# E-MAIL VALIDATION SYSTEMS

- Sender Policy Framework (SPF)
  - prevents e-mail spam by detecting email spoofing through verification of sender IP addresses
- Domain Keys Identified Mail (DKIM)
  - allows to check that incoming mail from a domain is authorized by that domain's administrators and that the email (including attachments) has not been modified during transport
- Domain-based Message Authentication, Reporting, and Conformance (DMARC)
  - scalable mechanism by which a mail-originating organization can express policies for message validation, disposition, and reporting, that a mail-receiving organization can use to improve mail handling – RFC 7489 (2015)
- Vouch by Reference (VBR)
  - implements sender certification by third-party entities
  - RFC 5518
  - certification based on DomainKeys Identified Mail (DKIM)
- Authenticated Received Chain (ARC)
  - preserves email authentication results across subsequent hops that may modify the message

# SENDER POLICY FRAMEWORK

- SPFv1 (or SPF Classic) protects the sender address (in envelope) by allowing the owner of a domain to specify a mail sending policy, namely which mail servers are authorized to send mail from the domain, using special DNS records (SPF, type 99)
  - http://www.openspf.org/Project_Overview
  - RFC 7208 (2014)
- Receivers verifying the SPF records may reject messages from unauthorized sources before receiving the body of the message
- If server accepts the sender, and also accepts recipients and body of message, it should insert a Return-Path field in the message header in order to save the sender address
  - While the address in the Return-Path often matches other originator addresses in the mail header such as From or Sender, this is not necessarily the case, and SPF does not prevent forgery of these other addresses

# SPF EXAMPLE

Bob owns domain example.net. He also sometimes sends mail through his GMail account and contacted GMail's support to identify the correct SPF record for GMail.

Since he often receives bounces about messages he didn't send, he decides to publish an SPF record in order to reduce the abuse of his domain in e-mail envelopes:

```
example.net  TXT  "v=spf1 mx a:pluto.example.net
include:aspmx.googlemail.com -all"
```

# SPF EXAMPLE CONT'D

example.net  TXT
"v=spf1 mx a:pluto.example.net include:aspmx.googlemail.com -all"

| *SPF record items* | *explanation* |
|---|---|
| v=spf1 | SPF version 1 |
| mx | the incoming mail servers (MXes) of the domain are authorized to also send mail for example.net |
| a:pluto.example.net | the machine pluto.example.net is authorized, too |
| include:aspmx.googlemail.com | everything considered legitimate by gmail.com is legitimate for example.net, too |
| -all | all other machines are not authorized |

# GMAIL EXAMPLE

- mail servers (e.g. Gmail) that are contacted by clients may simply check client's IP against sender's domain name: on mismatch, message is rejected

```
<XXXX.YYYY@gmail.com>: host gmail-smtp-in.l.google.com[173.194.78.26]
    said: 550-5.7.1 [aa.bb.cc.dd] The IP you're using to send mail is not
    authorized to 550-5.7.1 send email directly to our servers. Please use the
    SMTP relay at your 550-5.7.1 service provider instead. Learn more at 550
    5.7.1 http://support.google.com/mail/bin/answer.py?answer=10336
    fl4si3665795wib.12 - gsmtp (in reply to end of DATA command)
```

# SPF TOOLS

- http://www.kitterman.com/spf/validate.html
- https://www.fraudmarc.com/spf-record-check/
- https://vamsoft.com/support/tools/spf-policy-tester
- https://www.dmarcanalyzer.com/spf/spf-validator/

analysis can be manually carried out by `nslookup`, setting query type to TXT (`set type=TXT`)

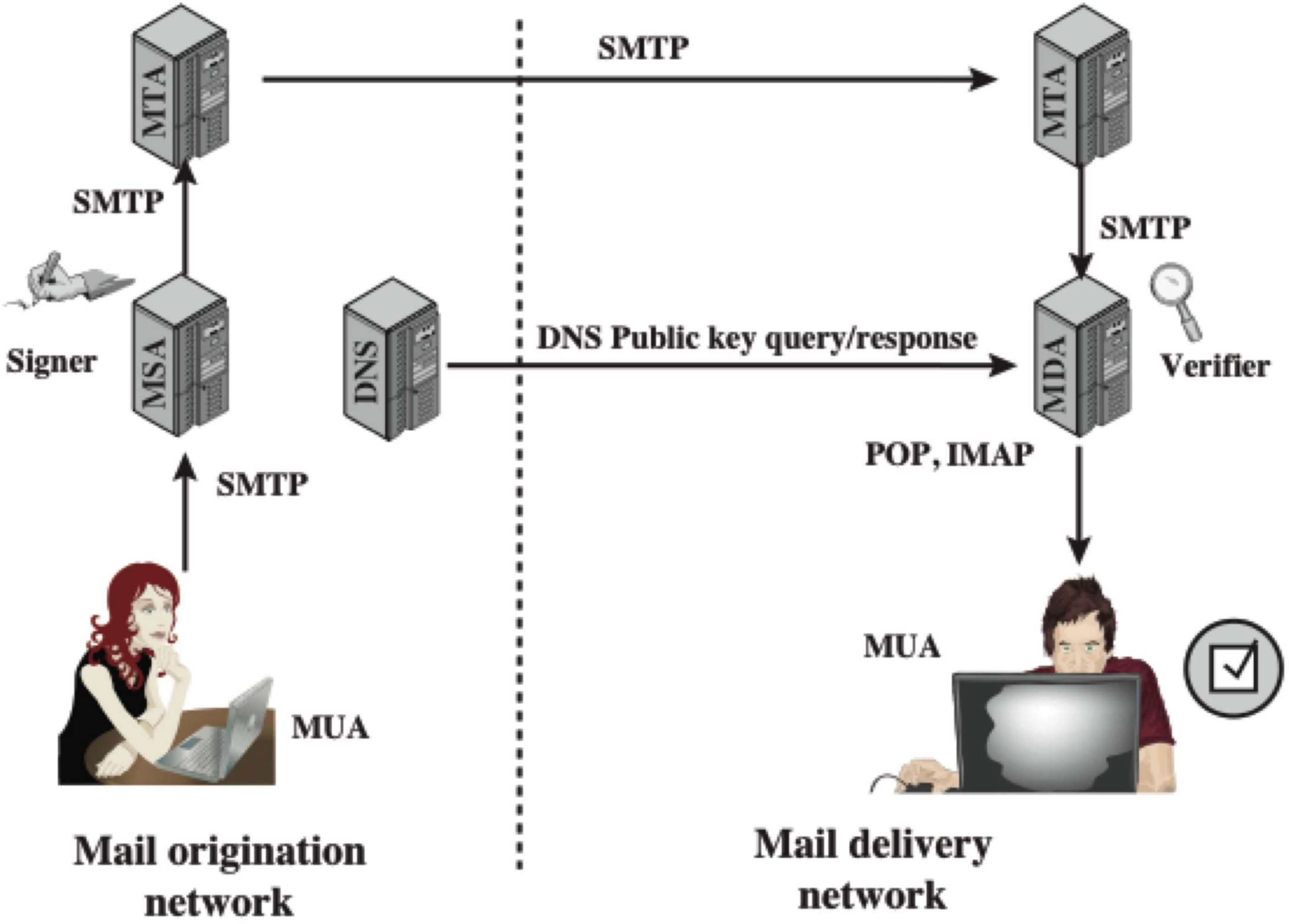
ex: `email.com` → `"v=spf1 redirect=mail.com"`

```
mail.com → "v=spf1 ip4:213.165.64.0/23 ip4:74.208.5.64/26
ip4:74.208.122.0/26 ip4:212.227.126.128/25
ip4:212.227.15.0/24 ip4:212.227.17.0/27
ip4:74.208.4.192/26 ip4:82.165.159.0/24
ip4:217.72.207.0/27 -all"
```

# DKIM

- DomainKeys Identified Mail (DKIM) is a specification for cryptographically signing e-mail messages, permitting a signing domain to claim responsibility for a message
- Message recipients (or agents acting in their behalf) can verify the signature by querying the signer's domain directly to retrieve the appropriate public key and thereby can confirm that the message was attested to by a party in possession of the private key for the signing domain
- DKIM is a proposed Internet Standard (RFC 6376 (2011): DomainKeys Identified Mail (DKIM) Signatures)
- DKIM has been widely adopted by a range of e-mail providers, including corporations, government agencies, gmail, yahoo, and many Internet service providers (ISPs)

# POSSIBLE DKIM DEPLOYMENT



# CANONICALIZATION

- e-mail servers and relay systems may modify email in transit, potentially invalidating a signature
- headers are subjected to a canonicalization algorithm
  - relaxed (tolerating) or simple (strict)
- bodies are also subjected to a canonicalization algorithm
  - choices for header/body are independent
- see RFC 4871 for details

# SELECTORS

- To support multiple concurrent public keys per signing domain, key namespace is subdivided using **selectors**
  - for example selectors might indicate the names of office locations, the signing date, or even the individual user
- Selectors are useful to implement some important use cases
  - domains that want to delegate signing capability for a specific address for a given duration to a partner, such as an advertising provider or other outsourced function
  - domains that want to allow frequent travelers to send messages locally without the need to connect with a particular MSA.
  - "affinity" domains (e.g., college alumni associations) that provide forwarding of incoming mail, but that do not operate a MSA for outgoing mail

# DKIM EXAMPLE

a = Hash/signing algorithm

q = Algorithm for getting public key

d = Signing domain

i = Signing identity

s = Selector

c = Canonicalization algorithm

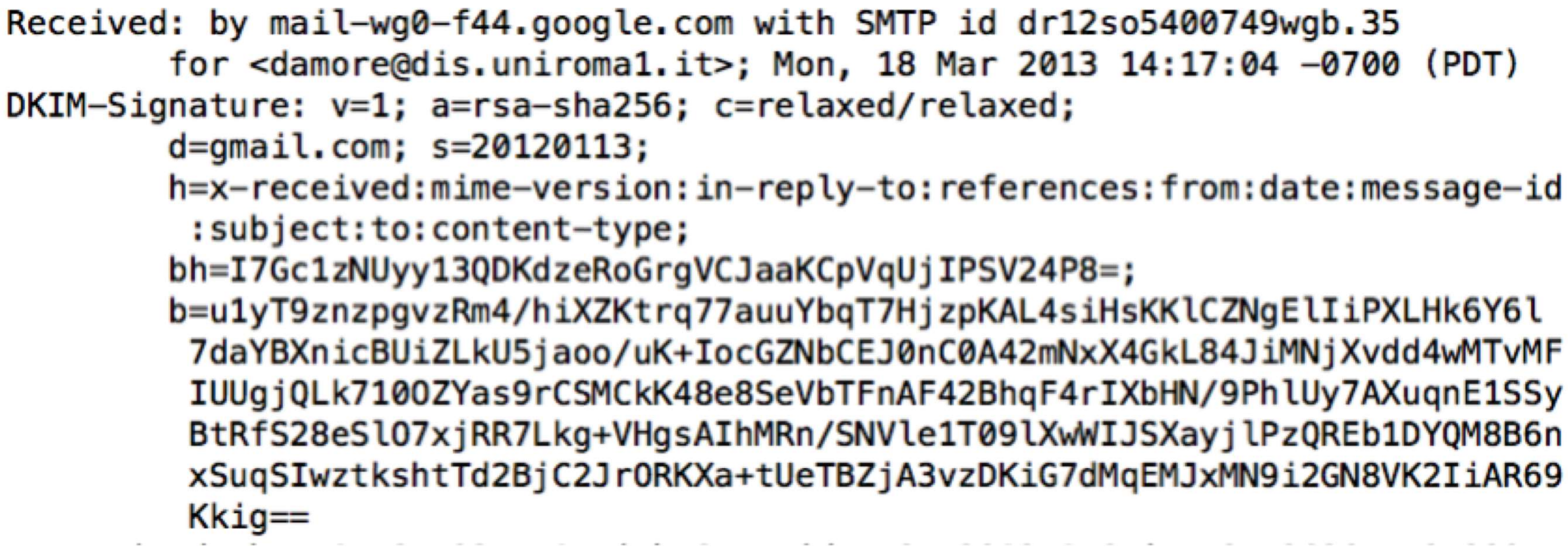
t = Signing time (seconds since 1/1/1970)

x = Expiration time

h = List of headers included in signature; dkim-signature is implied

b = The signature itself

bh = The hash of the canonicalized body part of the message



```
Received: by mail-wg0-f44.google.com with SMTP id dr12so5400749wgb.35
        for <damore@dis.uniroma1.it>; Mon, 18 Mar 2013 14:17:04 -0700 (PDT)
DKIM-Signature: v=1; a=rsa-sha256; c=relaxed/relaxed;
        d=gmail.com; s=20120113;
        h=x-received:mime-version:in-reply-to:references:from:date:message-id
         :subject:to:content-type;
        bh=I7Gc1zNUyy13QDKdzeRoGrgVCJaaKCpVqUjIPSV24P8=;
        b=u1yT9znzpgvzRm4/hiXZKtrq77auuYbqT7HjzpKAL4siHsKKlCZNgElIiPXLHk6Y6l
         7daYBXnicBUiZLkU5jaoo/uK+IocGZNbCEJ0nC0A42mNxX4GkL84JiMNjXvdd4wMTvMF
         IUUgjQLk710OZYas9rCSMCkK48e8SeVbTFnAF42BhqF4rIXbHN/9PhlUy7AXuqnE1SSy
         BtRfS28eSl07xjRR7Lkg+VHgsAIhMRn/SNVle1T09lXwWIJSXayjlPzQREb1DYQM8B6n
         xSuqSIwztkshtTd2BjC2JrORKXa+tUeTBZjA3vzDKiG7dMqEMJxMN9i2GN8VK2IiAR69
         Kkig==
```

# VOUCH BY REFERENCE

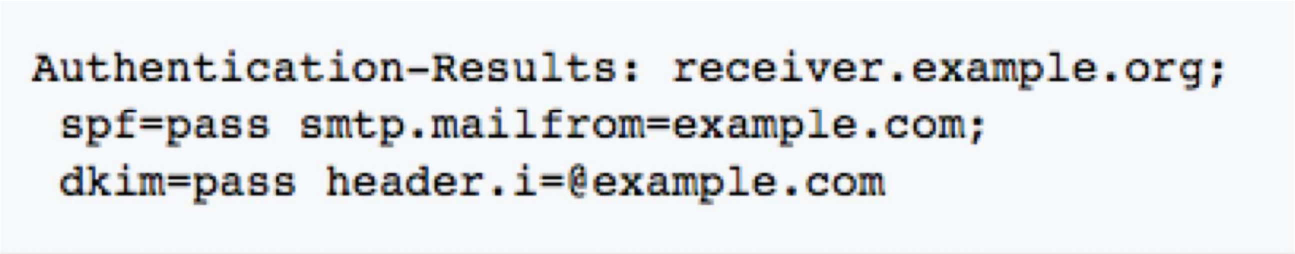
- VBR is a protocol used in Internet mail systems for implementing **sender certification by third-party entities**
  - Independent certification providers vouch for the reputation of senders by verifying the domain name that is associated with transmitted electronic mail
- **Email sender**. A user of a VBR email certification service signs its messages using DomainKeys Identified Mail (DKIM) and includes a VBR-Info field in the signed header
  - The sender may also use the Sender Policy Framework to authenticate its domain name
  - The VBR-Info: header field contains the domain name that is being certified, the type of content in the message, and a list of one or more vouching services, that is the domain names of the services that vouch for the sender for that kind of content:
    VBR-Info: md=domain.name.example; mc=type; mv=vouching.example:vouching2.example

# VOUCH BY REFERENCE

- **Email receiver**. An email receiver can authenticate the message's domain name using DKIM or SPF, thus finding the domains that are responsible for the message. It then obtains the name of a vouching service that it trusts, either from among the set supplied by the sender or from a locally configured set of preferred vouching services. Using the DNS, the receiver can verify whether a vouching service actually vouches for a given domain. To do so, the receiver queries a TXT resource record for the name composed: domain.name.example._vouch.vouching.example
- The returned data, if any, is a space-delimited list of all the types that the service vouches, given as lowercase ASCII. They should match the self-asserted message content. The types defined are transaction, list, and all. Auditing the message may allow to establish whether its content corresponds. The result of the authentication can be saved in a new header field, according to RFC 6212, like so: Authentication-Results: receiver.example; vbr=pass header.mv=vouching.example header.md=domain.name.example

# AUTHENTICATION RESULTS FIELD

- field added by an MTA that carries out authentication checks



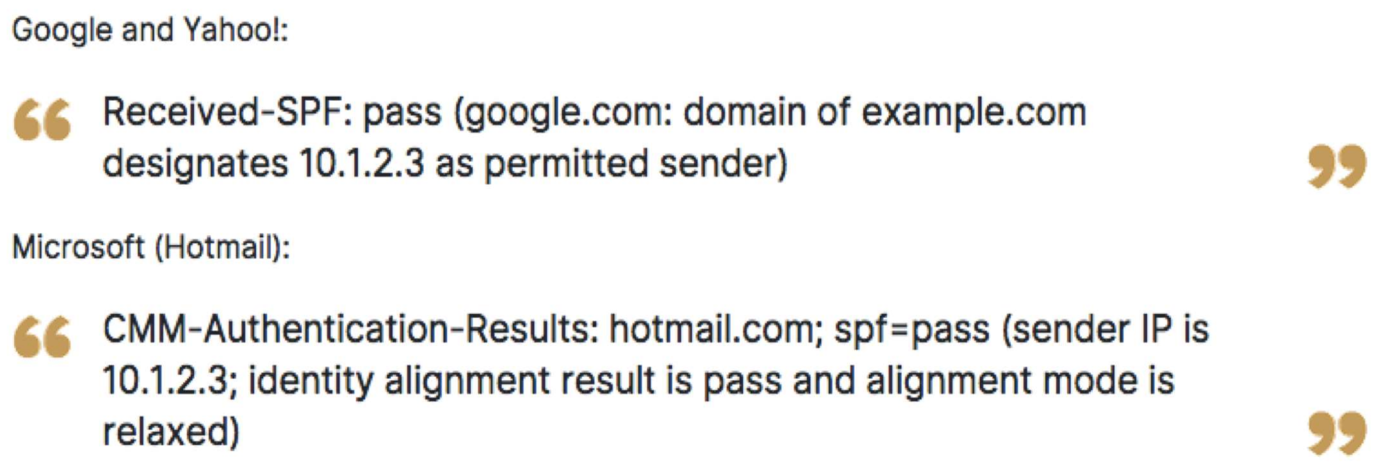
```
Authentication-Results: receiver.example.org;
 spf=pass smtp.mailfrom=example.com;
 dkim=pass header.i=@example.com
```

- receiver.example.org (authentication server) made both spf and dkim checks
- multiple Authentication-Results fields can be within the header
- RFC 7601 (2015)

from https://www.valimail.com/blog/understanding-email-authentication-headers/

# POSSIBLY MANY DIFFERENT STYLES

SPF



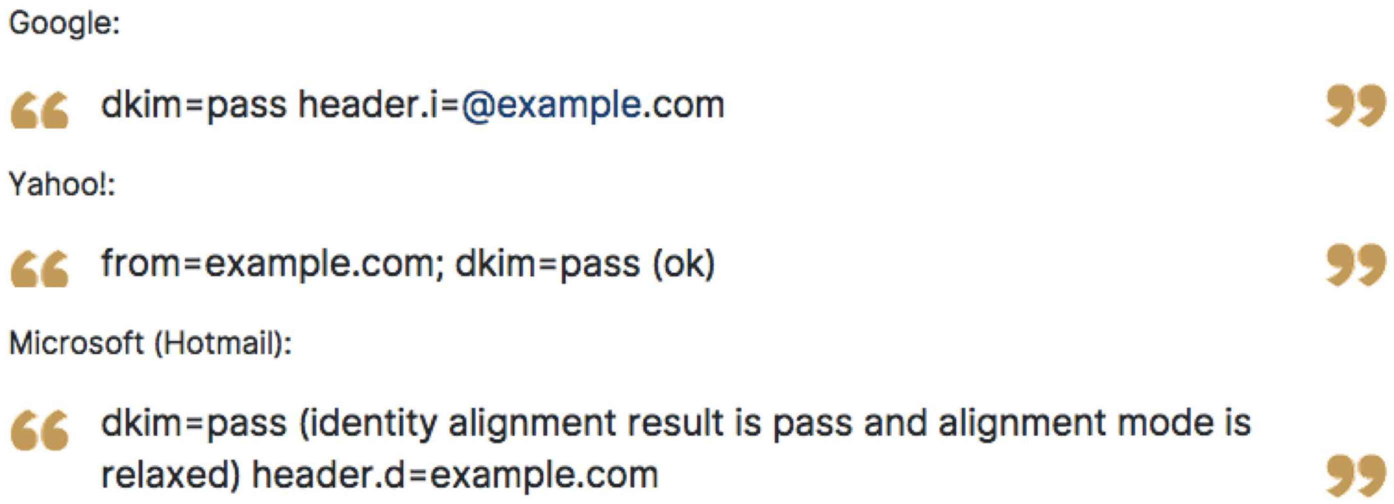
Google and Yahoo!:

> Received-SPF: pass (google.com: domain of example.com designates 10.1.2.3 as permitted sender)

Microsoft (Hotmail):

> CMM-Authentication-Results: hotmail.com; spf=pass (sender IP is 10.1.2.3; identity alignment result is pass and alignment mode is relaxed)

DKIM



Google:

> dkim=pass header.i=@example.com

Yahoo!:

> from=example.com; dkim=pass (ok)

Microsoft (Hotmail):

> dkim=pass (identity alignment result is pass and alignment mode is relaxed) header.d=example.com

# ARC

- Authenticated Received Chain (ARC) Protocol
  - draft-ietf-dmarc-arc-protocol-13 (March 2018) https://datatracker.ietf.org/doc/draft-ietf-dmarc-arc-protocol/
- Main idea:
  - handlers of an email message make authentication of message when they process it
  - they create an attached, authenticated record of the status at each step along the handling path
  - final recipient will use such records in making choices about the disposition of the message
- Changes in the message that might break existing authentication mechanisms can be identified through the ARC set of header fields

# ARC OVERVIEW

- ARC provides a **chain of custody** for a message, allowing each entity that handles the message to see
  - what entities handled it before
  - what the authentication status of the message was at each step in the handling
- The handling entity can put its own entry into the chain of custody and then relay the message to the next handler
- On final delivery, the decision whether to accept/reject/discard/quarantine it, can be based on the chain of custody and on local policies
  - can also take into account the advertised policies of the sending domain

# ARC EXAMPLE 1/2

- Sender from `mysender.example` posts a message to a mailing list hosted at `listmania.example`
- mailing list modifies the message by prepending the list name to the subject line, then sends it to subscribers
- one of the subscribers is `alice@mail.service.example`, which receives the message from `listmania.example`
- if original message was DKIM-signed and `mysender.example` published an SPF record, the handling by the mailing list will break the DKIM signature because of the message modification, and the forwarding will cause checks to fail in the next step

# ARC EXAMPLE 2/2

- `listmania.example` can add ARC headers to the message to add itself to the document's chain of custody
- when `mail.service.example` sees the message, it can see that SPF and DKIM validation fail, but it can also see that both of these succeeded when they were checked by `listmania.example`, and can verify `listmania`'s assertion
- `mail.service.example` can see that
  - `mysender.example` publishes a policy asking that unauthenticated messages be rejected
  - the assertion by `listmania.example` that the message was correctly authenticated when the message arrived there
- if it accepts that assertion and that modifications made were benign, it can deliver the message, rather than reject it

# ARC SUMMARY

- ARC basically works by signature(s), operating similarly to DKIM. A good signature provides the following assertions:
- at the time that the intermediary (signing) entity processed the message, the various assertions (SPF, DKIM-Signature(s) and/or ARC sets) already attached to the message by previous entities were valid
- for a validated ARC signature, the domain name in the signature takes some responsibility for handling of the message and that the covered content of message is unchanged since that signature was applied (same as DKIM)
- ARC evaluation results are bound into the ARC chain sequence

# IMPLEMENTATION: ARC FIELDS

- The ARC protocol adds an "ARC Set" of three new header fields to the message
  - ARC-Authentication-Results (referred as "AAR")
  - ARC-Message-Signature (referred as "AMS")
  - ARC-Seal (referred as "AS")
- ARC participants always add all of these header fields before relaying a message to the next hop
- they each have an "instance number" that increases with each MTA in the handling chain so that their original order can be preserved and the three related header fields can be processed as a set

# ARC-AUTHENTICATION-RESULTS

- virtually identical in syntax to an Authentication-Results field
- records the results of all message authentication checks done by the recording MTA at the time the message arrived
- additional information may be placed in this field compared to a standard Authentication-Results field to support a more complete DMARC report (on policies)

# ARC-MESSAGE-SIGNATURE

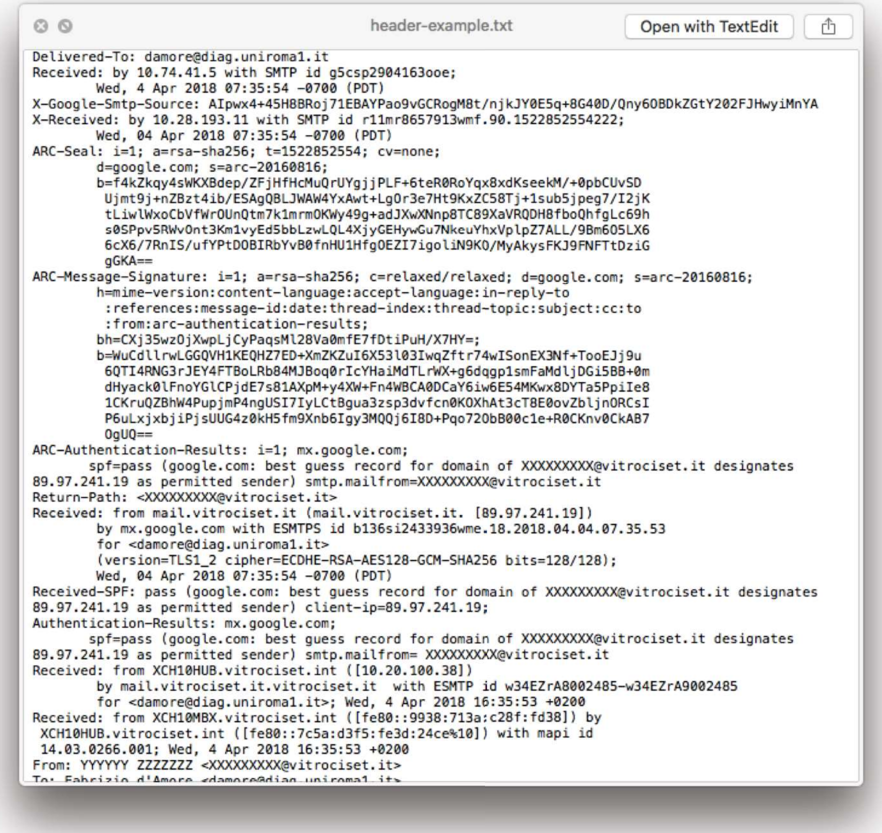
- virtually identical in syntax to DKIM-Signature
- contains the signature about the message header and body as they existed at the time of handling by the MTA adding it (including any modifications made by this agent)

# ARC-SEAL

- highly similar in structure and format to a DKIM-Signature

- applies a digital signature that protects the integrity of all three of these new fields when they are added by an MTA, plus all instances of these fields added by prior MTAs

# A REAL EXAMPLE



```
Delivered-To: damore@diag.uniroma1.it
Received: by 10.74.41.5 with SMTP id g5csp2904163ooe;
        Wed, 4 Apr 2018 07:35:54 -0700 (PDT)
X-Google-Smtp-Source: AIpwx4+45H8BRoj71EBAYPao9vGCRogM8t/njkJY0E5q+8G40D/Qny6OBDkZGtY202FJHwyiMnYA
X-Received: by 10.28.193.11 with SMTP id r11mr8657913wmf.90.1522852554222;
        Wed, 04 Apr 2018 07:35:54 -0700 (PDT)
ARC-Seal: i=1; a=rsa-sha256; t=1522852554; cv=none;
        d=google.com; s=arc-20160816;
        b=f4kZkqy4sWKXBdep/ZFjHfHcMuQrUYgjjPLF+6teR0RoYqx8xdKseekM/+0pbCUvSD
         Ujmt9j+nZBzt4ib/ESAgQBLJWAW4YxAwt+LgOr3e7Ht9KxZC58Tj+1sub5jpeg7/I2jK
         tLiwlWxoCbVfWrOUnQtm7k1mrmOKWy49g+adJXwXNnp8TC89XaVRQDH8fboQhfgLc69h
         s0SPpv5RWvOnt3Km1vyEd5bbLzwLQL4XjyGEHywGu7NkeuYhxVplpZ7ALL/9Bm6O5LX6
         6cX6/7RnIS/ufYPtDOBIRbYvB0fnHU1HfgOEZI7igoliN9KO/MyAkysFKJ9FNFTtDziG
         gGKA==
ARC-Message-Signature: i=1; a=rsa-sha256; c=relaxed/relaxed; d=google.com; s=arc-20160816;
        h=mime-version:content-language:accept-language:in-reply-to
         :references:message-id:date:thread-index:thread-topic:subject:cc:to
         :from:arc-authentication-results;
        bh=CXj35wzOjXwpLjCyPaqsMl28Va0mfE7fDtiPuH/X7HY=;
        b=WuCdllrwLGGQVH1KEQHZ7ED+XmZKZuI6X53l03IwqZftr74wISonEX3Nf+TooEJj9u
         6QTI4RNG3rJEY4FTBoLRb84MJBoq0rIcYHaiMdTLrWX+g6dqgp1smFaMdljDGi5BB+0m
         dHyack0lFnoYGlCPjdE7s81AXpM+y4XW+Fn4WBCA0DCaY6iw6E54MKwx8DYTa5PpiIe8
         1CKruQZBhW4PupjmP4ngUSI7IyLCtBgua3zsp3dvfcn0KOXhAt3cT8E0ovZbljnORCsI
         P6uLxjxbjiPjsUUG4z0kH5fm9Xnb6Igy3MQQj6I8D+Pqo72ObB00c1e+R0CKnv0CkAB7
         OgUQ==
ARC-Authentication-Results: i=1; mx.google.com;
       spf=pass (google.com: best guess record for domain of XXXXXXXXX@vitrociset.it designates
89.97.241.19 as permitted sender) smtp.mailfrom=XXXXXXXXX@vitrociset.it
Return-Path: <XXXXXXXXX@vitrociset.it>
Received: from mail.vitrociset.it (mail.vitrociset.it. [89.97.241.19])
        by mx.google.com with ESMTPS id b136si2433936wme.18.2018.04.04.07.35.53
        for <damore@diag.uniroma1.it>
        (version=TLS1_2 cipher=ECDHE-RSA-AES128-GCM-SHA256 bits=128/128);
        Wed, 04 Apr 2018 07:35:54 -0700 (PDT)
Received-SPF: pass (google.com: best guess record for domain of XXXXXXXXX@vitrociset.it designates
89.97.241.19 as permitted sender) client-ip=89.97.241.19;
Authentication-Results: mx.google.com;
       spf=pass (google.com: best guess record for domain of XXXXXXXXX@vitrociset.it designates
89.97.241.19 as permitted sender) smtp.mailfrom= XXXXXXXXX@vitrociset.it
Received: from XCH10HUB.vitrociset.int ([10.20.100.38])
        by mail.vitrociset.it.vitrociset.it  with ESMTP id w34EZrA8002485-w34EZrA9002485
        for <damore@diag.uniroma1.it>; Wed, 4 Apr 2018 16:35:53 +0200
Received: from XCH10MBX.vitrociset.int ([fe80::9938:713a:c28f:fd38]) by
 XCH10HUB.vitrociset.int ([fe80::7c5a:d3f5:fe3d:24ce%10]) with mapi id
 14.03.0266.001; Wed, 4 Apr 2018 16:35:53 +0200
From: YYYYYY ZZZZZZZ <XXXXXXXXX@vitrociset.it>
To: Fabrizio d'Amore <damore@diag.uniroma1.it>
```

# MORE INSIGHT

- SPF (Sender Policy Framework) http://www.openspf.org/
- VBR (Vouch_by_Reference) RFC 5518 (2009)
- DKIM (Domain Keys Identified Mail) http://www.dkim.org/
- DMARC (Domain-based Message Authentication, Reporting and Conformance) https://dmarc.org/
- ARC (Authenticated Received Chain) http://arc-spec.org/
- Other
  - https://mandrill.zendesk.com/hc/en-us/articles/205582267-About-SPF-and-DKIM
  - http://www.openspf.org/Related_Solutions
  - https://www.valimail.com/blog/understanding-email-authentication-headers/
  - https://dyn.com/blog/email-authentication-101/
- and…. don't forget to check the Wikipedia pages for all the items above!