DKIM Interoperability Event Report

Murray S. Kucherawy <msk@sendmail.com>
Tony Hansen <tony@att.com>
Michael Thomas <mat@cisco.com>

12/4/2007

October 24-25

- Hosted by Alt-N in Dallas, TX, USA
  - (thanks Arvel Hathcock, and everyone else there!)
- “Chaired” by Dave Crocker
- 20 companies and organizations
  - as far away as France and Japan
  - Alt-N Technologies, AOL, AT&T Inc., Bizanga Ltd., Brandenburg InternetWorking, Brandmail Solutions, ColdSpark, Constant Contact, Inc., DKIMproxy, Domain Assurance Council, Google Inc., ICONIX Inc., Internet Initiative Japan (IIJ), Ironport Systems, Message Systems, Port25 Solutions, Postfix, Sendmail, Inc., StrongMail Systems, and Yahoo! Inc.

Results

- end to end testing
  - Plus several “torture tests”
  - Some people also tested Vouch By Reference (VBR)
- Matrix of senders vs. recipients filled by end of second day
- Only minor issues remained unresolved
  - Some due to differences between what is in the text and what is in the ABNF

Results

Signature Interoperability Issues

Empty message bodies

- the “simple” body canonicalization says precisely what to do in the case of an empty message body
- “relaxed” does not
- Consensus is that the “relaxed” body canonicalization of the null body is the null input
- Majority felt it was conspicuous that “simple” was explicit while “relaxed” was not
- Errata: add clarification statement on expected values for relaxed (see next slide)

Empty Body Hash Values

Body c=	Sha	Base64 Hash Value
simple (CRLF)	sha1	uoq1oCgLlTqpdDX/iUbLy7J1Wic=
relaxed (“”)	sha1	2jmj7l5rSw0yVb/vlWAYkK/YBwk=
simple (CRLF)	sha256	frcCV1k9oG9oKj3dpUqdJg1PxRT2RSN/XKdLCPjaYaY=
relaxed (“”)	sha256	47DEQpj8HBSa+/TImW+5JCeuQeRkm5NMpJWZG3hSuFU=

No Trailing CR-LF

- What if body is non-empty, but does not end in CRLF?
  - Only possible with BDAT or non-SMTP transport mechanisms
- “simple” (§3.4.3) says to add a CRLF
- “relaxed” (§3.4.4) says nothing
- Consensus is to add a CRLF for Relaxed if
  1. it was missing and
  2. the body is not empty
- Errata: Add statement on what to do for Relaxed

Use of signature when querying "reputation"

- There is insufficient guidance about whether the domain in "i=" or the domain in "d=" should be used when generating a domain reputation query based on a DKIM signature verification
- Participants were asked to think about this and try to generate language for inclusion in later specifications to provide such guidance
- Is this in scope for the working group?
  - If not, where?

Is “a=” required or optional?

- §3.3 says that rsa-sha256 is the default if no algorithm is specified
- §3.5 says “a=” is REQUIRED
- Oops, pick one

FWS in z= tag

- §3.5 ABNF for “z=”

  sig-z-tag = %x7A [FWS] "=" [FWS] sig-z-tag-copy
  *([FWS] "|" sig-z-tag-copy)
  sig-z-tag-copy = hdr-name ":" qp-hdr-value

- Does not allow any FWS between the "|" and the following header name in sig-z-tag-copy
- By the ABNF, the informative example that immediately follows is **invalid:**

  z=From:foo@eng.example.net|To:joe@example.com|
  ----Subject:demo=20run|Date:July=205,…

FWS in z= tag

- The [FWS] is redundant there; sig-z-tag-copy ends with qp-hdr-value, which can already end with arbitrary FWS
- Perhaps the spec meant to say:
  sig-z-tag = %x7A [FWS] "=" [FWS] sig-z-tag-copy
  *("|" [FWS] sig-z-tag-copy)
  which isn't redundant and agrees with the example?
- No FWS allowed between the hdr_name and the following ":“:
  sig-z-tag-copy = hdr-name [FWS] ":" qp-hdr-value
- Errata: either fix ABNF or fix the informative example
  - Too late to fix ABNF?

When does x= take effect?

- §3.5 says the “x=” value is an “absolute date”
- A receiver’s notion of absolute time might not match the sender’s notion of absolute time
  - The document may not expire exactly when sender thinks it should
- A receiving implementation has these choices:
  - Try to decide how far apart sender’s notion of absolute time is from the receiver’s notion of absolute time, based on header information (ugh)
  - Use local knowledge of what the absolute time is
  - Add in a “fudge factor” to acknowledge possible clock drift
- Errata: add statement saying something like: “Due to clock drift, the receiver’s notion of when to consider the signature expired may not match exactly when the sender is expecting. Receiver’s MAY add a ‘fudge factor’ to allow for such possible drift.”

Invalid q=, etc. values

- q=foo/bar:dns/txt:exam/ple
- Nothing in text about unknown values
- But ABNF says unknown values are for “future extension”
- Consensus: ignore unknown values
- Errata: Add statement saying unknown values must be ignored in signature “q=” and key “h=”, “k=”, “s=”, “t=”

DNS Key Interoperability Issues

“s=” in key records

- §3.6.1 doesn't say what to do if one of the colon-separated words is a word not enumerated in the “currently defined service types”

  ```
  s=foo:email:bar
  ```

- No explicit guidance about what to do with clearly bogus values, e.g.

  ```
  s=*:email
  ```

- Consensus is to ignore any colon-separated value not recognized and only pay attention to “*” and “email” for DKIM e-mail implementations
- Errata: add such a statement

Multiple TXT records

- §3.6.2.2. states the handling of multiple TXT records is “undefined”, but it still came up in conversation several times
- Also noteworthy is that SSP has not yet added any discussion on this topic
- Are you allowed to have two TXT records with the same selector name for different sha values?
- Not a major issue at this time?
- Or should we add a definite MUST NOT?

“t=y” Semantics

- Different usage by implementations
- Some people ignore it entirely
- Others change “hard fail” to “neutral”

- Do nothing?

g=foo*bar

- Is “g=” value allowed to have a “*” in the middle?
- §3.6.1 talks about x* and *x
- ABNF indicates anywhere and multiple
- Consensus: yes

- Errata: Give additional examples like “foo*bar”, “ex*am*ple” and “*exam*ple”.

Interoperability Issues in Signature & DNS Record Interaction

“h=” (key) vs. “a=” (signature)

- One participant felt the requirement to correlate these two values was not sufficiently normative
- ??

Other Interoperability Issues

Are “forgiving” Parsers Allowed?

- dkim-signature: x=123; ;

↑

empty tag=value invalid per ABNF

- Pro forgiving:
  - Where’s the harm?
- Anti forgiving:
  - It drags us all down to the “lowest common denominator”
  - Makes “Conformance Testing” harder
- Does it lead to security problems if some parsers are liberal and others are not?
- No consensus
- Straw for errata: A receiver SHOULD use strict ABNF checks.

Other Results

Validation Test Plan?

- Validation tests are broken down into the following categories:
  - Canonicalization
  - Signature
  - Public Key
  - Tags
    - Generic
    - Signatures
    - Public Key Record
  - Signing/Verifying Algorithm
  - System
- Tim Draegen <tdraegen@ironport.com>

Q & A