism. Since ARC aids message delivery in the event of an authentication failure, ARC Sealers should be treated with suspicion, so that a malicious actor cannot Seal spam or other fraudulent messages to aid their delivery, too.

9.5. Replay Attacks

Since ARC inherits heavily from DKIM, it has similar attack vectors. In particular, the Replay Attack described in [RFC6376] section 8.6 is potentially amplified by ARC's chained statuses. In an ARC replay attack, a malicious actor would take an intact and passing ARC Chain, and then resend it to many recipients without making any modifications that invalidate the latest AMS or AS. The impact to a receiver would be more DNS lookups and signature evaluations. This scope of this attack can be limited by caching DNS queries and following the same signing scope guidance from [RFC6376] section 5.4.1.

10. IANA Considerations

[[*Note to the RFC Editors:* "dkim - header - s" is defined both here and in [I-D-7601bis]. Please delete the overlap from whichever document goes through the publication process after the other.]]

This draft introduces three new headers fields and updates the Email Authentication Parameters registry with one new authentication method and several status codes.

10.1. Email Authentication Results Names Registry Update

This draft adds one Auth Method with three Codes to the IANA "Email Authentication Result Names" registry:

- Auth Method : arc
  Code: "none", "pass", "fail"
  Specification: [I-D.ARC] 2.2
  Status: active

10.2. Email Authentication Methods Registry Update

This draft adds several new items to the Email Authentication Methods registry, most recently defined in [I-D-7601bis]:

- Method: arc
  Definition: [I-D.ARC]
  ptype: smtp
  Property: client-ip
  Value: IP address of originating SMTP connection
  Status: active
  Version: 1

- Method: arc
  Definition: [I-D.ARC]

ptype: header
Property: oldest-pass
Value: The instance id of the oldest validating AMS, or 0 if they all pass (see Section 5.2)
Status: active
Version: 1

- Method: dkim
  Definition: [RFC6376]
  ptype: header
  Property: s
  Value: value of signature "s" tag
  Status: active
  Version: 1

10.3. 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: IETF
  Specification document(s): [I-D.ARC]
  Related information: [RFC6376]

- Header field name: ARC-Message-Signature
  Applicable protocol: mail
  Status: draft
  Author/Change controller: IETF
  Specification document(s): [I-D.ARC]
  Related information: [RFC6376]

- Header field name: ARC-Authentication-Results
  Applicable protocol: mail
  Status: standard
  Author/Change controller: IETF
  Specification document(s): [I-D.ARC]
  Related information: [I-D-7601bis]

11. Experimental Considerations

The ARC protocol is designed to address common interoperability issues introduced by intermediate message handlers. Interoperability issues are described in [RFC6377] and [RFC7960].

As the ARC protocol is implemented by intermediary handlers over time, the following should be evaluated in order to determine the success of the protocol in accomplishing the intended benefits.

11.1. Success Consideration

In an attempt to deliver legitimate messages that users desire, many receivers use heuristic-based methods to identify messages that arrive via indirect delivery paths.

ARC will be a success if the presence of Authenticated Received Chains allows for improved decision making when processing legitimate messages.

11.2. Failure Considerations

ARC should function without introducing significant new vectors for abuse (see Section 9). If unforseen vectors are enabled by ARC, then this protocol will be a failure. Note that weaknesses inherent in the mail protocols ARC is built upon (such as DKIM replay attacks and other known issues) are not new vectors which can be attributed to this specification.

11.3. Open Questions

The following open questions are academic and have no clear answer at the time of the development of the protocol. However, additional deployment should be able to gather the necessary data to answer some or all of them.

11.3.1. Value of the ARC-Seal (AS) Header

Data should be collected to show if the ARC-Seal (AS) provides value beyond the ARC Message Signature (AMS) for either making delivery decisions or catching malicious actors trying to craft or replay malicious chains.

11.3.2. DNS Overhead

Longer Authenticated Received Chains will require more queries to retrieve the keys for validating the chain. While this is not believed to be a security issue (see Section 9.2), it is unclear how much overhead will truly be added. This is similar to some of the initial processing and query load concerns which were debated at the time of the DKIM specification development.

Data should be collected to better understand usable length and distribution of lengths found in valid Authenticated Received Chains

along with the the DNS impact of processing Authenticated Received Chains.

An effective operational maximum will have to be developed through deployment experience in the field.

11.3.3. What Trace Information is Valuable

There are several edge cases where the information in the AAR can make the difference between message delivery or rejection. For example, if there is a well known mailing list that seals with ARC but doesn't do its own initial DMARC enforcement, an Internet Mail Handler with this knowledge could make a delivery decision based upon the authentication information it sees in the corresponding AAR header.

Certain trace information in the AAR is useful/necessary in the construction of DMARC reports.

Certain receivers believe the entire set of trace information would be valuable to feed into machine learning systems to identify fraud and/or provide other signals related to message delivery.

It is unclear what trace information will be valuable for all receivers, regardless of size.

Data should be collected on what trace information receivers are using that provides useful signals that affect deliverability, and what portions of the trace data are left untouched or provide no useful information.

Since many such systems are intentionally proprietary or confidential to prevent gaming by abusers, it may not be viable to reliably answer this particular question. The evolving nature of attacks can also shift the landscape of "useful" information over time.

12. Implementation Status

[[Note to the RFC Editor: Please remove this section before publication along with the reference to [RFC7942].]]

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 [RFC7942]. 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.

This information is known to be correct as of the eighth interoperability test event which was held on 2018-03-17 at IETF101.

For a few of the implementations, later status information was available as of June 2018.

12.1. GMail test reflector and incoming validation

Organization: Google
Description: Internal production implementation with both debug analysis and validating + sealing pass-through function
Status of Operation: Production - Incoming Validation
Coverage: Full spec implemented as of [ARC-DRAFT-14]
Licensing: Proprietary - Internal only
Implementation Notes:

- Full functionality was demonstrated during the interop testing on 2018-03-17.

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

12.2. 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: Beta
Coverage: ARC Chain validity status checking is operational, but only applied to email addresses enrolled in the test program. This system conforms to [ARC-DRAFT-05]
Licensing: Proprietary - Internal only
Implementation Notes:

- 2017-07-15: Full functionality verified during the interop testing.

- 2018-06: Partially retired but still accessible by special request due to the in process evolution of the AOL mail infrastructure to the integrated OATH environment. The implementation was based on the Apache James DKIM code base and may be contributed back to that project in the future.

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

12.3. dkimpy

Organization: dkimpy developers/Scott Kitterman
Description: Python DKIM package
Status of Operation: Production
Coverage:

- 2017-07-15: The internal test suite is incomplete, but the command line developmental version of validator was demonstrated to interoperate with the Google and AOL implementations during the interop on 2017-07-15 and the released version passes the tests in [ARC-TEST] arc_test_suite [3] with both python and python3.

Licensing: Open/Other (same as dkimpy package = BCD version 2)
Contact Info: https://launchpad.net/dkimpy

12.4. OpenARC

Organization: TDP/Murray Kucherawy
Description: Implemention of milter functionality related to the OpenDKIM and OpenDMARC packages
Status of Operation: Beta
Coverage: Built to support [ARC-DRAFT-14]
Licensing: Open/Other (same as OpenDKIM and OpenDMARC packages)
Implementation Notes:

- The build is FreeBSD oriented but some packages have been built for easier deployment on RedHat-based Linux platforms.

- Some issues still exist when deploying in a chained milter arrangement (such as OpenSPF -> OpenDKIM -> OpenDMARC -> OpenARC) with coordination between the stages. When deployed in a "sandwich" configuration around an MLM, there is no effective mechanism to convey trust from the ingress (validator) to egress (signer) instances. (_NOTE_: this is expected to resolved with a new release of OpenDMARC expected in mid-2018.)

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

12.5. Mailman 3.x patch

Organization: Mailman development team
Description: Integrated ARC capabilities within the Mailman 3.2 package
Status of Operation: Patch submitted
Coverage: Based on OpenARC

Licensing: Same as mailman package - GPL
Implementation Notes:

- Appears to work properly in at least one beta deployment, but waiting on acceptance of the pull request into the mainline of mailman development

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

12.6. 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-05]
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.

- 2017-07-15: Testing shows that results match the other implementations listed in this section.

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

12.7. Rspamd

Organization: Rspamd community
Description: ARC signing and verification module
Status of Operation: Production, though deployment usage is unknown
Coverage: Built to support [ARC-DRAFT-14]
Licensing: Open source
Implementation Notes:

- 2017-06-12: Released with version 1.6.0

- 2017-07-15: Testing during the interop showed that the validation functionality interoperated with the Google, AOL, dkimpy and MailerQ implementations

Contact Info: https://rspamd.com/doc/modules/arc.html and https://github.com/vstakhov/rspamd

12.8. PERL MAIL::DKIM module

Organization: FastMail
Description: Email domain authentication (sign and/or verify) module, previously included SPF / DKIM / DMARC, now has ARC added
Status of Operation: Production, deployment usage unknown
Coverage: Built to support [ARC-DRAFT-10]
Licensing: Open Source
Implementation Notes:

- 2017-12-15: v0.50 released with full test set passing for ARC

Contact Info: http://search.cpan.org/~mbradshaw/Mail-DKIM-0.50/

12.9. PERL Mail::Milter::Authentication module

Organization: FastMail
Description: Email domain authentication milter, uses MAIL::DKIM (see above)
Status of Operation: Intial validation completed during IETF99 hackathon with some follow-on work during the week
Coverage: Built to support [ARC-DRAFT-14]
Licensing: Open Source
Implementation Notes:

- 2017-07-15: Validation functionality which interoperates with Gmail, AOL, dkimpy was demonstrated; later in the week of IETF99, the signing functionality was reported to be working

- 2017-07-20: ARC functionality has not yet been pushed back to the github repo but should be showing up soon

Contact Info: https://github.com/fastmail/authentication_milter

12.10. Sympa List Manager

Organization: Sympa Dev Community
Description: Work in progress
Status of Operation: Work in progress
Coverage: unknown
Licensing: open source
Implementation Notes:

- 2018-01-05: Tracked as https://github.com/sympa-community/sympa/issues/153

Contact Info: https://github.com/sympa-community

12.11. Oracle Messaging Server

Organization: Oracle
Description:
Status of Operation: Intial development work during IETF99 hackathon. Framework code is complete, crypto functionality requires integration with libsodium
Coverage: Work in progress
Licensing: Unknown
Implementation Notes:

- 2018-03: Protocol handling components are completed, but crypto is not yet functional.

Contact Info: Chris Newman, Oracle

12.12. MessageSystems Momentum and PowerMTA platforms

Organization: MessageSystems/SparkPost
Description: OpenARC integration into the LUA-enabled Momentum processing space
Status of Operation: Beta
Coverage: Same as OpenARC
Licensing: Unknown
Implementation Notes:

- Initial deployments for validation expected in mid-2018.

Contact Info: TBD

12.13. Exim

Organization: Exim developers
Status of Operation: Operational; requires specific enabling for compile.
Coverage: Full spec implemented as of [ARC-DRAFT-13]
Licensing: GPL
Contact Info: exim-users@exim.org
Implementation notes:

- Implemented as of Exim 4.91

13. References

13.1. 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>.

[RFC3463] Vaudreuil, G., "Enhanced Mail System Status Codes", RFC 3463, DOI 10.17487/RFC3463, January 2003, <https://www.rfc-editor.org/info/rfc3463>.

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

[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>.

[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, <https://www.rfc-editor.org/info/rfc6376>.

[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>.

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

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

[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>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

13.2. Informative References

[ARC-DRAFT-05]
Andersen, K., "Authenticated Received Chain (ARC) Protocol (I-D-05)", n.d., <https://tools.ietf.org/html/draft-ietf-dmarc-arc-protocol-05>.

[ARC-DRAFT-10]
Andersen, K., "Authenticated Received Chain (ARC) Protocol (I-D-10)", n.d., <https://tools.ietf.org/html/draft-ietf-dmarc-arc-protocol-10>.

[ARC-DRAFT-13]
Andersen, K., "Authenticated Received Chain (ARC) Protocol (I-D-13)", n.d., <https://tools.ietf.org/html/draft-ietf-dmarc-arc-protocol-13>.

[ARC-DRAFT-14]
Andersen, K., "Authenticated Received Chain (ARC) Protocol (I-D-14)", n.d., <https://tools.ietf.org/html/draft-ietf-dmarc-arc-protocol-14>.

[ARC-MULTI]
Andersen, K., "Using Multiple Signing Algorithms with ARC", January 2018, <https://tools.ietf.org/html/draft-ietf-dmarc-arc-multi-01>.

[ARC-TEST]
Blank, S., "ARC Test Suite", January 2017, <https://github.com/Valimail/arc_test_suite>.

[ARC-USAGE]
Jones, S., Adams, T., Rae-Grant, J., and K. Andersen, "Recommended Usage of the ARC Headers", April 2018, <https://tools.ietf.org/html/draft-ietf-dmarc-arc-usage-05>.

[ENHANCED-STATUS]
"IANA SMTP Enhanced Status Codes", n.d., <http://www.iana.org/assignments/smtp-enhanced-status-codes/smtp-enhanced-status-codes.xhtml>.

[I-D-7601bis]
Kucherawy, M., "Message Header Field for Indicating Message Authentication Status", February 2018, <https://datatracker.ietf.org/doc/draft-ietf-dmarc-rfc7601bis/>.

[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>.

[RFC7942] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", BCP 205, RFC 7942, DOI 10.17487/RFC7942, July 2016, <https://www.rfc-editor.org/info/rfc7942>.

[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>.

13.3. URIs

[1] mailto:arc-discuss@dmarc.org

[2] mailto:arc-discuss@dmarc.org

[3] https://github.com/Valimail/arc_test_suite

[4] mailto:arc-discuss@dmarc.org

[5] https://trac.ietf.org/trac/dmarc/ticket/17

[6] mailto:dmarc@ietf.org

[7] mailto:arc-discuss@dmarc.org

[8] mailto:arc-interop@dmarc.org

[9] https://arc-spec.org

Appendix A. Appendix A - Design Requirements

[[Note: This section is re-inserted for background information from early versions of the spec.]]

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

A.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.

A.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.

Appendix B. Appendix B - Example Usage

[[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 which will be included in a future draft. Issue 17 [5]]]

[[Note: Need input from the WG as to what sort of sequence of examples would be considered useful - otherwise we'll just drop this section entirely.]]

<removed for now to reduce confusion>

Appendix C. Acknowledgements

This draft originated with 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 D. Comments and Feedback

Please address all comments, discussions, and questions to dmarc@ietf.org [6]. Earlier discussions can be found at arc-discuss@dmarc.org [7]. Interop discussions planning can be found at arc-interop@dmarc.org [8].

Some introductory material for less technical people can be found at https://arc-spec.org [9].

Authors' Addresses

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

Email: kurta@linkedin.com

Brandon Long (editor)
Google

Email: blong@google.com

Seth Blank (editor)
Valimail

Email: seth@valimail.com

Murray Kucherawy (editor)
TDP

Email: superuser@gmail.com

Tim Draegen (editor)
dmarcian

Email: tim@dmarcian.com