JP AAWG JAPAN ANTI-ABUSE WORKING GROUP

マニアックセッション
～次世代のメールプロトコルの斜め読み

DMARCbis, DKIM2　などなど

HORNETSECURITY

平野善隆

2025/11/04

自己紹介

名前　　平野 善隆

所属　　Hornetsecurity株式会社 (旧Vade Japan)
Principal Messaging Engineer

好きな技術　　メール、DNS、Python、Go
AWS、Serverless

趣味　　長距離の自転車大会(1,200kmとか、2,000kmとか)
バンド演奏

主な活動　　M³AAWG
JPAAWG
迷惑メール対策推進協議会
Audax Randonneurs Nihonbashi

メールとの関わり

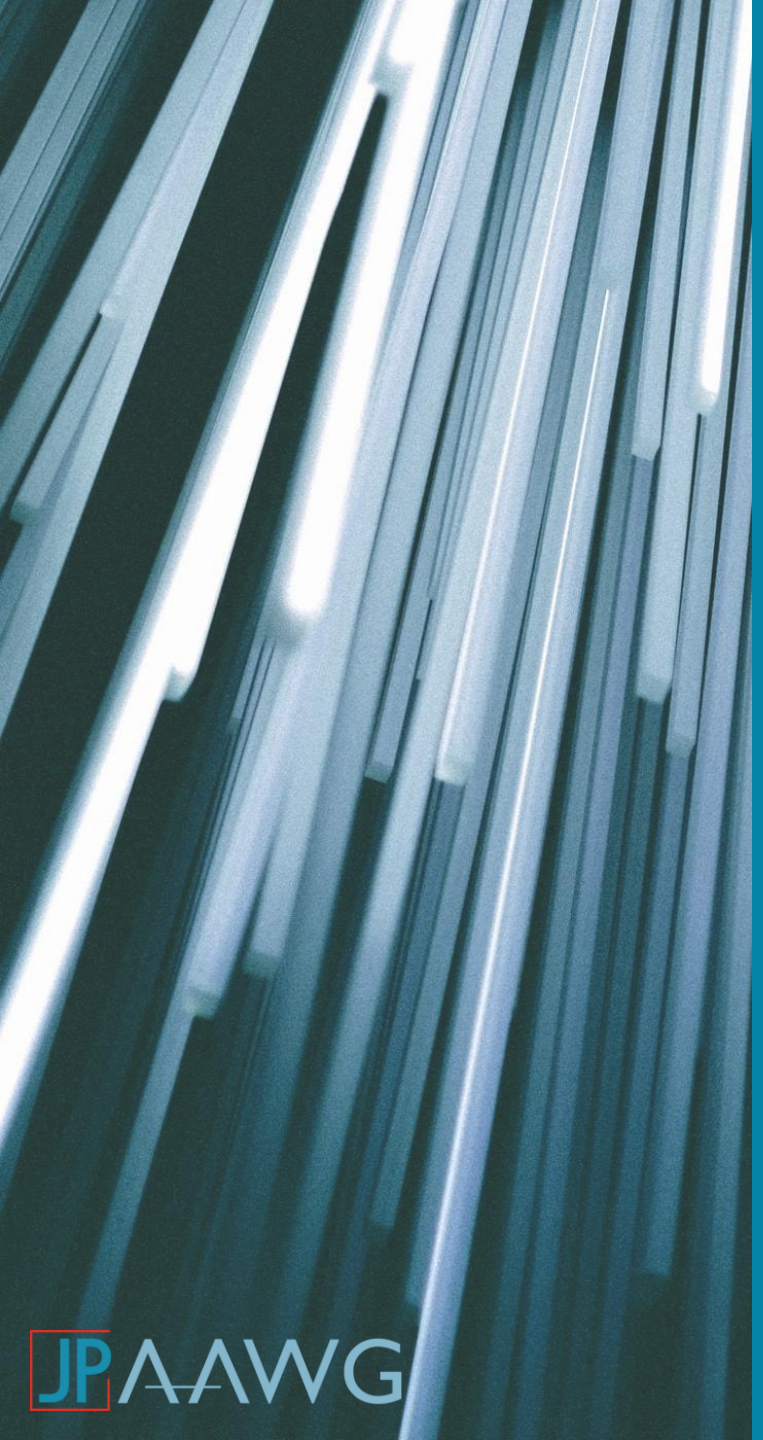
1990	パソコン通信などでメールに触れる
199x	ドメインを取得して近所のISPに個人のサーバーを置かせてもらって運用開始
2000	外人さんの多い会社に転職したのでメールの漢字にふりがなを付けたりして遊ぶ (のちのhiragana.jp)
	個人のサーバーをちゃんとしたデータセンターに移動。 imail.ne.jpというドメインを取って一攫千金を夢見るが挫折
2004	メールの会社に入社
以降	スパムフィルタ、誤送信防止製品の開発やサービスの立ち上げ。 PPAPの礎を築く。
2023	Vade Japan(現Hornetsecurity)に入社

注意事項

- あくまで斜め読みです
- 間違ってたらごめんなさい
- 深い質問をされても、たぶん困ります

お題

- DMARCbis
- DKIM2
- Secure SMTP/TLS SRV Announcement
- RFC5321bis
- RFC5322bis
- SMTP Service Extension for Client Identity
- The LIMITS SMTP Service Extension
- Deploying and Using Email in Deep Space
- BIMI on an Independent MUA



DMARCbis

次世代のDMARC
https://datatracker.ietf.org/doc/draft-ietf-dmarc-dmarcbis/41/

JPAAWG

DMARCbis

- 今のDMARCの嫌なところを解決しています
- 曖昧なところを詳しく書き直しています
- 今のDMARCのRFC、RFC7489が廃止になります
- DMARCのPublic Suffix Domainに関するRFC9091が廃止になります

- 次世代といいつつ、たぶんそろそろ出ます

新しく追加されたタグ

- 新しく追加されたタグ
  - np, psd, t

- 削除されたタグ
  - pct, ri, rf

新しいタグ (np)

- 存在しないサブドメインに対するポリシー

- 存在しないとは?
  - sub.example.jpが存在するかどうか
  - _dmarc.sub.example.jpがあるかどうかは関係ない

example.jp　v=dmarc1; p=none, sp=quarantine, np=reject

sub.example.jpが存在しない → reject

sub.example.jpは存在する

_dmarc.sub.example.jpが存在しない → quarantine

_dmarc.sub.example.jpがp=none → none

p, sp, npの関係

親ドメイン p	sp	np	Subdomain 存在する?	_dmarc	参照ポリシー
あり	なし	なし	存在しない	–	親ドメインのp
あり	なし	なし	存在する	なし	親ドメインのp
あり	なし	なし	存在する	あり	subドメインのp
あり	あり	なし	存在しない	–	親ドメインのsp
あり	あり	なし	存在する	なし	親ドメインのsp
あり	あり	なし	存在する	あり	subドメインのp
あり	なし	あり	存在しない	–	親ドメインのnp
あり	なし	あり	存在する	なし	親ドメインのp
あり	なし	あり	存在する	あり	subドメインのp
あり	あり	あり	存在しない	–	親ドメインのnp
あり	あり	あり	存在する	なし	親ドメインのsp
あり	あり	あり	存在する	あり	subドメインのp

新しいタグ (psd)

- Public Suffix Domainを表すタグ
  - y, n, u が書ける

現行のDMARCの仕組み

- a.b.c.hornetsecurity.com
  - 1. _dmarc.a.b.c.hornetsecurity.com を調べる
    - → あれば使う
  - 2. なければ
    _dmarc.b.c.hornetsecurity.com を調べる?
    _dmarc.hornetsecurity.com を調べる?
  - 3. _dmarc.hornetsecurity.comにもなければ
    _dmarc.com を調べる?

組織ドメインを調べる

調べない

組織ドメインより上は調べない

Public Suffix Domainとは

- jp, co.jp, com, net などの組織ドメインの1つ上の階層のこと
- もともとは、webのクッキーの境界のため

company1.net
company2.net

お互いクッキーが見えるとまずい

sub1.example.net
sub2.example.net

お互いクッキーが見えても問題ない

website1.cloudfront.net
website2.cloudfront.net

こちらも、まずい

cloudfront.netもPublic Suffix Domain

- Public Suffix Listというものが(いくつか?)ある
  が、管理が曖昧

新しいタグ (psd)

- 既存の曖昧なPublic Suffix Listを使わずに、
  自分で境界を宣言するためのタグ
  - y: 自分はPublic Suffix Domainです。ここより上は探索しないでね。
    例: jp, co.jp, cloudfront.net
  - n: 自分は組織ドメインまたはそのサブドメインです。
    ここより上は探索しなくていいよ。
  - u: 不明 (デフォルト)
    組織ドメインかPSDになるまで上位のドメインを探索する。
    → レコードがない場合もこれ

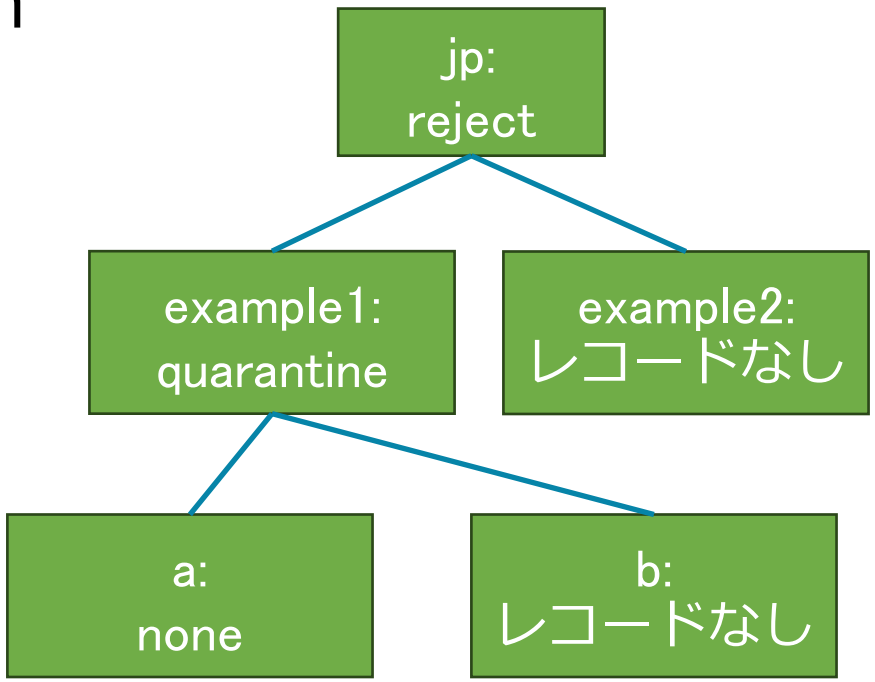
a.9.8.7.6.5.4.3.2.com

辿るのは上位8階層だけ

こんな感じ?

- _dmarc.jp　　p=reject; psd=y
- _dmarc.example1.jp　　p=quarantine; psd=n
- _dmarc.a.example1.jp　　p=none; psd=n
- _dmarc.b.example1.jp　　レコードなし
- _dmarc.example2.jp　　レコードなし

- example1.jp　→ quarantine
- a.example1.jp　→ none
- b.example1.jp　→ quarantine
- example2.jp　→ reject



現在のDMARCではPSDまで辿らないので
DMARCは「なし」になる

新しいタグ (t)

- テストモード
- 1段階低いポリシーが適用される
  - rejectの場合はquarantine
  - quarantineの場合はnone
  - noneの場合は変化なし
- 値はyとn
- デフォルトはn
- 詳しくは付録A6で議論

t=y と pct=0

- pctは0～100の任意の値を設定できたが、受信側は事実上0と100しか実装していなかった。
- 「p=none」と「p=quarantine; pct=0」に違いがある?!
- メーリングリストなどでヘッダFromを書き換える機能がある。
  → p=quarantine, rejectの場合には書き換わる
  → p=noneでは書き換わらない
- 0, 100しかないのにpctはおかしい
- tタグを新たに導入、pctは廃止

廃止されたタグ

- pct: tに置き換え
- ri: レポートの送信間隔。
  誰も実装せず無視して日次で送信。→ 廃止
- rf: 失敗レポートのフォーマット
  afrf(ARF形式)とiodefが指定できたが、ARFに固定。

レポートを外部ドメインに送るときの例のバグ修正

- 現行DMARCのRFC7489

```
example.com._report._dmarc    IN    TXT    "v=DMARC1"
```

- DMARCbis

```
example.com._report._dmarc    IN    TXT    "v=DMARC1;"
```

ここの「;」

用語集

- DMARCbisには詳しい用語集が付いている (3章)

- Monitoring Mode ← p=noneの状態
- Enforcement ← p=quarantineやrejectの状態

ドメイン所有者のアクション

- SPFをヘッダFromのドメインで公開
- DKIMをヘッダFromのドメインで署名するように設定
- DMARCレポートを受けられるように準備
- p=noneとruaタグを書いたDMARCレコードを公開
- レポートを集めて分析
- 認証に失敗している原因を修正
- ポリシーを厳しくするかどうかを決める

必ずしもp=rejectにするのがいいとは限らない

慎重になるべきポイント

- 卒業生向けメール転送サービス、部署名のエイリアス、メーリングリストなどに送る場合
- かつ
- 中継するメールサーバーがヘッダFromアドレスを書き換えない場合

- SPFに依存してはならない(MUST NOT)
- 有効なDKIM署名が必要 (MUST)

慎重になるべきポイント2

- 一般ユーザーが日常的にメールを送るようなドメインで、インターネット上のメーリングリストに送るような場合

- p=rejectにするべきではない (SHOULD NOT)
- p=rejectにしたい場合は
  - 1ヶ月間p=noneでレポートを分析
  - さらに1ヶ月間p=quarantineでレポートを分析、noneの時と比較する
  - p=rejectにする場合は、ユーザーに
    - メーリングリストに投稿させないようにするか
    - メーリングリストの参加が妨げられる可能性があることを周知すべき (SHOULD)

メール受信者のアクション

- ヘッダFromからドメインを抽出
- DMARCが宣言されているか確認
- SPF、DKIMのチェック
- アラインメントを確認
- DMARCのPASS, FAILを決定
- ポリシーを適用する
- 結果を保存する
- Aggregateレポートを送る
- 失敗レポートを送ってもいい

ポリシー強制の懸念

- DMARC PASSはヘッダFromドメインが正当なものであることを示すにすぎず、メッセージの価値判断をするものではない
- DMARC PASSでも隔離したり、Rejectしてもよい

- p=rejectでDMARCが失敗しても、メールを受け入れてもよい
- p=rejectだけを理由にRejectすべきではない (SHOULD NOT)

- p=noneの場合は既存の処理プロセスを変更してはならない (MUST NOT)

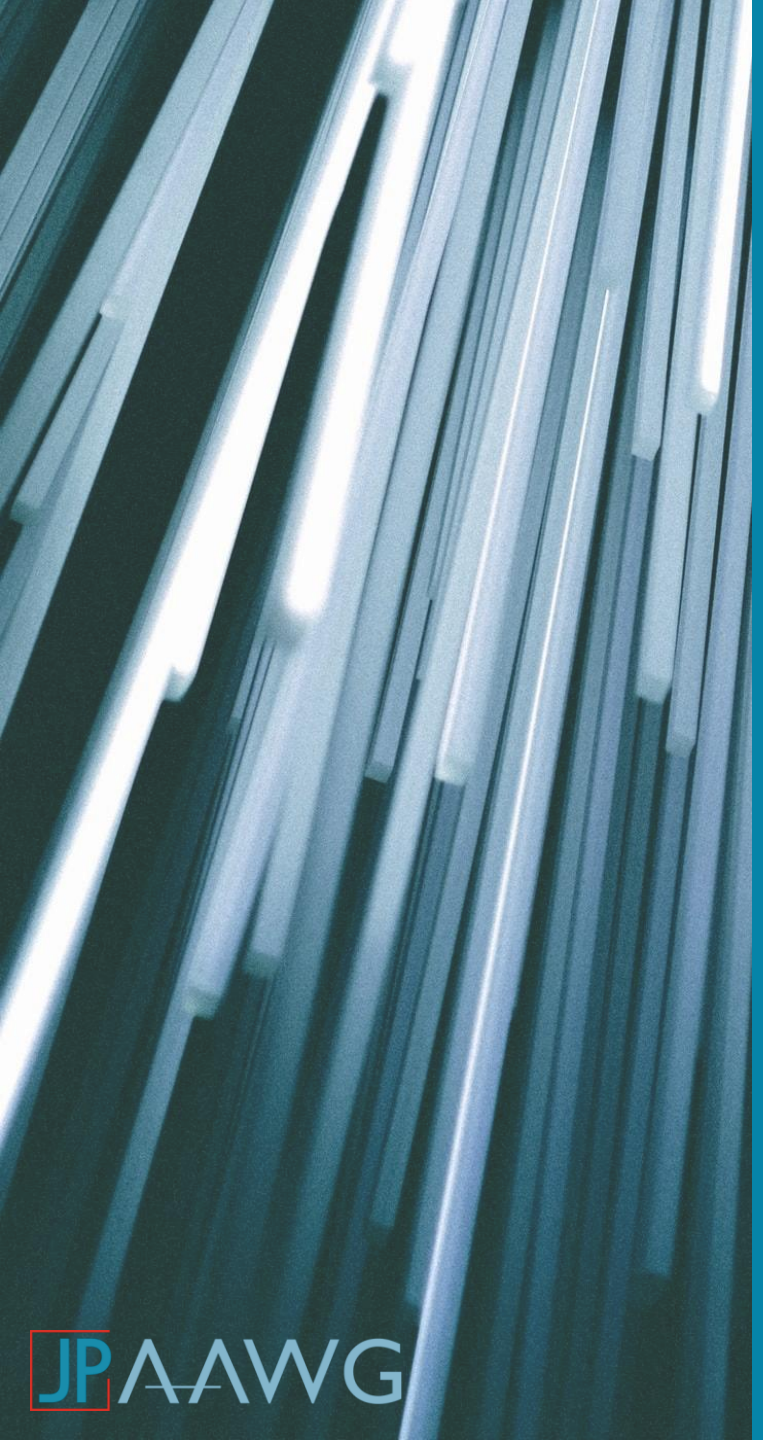
慎重になるべきポイント

- メーリングリスト管理ソフトは、メールが届かない場合、リストから購読解除する
- メーリングリスト経由のメールをp=rejectだからといって拒否すると、送信者がメーリングリストから購読解除されてしまう
- この場合、受信サーバーはp=rejectのみに基づいて拒否してはならない(MUST NOT)

ARCよりはヘッダFromの書き換え

- ここ10年でメーリングリストソフトはすでに
  bob@example.com を
  bob=example.com@user.somelist.example
  のように書き換えるような修正をおこなった
- 理想とはほど遠いがメーリングリスト業者は受け入れている

- ARCも解決策のひとつで研究は現在も進行中であるが、
  この文書の発行時点では、広く普及するには至っていない
- 将来、広く採用されるようになった場合には、
  本文書は更新される



DKIM2

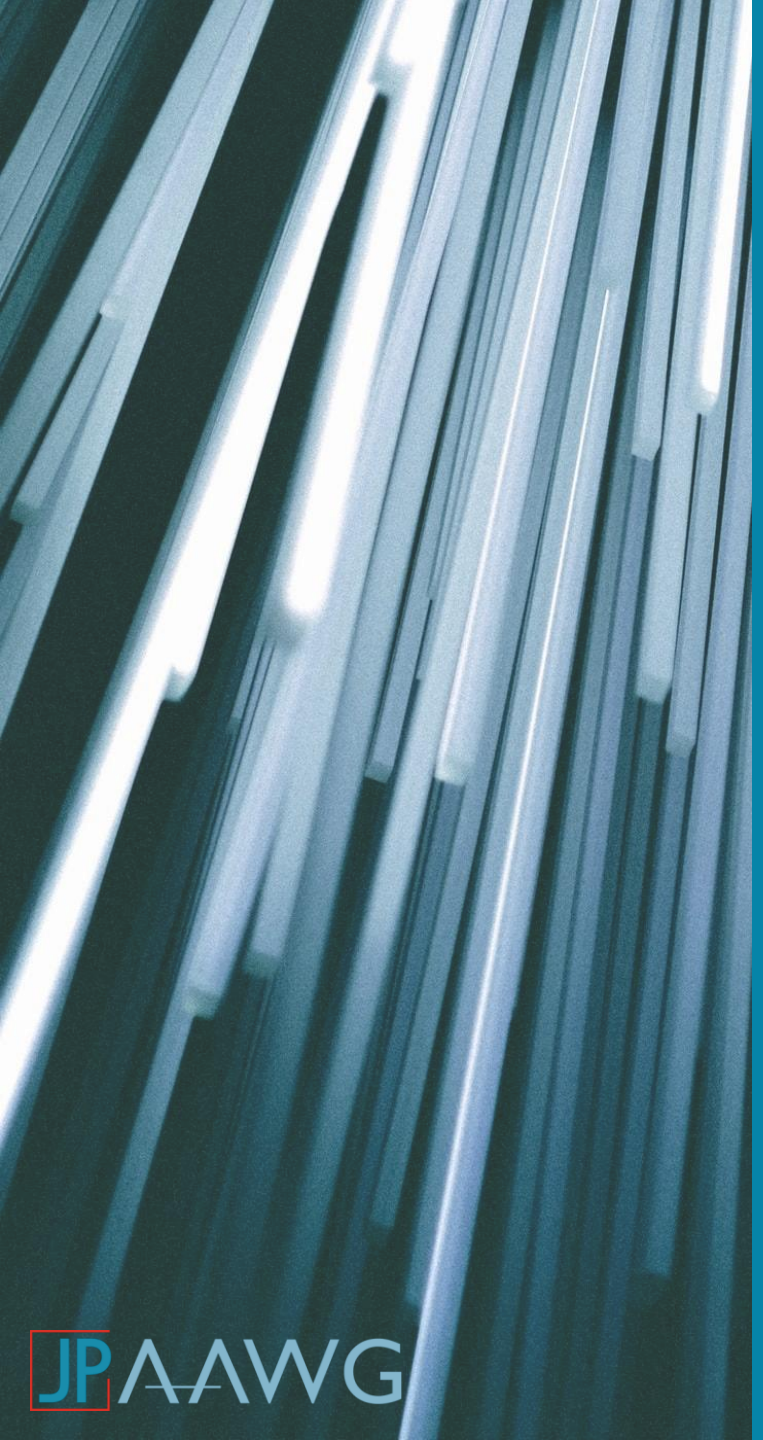
次世代のDKIM
https://datatracker.ietf.org/doc/draft-ietf-dmarc-dmarcbis/41/

JPAAWG

DKIM2のRFCドラフト

- DKIM2 - signing the source and destination of every email (draft-ietf-dkim-dkim2-motivation-01)
- DomainKeys Identified Mail Signatures v2 (DKIM2) (draft-clayton-dkim2-spec-03)
- DKIM2 Header Definitions (draft-gondwana-dkim2-header-05)

- Domain Name Specification for DKIM2 (draft-chuang-dkim2-dns-03)
- A method for describing changes made to an email (draft-gondwana-dkim2-modification-alegbra-04)
- DKIM2 Procedures for bounce processing (draft-robinson-dkim2-bounce-processing-01)
- DKIM2 Message Examples (draft-robinson-dkim2-message-examples-00)



DKIM2 - signing the source and destination of every email

draft-ietf-dkim-dkim2-motivation-02 (2025-11-03)

JP AAWG

DKIM2 - signing the source and destination of every email

- 2007に公開されたDKIMの問題点を解決する目的

- DKIM-Replay攻撃
- 署名者へのフィードバック
- 署名対象ヘッダのリストの簡素化
  - 特定のヘッダは必ず署名
  - 差分は保存
  - → Header Stuffing Attackから守る

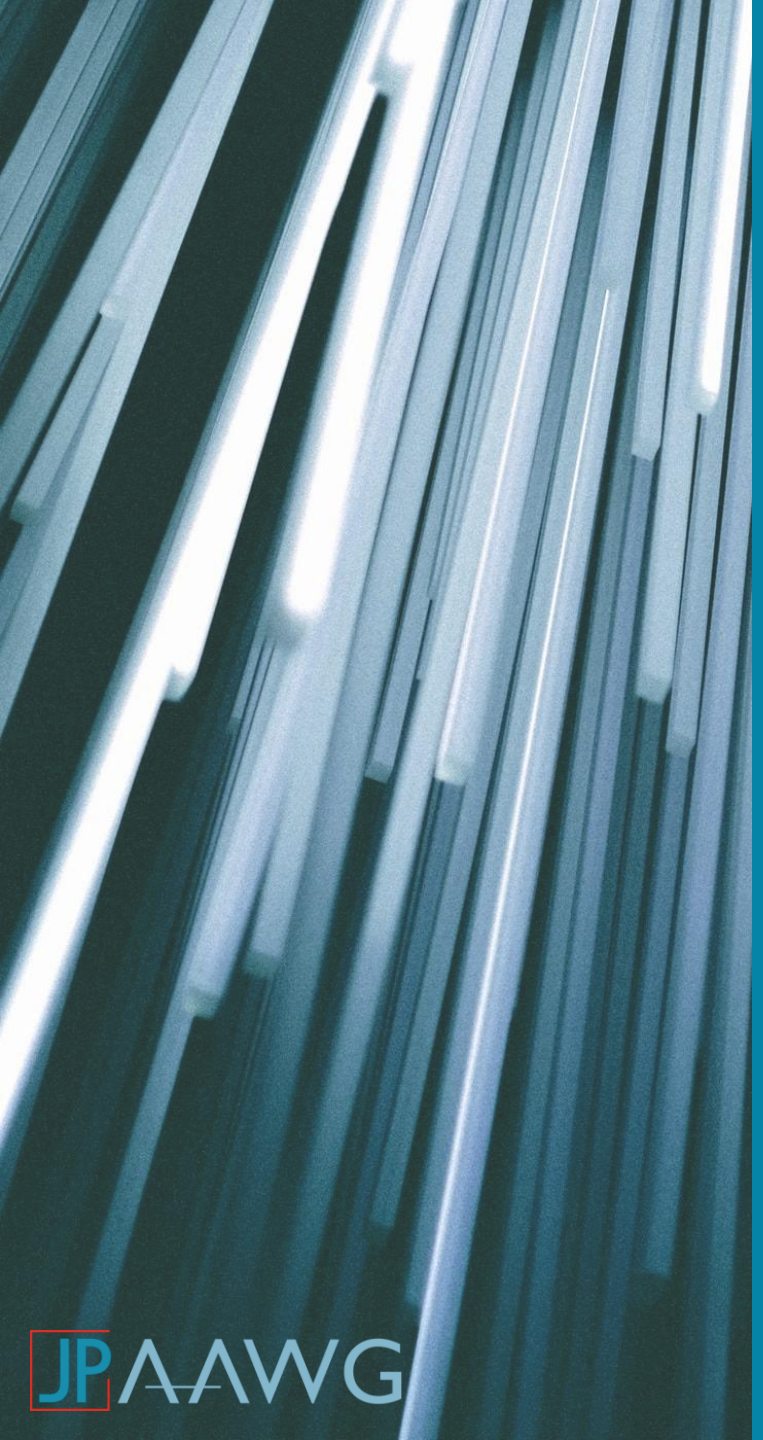
    From: nisemono@example.jp ← 悪い人が追加
    From: honmono@example.com

~~配送途中ですること~~

- ~~正当な受信者を記録する~~
  - ~~DKIM Replay Attackの防止~~
- ~~中継されるたびに署名を追加し、連鎖させる~~
- ~~バウンスは署名され、元のパスを辿って返送される~~
  - ~~Back Scatterの防止~~
- ~~変更する場合、変更内容を記録する~~
  - ~~URL書き換えなど契約に基づいて変更している場合など戻す必要がないものについては不要~~

メールシステム運用者の悩み

- 有効な DKIM署名付きメールを正当に受信しても、署名された部分の中に自分が意図された受信者であるという証拠が存在しない場合がある。
- メールのバウンス（SMTP-FROM）アドレスが、そのメッセージの転送経路に一切関与していないドメインを示す場合がある。
- メールが転送者やメーリングリストによって改変されることがあり、メッセージのどの部分が変更されたのかを知る手段が存在しない。



DomainKeys Identified Mail Signatures v2 (DKIM2)

draft-clayton-dkim2-spec-03 (2025-11-03)

JP AAWG

DomainKeys Identified Mail Signatures v2 (DKIM2)

- DKIM2のメインプロトコルを説明する資料

- 以下がDKIMとは異なる
  - 新しいDKIM2-Signatureヘッダ
  - 署名、検証手順の変更
  - 暗号アルゴリズムと鍵管理の変更
  - その他セキュリティ考慮

DKIM2-Signatureヘッダ

- i: 署名の連番
- d: 署名者のドメイン名
- s1: セレクタ
- mv: メッセージのバージョン
- t: タイムスタンプ
- mf: MAIL FROM (Envelope From)
- rt: RCPT TO (Envelope To)
- a1: 署名のアルゴリズム
- b1: 署名データ
- bh1: 本文のハッシュ
- h: 署名するヘッダ名 (option)
- f: フラグ (option)
- ~~pp: 代理署名時の元のドメイン (option)~~ ← 昨日なくなった
- n: base64の文字列。署名者が自由に使ってよく、受信時には利用されない (option)

i=タグ

- 中継毎にカウンターを増加
- 最大50

```
MAIL FROM: <a@origin.example>
RCPT TO: <b@alias.example>
DATA
DKIM2-Signature: i=1; d=origin.example; s1=key1;
    mf=a@origin.example; rt=b@alias.example; b1=PASS
From: <a@origin.example>

A Message!
```

```
MAIL FROM: <b@alias.example>
RCPT TO: <c@destination.example>
DATA
DKIM2-Signature: i=2; d=alias.example; s1=key4321;
    mf=b@alias.example; rt=c@destination.example; b1=PASS
DKIM2-Signature: i=1; d=origin.example; s1=key1;
    mf=a@origin.example; rt=b@alias.example; b1=PASS
From: <a@origin.example>

A Message!
```

h=タグ

- 転送する際に追加した署名対象のヘッダ名
- 追加するヘッダは既存の同じ名前のヘッダより前に追加する
- 同じ名前のヘッダを複数追加する場合は、回数分書く
- ReceivedやX-ヘッダなど署名に含めないものは書いてはいけない
- MailVersionが使われる場合は、そのヘッダはここには書かない

t=タグ

- 署名のタイムスタンプ (RFC3339のdate-time形式(UTC))
  - 2006-01-02T15:04:05 みたいなやつ
- 2週間以前のものは無視されるかも
  - draft-gondwana-dkim2-headerでは1週間以前のものは受信拒否

f=タグ

- すでにやったこと・受信サーバにやってほしいこと

- exploded
  - 他の人に送った
- modifiedbody
  - 本文を変更した
- modifiedheader
  - ヘッダを変更した
- donotmodify
  - 変更しないで
- donotexplode
  - 他の人に送らないで
- feedback
  - Feedbackレポートを送って

s1, a1, b1, bh1 / s2, a2, b2, bh2

- 複数の署名を付けられるようになっている
- どちらか失敗したら、全体が失敗とみなす
- たぶんこんな感じ

```
DKIM2-Signature: i=1; d=example.jp;
  s1=sel1; a1=rsa-sha256;     bh1=abc; b1=abcdefg;
  s2=sel2; a2=ed25519-sha256; bh2=abc; b2=hijklmn;
From: <a@example.jp>

A Message!
```

bh=タグ

- 本文のハッシュ値
- DKIM1ではrelaxedとsimpleを選択できた
- DKIM2ではrelaxedに固定 (選択できない)

pp=タグ

- ~~転送時等に代理で署名した時の本来のドメイン~~
- ~~origin.exampleの代わりにforwarder.exampleが署名~~

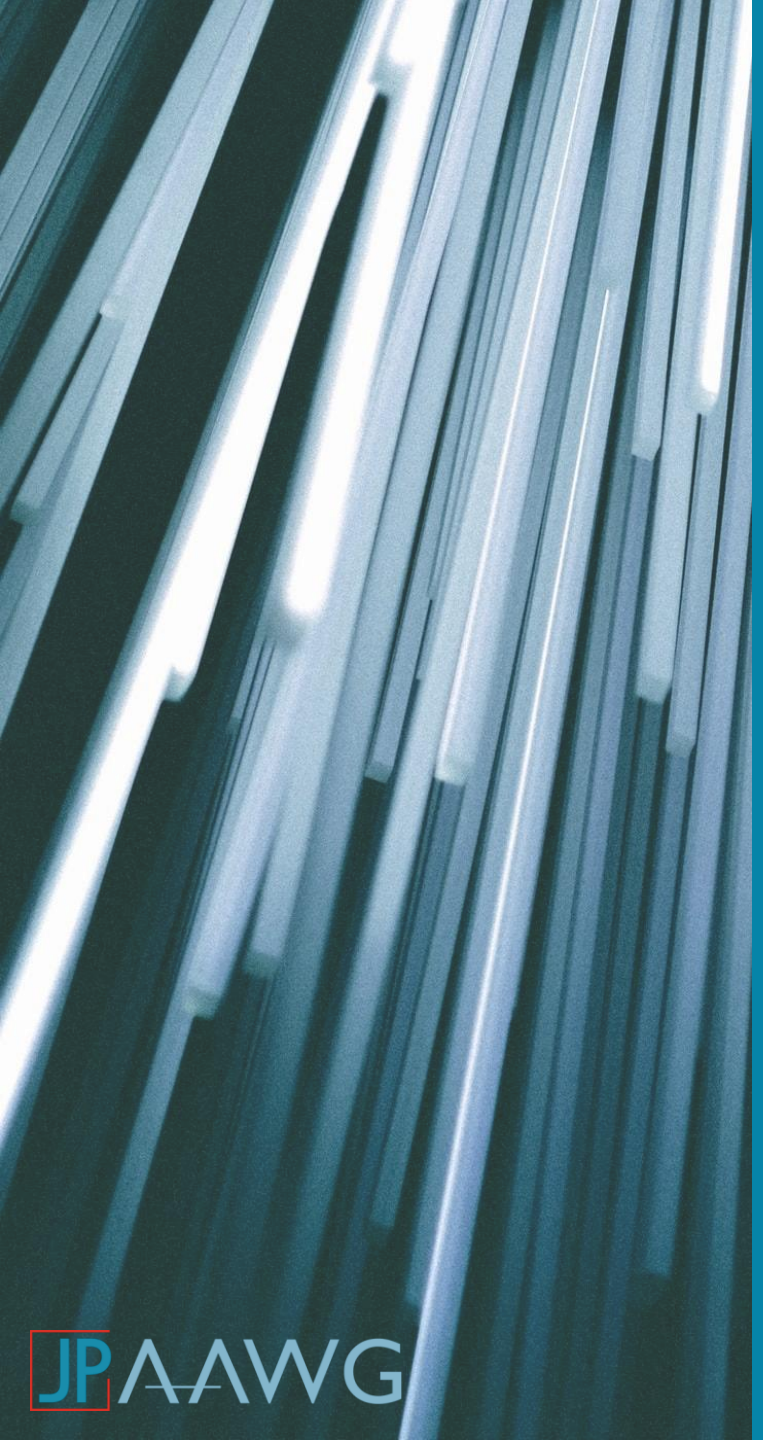
~~→ d=forwarder.example; pp=origin.example~~

- ~~origin.exampleのDNSの_ppレコードには、forwarder.exampleを認可する旨を記述~~

2025/11/3のドラフトで消えた

mv=タグ

- メールの改編履歴のヘッダMailVersionの番号と連動
- 後述



A method for describing changes made to an email

draft-gondwana-dkim2-modification-alegbra-04
(2025-10-17)

JP AAWG

A method for describing changes made to an email

- ヘッダや本文の変更を記録する方法
- MailVersion というヘッダを追加

MailVersionヘッダ

- v= リビジョン番号
- bh= このリビジョンのハッシュ
- bin.n.m= Part毎のハッシュ
- b= 本文のレシピ
- h.header名= ヘッダのレシピ

本文のレシピ

- c: start-end の範囲の行をコピーする
- b: base64で書かれた行をdecodeして挿入
- t: textで書かれた行を挿入
- z: 差分は保存しない。元に戻せない

本文501行目にあったURLの行を削除したときのヘッダ

```
MailVersion: v=2; bh=...;
    b=c:1-500,
    b:PGEgaHJlZj0iaHR0cHM6Ly93d3cuZXhhbXBsZS5jb20iPkV4YW1wbGU8L2E+Cg==,
    c:501-702
```

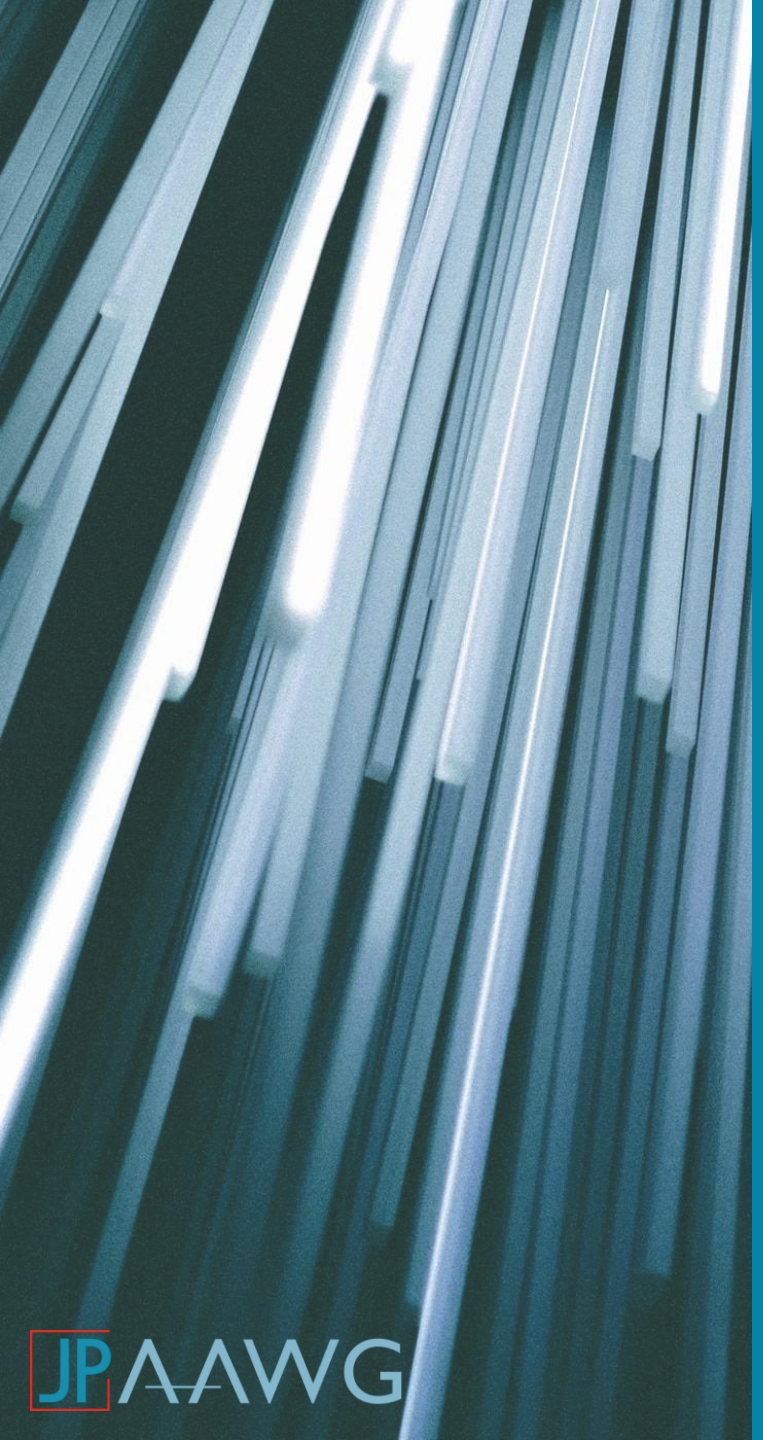
Example

ヘッダのレシピ

- d: int 番目のヘッダを削除、または*で全て削除
- b: base64で書かれたヘッダをdecodeして挿入
- t: textで書かれたヘッダを挿入
- z: 差分は保存しない。元に戻せない

A replacement for DKIMというSubjectを変更した
Reply-Toヘッダを追加した

```
MailVersion: v=3; bh=...;
    h.Subject=d:*,t:A replacement for DKIM;
    h.Reply-To=d:*
```



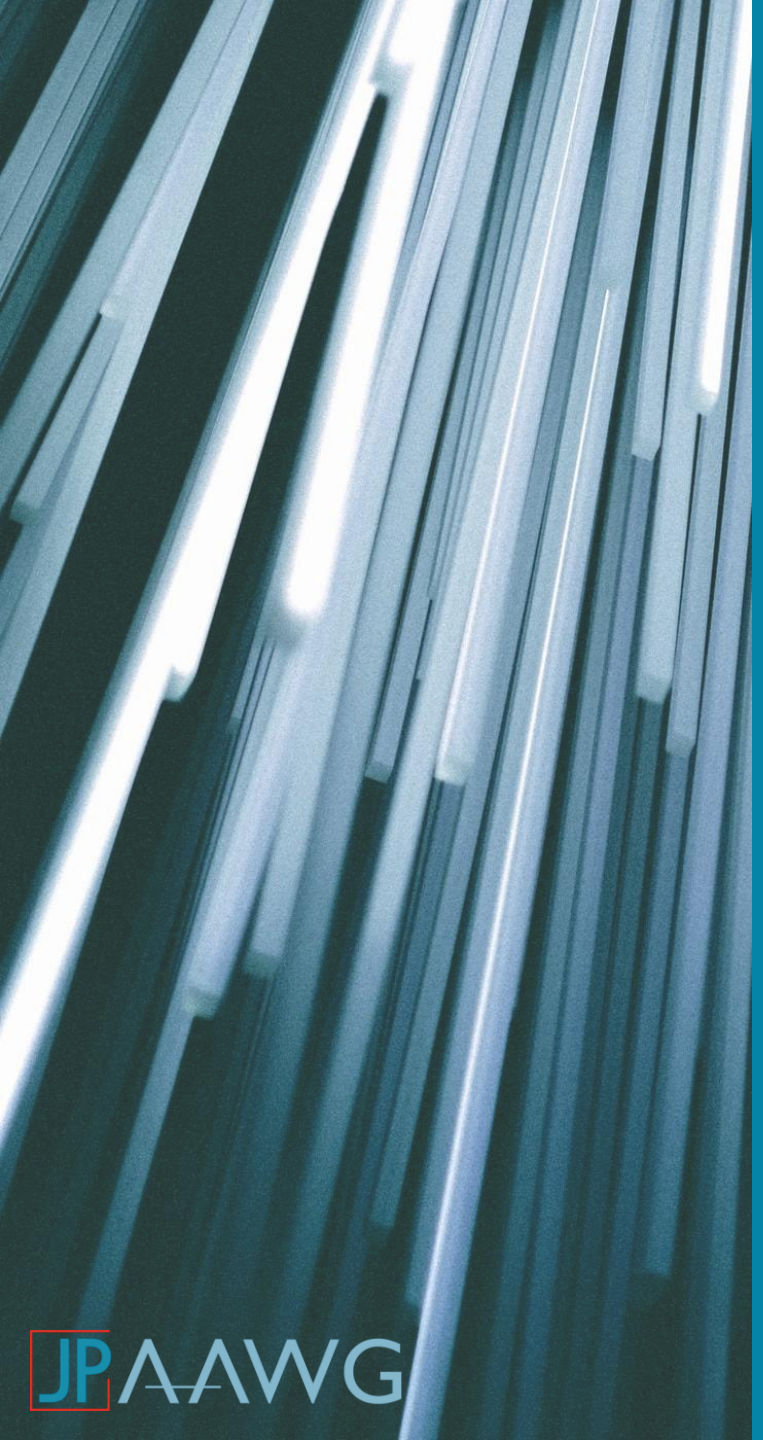
Domain Name Specification for DKIM2

draft-chuang-dkim2-dns-03 (2025-10-20)

JPAAWG

DNSへの鍵の公開方法

- 基本的にはDKIM1と同じ
- k=にed25519が利用可能
  - DKIM1にはRFC8463で追加されたもの
  - DKIM2には標準装備
  - Defaultはrsa
- DKIM2完成時にはed25519-sha256の実装が必須(MUST)に変更される
- sha1はdeprecated



DKIM2 Procedures for bounce processing

draft-robinson-dkim2-bounce-processing-01
(2025-7-7)

JP AAWG

DKIM2 Procedures for bounce processing

- BounceはDKIM2-Signatureヘッダのmf=に対して送る
- これを繰り返して、元の送信者に届く

→ バックスキャッター防止

例: 転送

- a.comからb.comにメールを送信

```
MAIL FROM: <original@a.com>
RCPT TO: <dest@b.com>
DATA
DKIM2: i=1; d=a.com; mf=original@a.com; rt=dest@b.com
From: Sender <original@a.com>
To: <dest@b.com>

I hope this email reaches its destination
```

- b.comがc.comに転送

```
MAIL FROM: <something@b.com>
RCPT TO: <newdest@c.com>
DATA
DKIM2: i=2; d=b.com; mf=something@b.com; rt=newdest@c.com
DKIM2: i=1; d=a.com; mf=original@a.com; rt=dest@b.com
From: Sender <original@a.com>
To: <dest@b.com>

I hope this email reaches its destination
```

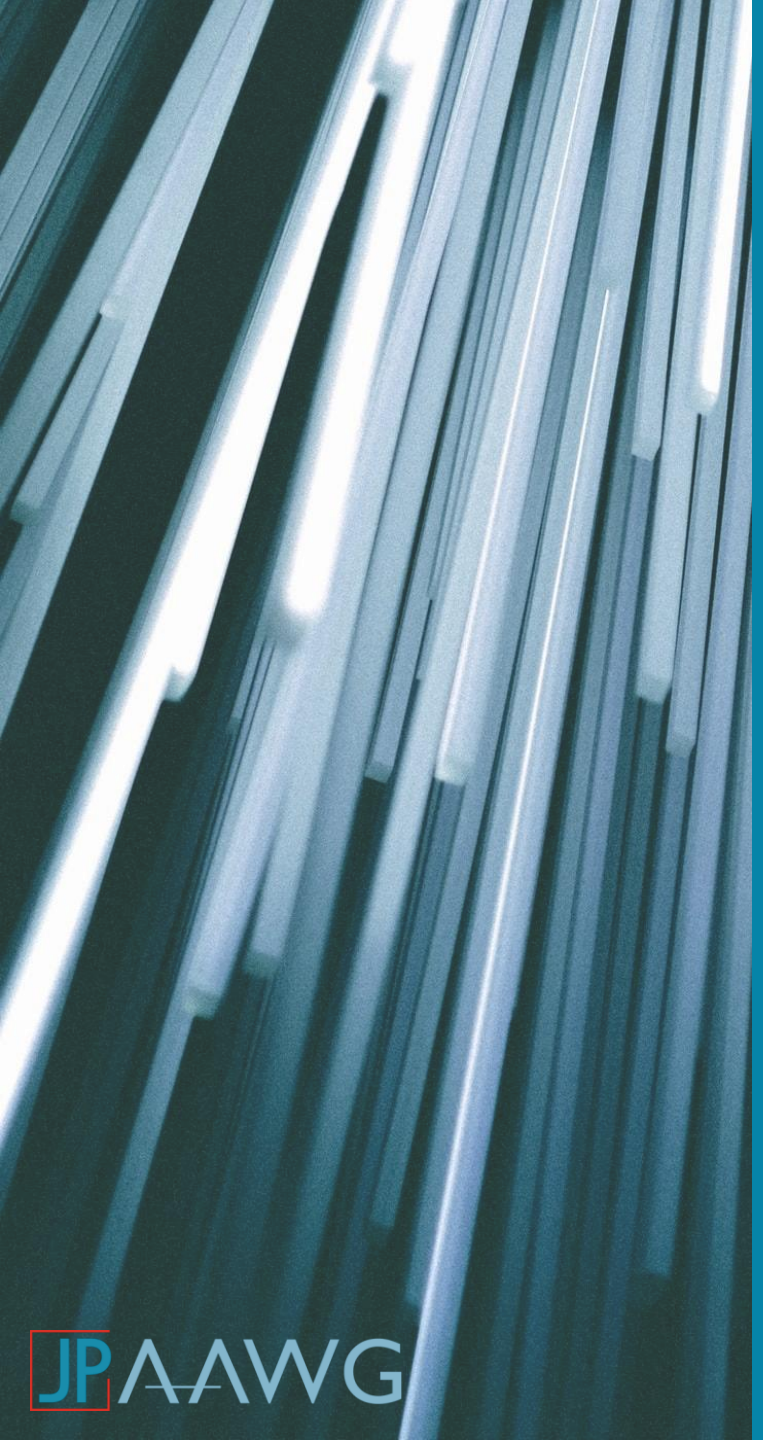
例: 転送先からBouce (c → b)

```
MAIL FROM: <>
RCPT TO: <something@b.com>
DATA
DKIM2: i=1; d=c.com; rt=something@b.com
From: <postmaster@c.com>
To: <something@b.com>
Subject: DSN for ...
Content-Type: multipart/report; boundary="divider43541325151"

--divider43541325151
Content-Type: text/plain

This message is being returned.

--divider43541325151
Content-Type: message/delivery-status

--divider43541325151
Content-Type: message/rfc822
DKIM2: i=2; d=b.com; mf=something@b.com; rt=newdest@c.com
DKIM2: i=1; d=a.com; mf=original@a.com; rt=dest@b.com
From: Sender <original@a.com>
To: <dest@b.com>

I hope this email reaches its destination
--divider43541325151--
```

例: 元の送り先からBouce (b → a)

```
MAIL FROM: <>
RCPT TO: <original@a.com>
DATA
DKIM2: i=1; d=b.com; rt=original@a.com
From: <postmaster@b.com>
To: <original@a.com>
Subject: DSN for ...
Content-Type: multipart/report; boundary="divider89869878"

--divider89869878
Content-Type: text/plain

This message is being returned.

--divider89869878
Content-Type: message/delivery-status

--divider89869878
Content-Type: message/rfc822
DKIM2: i=1; d=a.com; mf=original@a.com; rt=dest@b.com
From: Sender <original@a.com>
To: <dest@b.com>

I hope this email reaches its destination
--divider89869878-- .
```

i=2のヘッダは消えている



Secure SMTP/TLS SRV Announcement

draft-nurpmeso-smtp-tls-srv-06 (2025-7-20)

JP AAWG

Secure SMTP/TLS SRV Announcement

- MXではなくSRVレコードを使う
- Port25はSTART TLS、Port842はImplicit TLS(暗黙的なTLS)

SRVレコード

- RFC2782 (2000/2月)
- ホスト名: _サービス名._プロトコル.名前
- 内容: 優先度　重み　ポート　対象サーバー

```
_abc._tcp.example.jp SRV  10  30  8080  api.example.jp.
                     SRV  10  70  8081  api.example.jp.
                     SRV  20   0  8080  api_back.example.jp.
```

Secure SMTP/TLSの仕組み

- SRVレコードのサービス名は「_smtp-tls」
- portが25の場合はSTART TLS
- portが25以外の場合はImplicit TLS(暗黙のTLS)
  - portは842であるべき
- SRVレコードがあるときはMXは調べない

START TLSの場合

```
_smtp-tls._tcp.example.jp SRV  0  0  25  mail.example.jp
```

Implicit TLSの場合

```
_smtp-tls._tcp.example.jp SRV  0  0  842  mail.example.jp
```

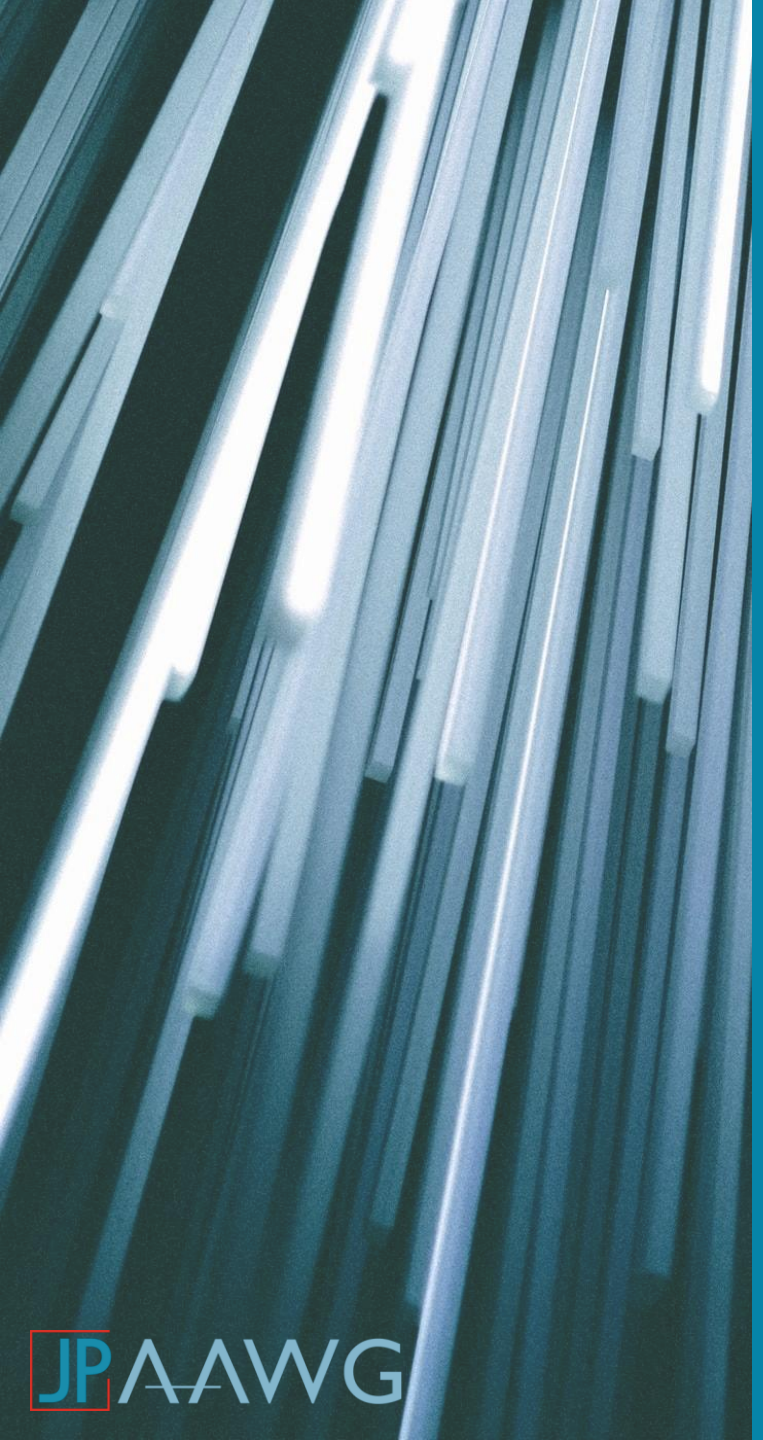
RFC5321bis

draft-ietf-emailcore-rfc5321bis-44 (2025-7-31)

JP AAWG

RFC5321からの変更点

- 基本的な動作は変わっていない
- 構文定義(ABNF)の修正・更新・明確化
- Source Routeに関する説明を削除
  - @example1.jp,@example2.jp:hirano@example3.jp
- 索引などの追加
- EHLOが複数来た場合や引数の明確化
- MTAとMSAの区別
- EHLO, HELOの引数の後ろに余分なテキストを付けることを禁止 (SHOULD NOT → MUST NOT)
  - EHLO localhost abcdefg ✖



RFC5322bis

draft-ietf-emailcore-rfc5322bis-12 (2025-10-3)

JPAAWG

Appendix A.5 空白やコメントの例

RFC5321

```
From: Pete(A nice ¥) chap) <pete(his account)@silly.test(his host)>
To:A Group(Some people)
     :Chris Jones <c@(Chris's host.)public.example>,
         joe@example.org,
  John <jdoe@one.test> (my dear friend); (the end of the group)
Cc:(Empty list)(start)Hidden recipients  :(nobody(that I know))  ;
Date: Thu,
      13
        Feb
          1969
      23:32
               -0330 (Newfoundland Time)
Message-ID:              <testabcd.1234@silly.test>

Testing.
```

RFC5321bis

```
From: Pete(A nice ¥) chap) <pete@silly.test(his host is silly)>
To:A Group(Some people)
     :Ed Jones <e@a.test(.host of Ed)>,
         one@y.test,
  John <jdoe@one.test> (my dear friend); (the end of the group)
Cc:(Empty list)(start)Hidden recipients  :(nobody(that I know))  ;
Date: Thu,
      13
        Feb
          1969
      23:32
               -0330 (Newfoundland Time)
Message-ID:              <testabcd.1234@silly.test>

Testing.
```

Received:で空のトークンに空白を許可

- RFC5322
  - received = "Received:" *received-token ";" date-time CRLF

Received:; Mon, 02 Jan 2006 15:04:05 -0700 ○

Received: ; Mon, 02 Jan 2006 15:04:05 -0700 ✖

- RFC5322bis
  - received = "Received:“
    [1*received-token / CFWS] ";" date-time CRLF

Received:; Mon, 02 Jan 2006 15:04:05 -0700 ○

Received: ; Mon, 02 Jan 2006 15:04:05 -0700 ○

Message-Idの<>の中に名前が付いた

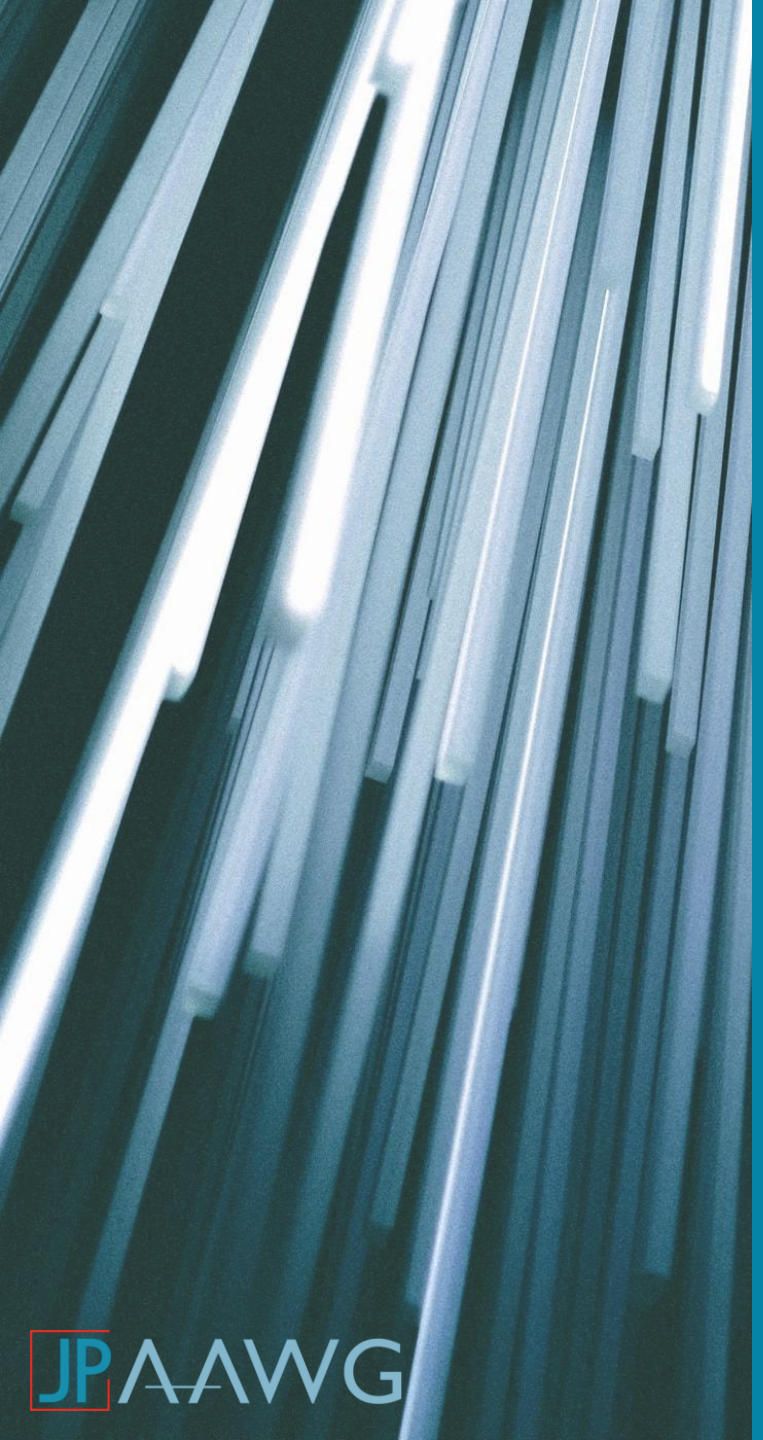
- RFC5322

  msg-id = [CFWS] "<" id-left "@" id-right ">" [CFWS]

- RFC5322bis
  - msg-id = [CFWS] "<" msg-id-internal ">" [CFWS]
    msg-id-internal = id-left "@" id-right

その他RFC5322からの変更点

- 基本的な動作は変わっていない
- タイムゾーンの表記からGMTを排除してUTCに統一
- 「printable」は空白を含む、「VCHAR」は空白を含まない
- その他細かいのたくさん



SMTP Service Extension for Client Identity

draft-storey-smtp-client-id-19 (2025-5-30)

JP AAWG

クライアント認証の追加

- SMTP認証済みクライアントを悪用したなりすましメールなどを防ぎたい
- モバイル端末内にあるSMTPクライアントなどは、ネットワークアドレスやEHLOの値など役に立たない
- SMTPを拡張して、SMTPクライアントが一意な識別子を送信できるようにする
- 識別子を元にSMTPサーバーは判断、制御できる

仕組み

- EHLOの応答: CLIENTID
  - TLS/START TLSが確立している場合のみ送信
- SMTPコマンド
  - CLIENTID　<client-id-type>　<client-id-token>
  - client-id-type: UUID、ライセンス、デバイスID、MACアドレスなど
- CLIENTIDコマンドはAUTHの前でのみ使用可能
- CLIENTIDは1度しか送信できない
- RSETでCLIENTIDは破棄される

ユースケース

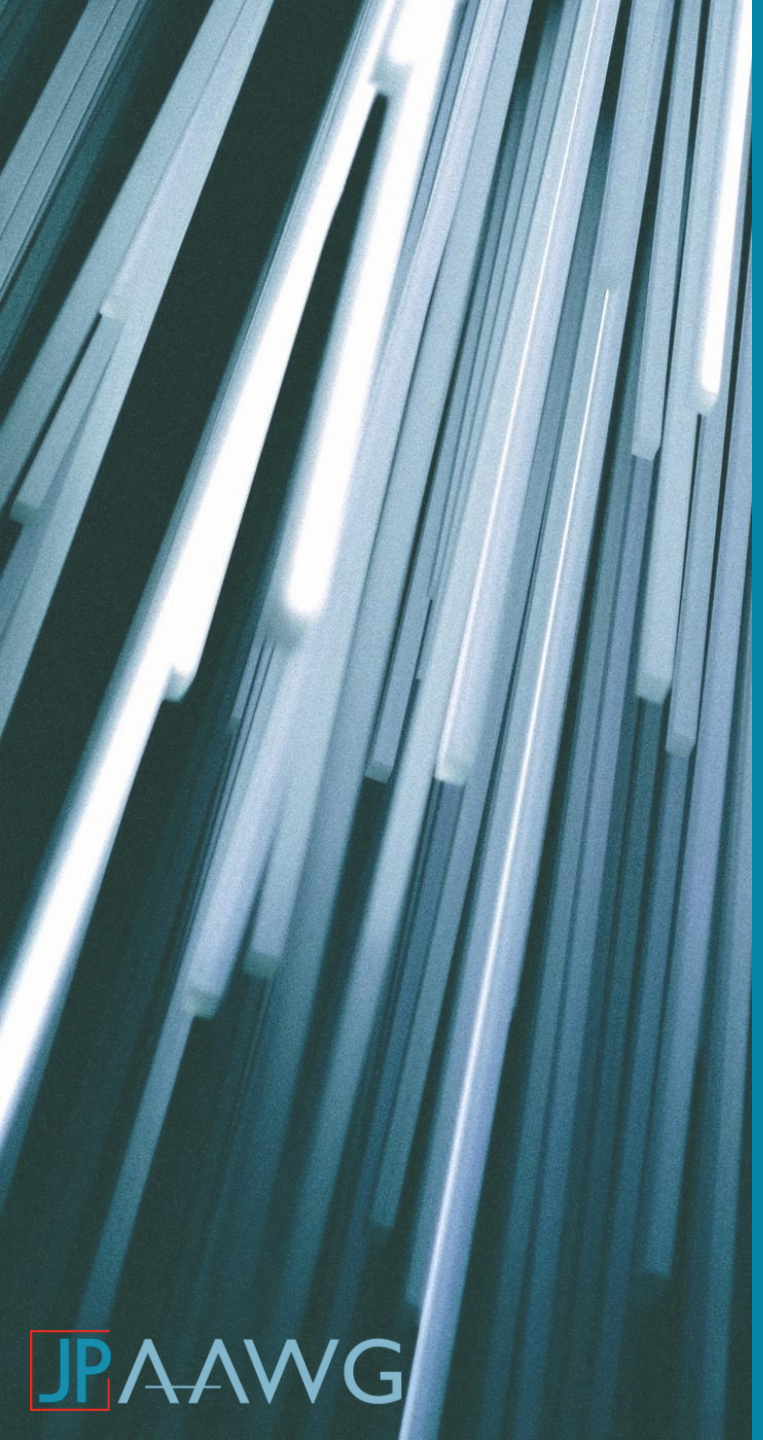
- 特定のデバイスからのみAUTHの使用を許可
- 同一のクライアントからたくさんのAUTHが成功していれば怪しいので制限する
- 過去に使われていたAUTHと違う場合制限する
- ライセンスキーなどをトークンにし、利用を許可する
- CLIENTIDのあり/なしで違うポリシーを適用する
- 複数のデバイスがIPを共有していても、CLIENTIDでレート制限すれば、他に影響を与えない
- CLIENTIDを第3の要素として認証に渡して利用する

例

C: [connection established]
S: 220 server.example.com ESMTP ready
C: EHLO client.example.net
S: 250-server.example.com
S: 250-STARTTLS
S: 250 AUTH LOGIN
C: STARTTLS
S: 220 Go ahead
C: <starts TLS negotiation>
C & S: <negotiate a TLS session>
C & S: <check result of negotiation>
C: EHLO client.example.net
S: 250-server.example.com
S: 250-AUTH LOGIN
S: 250 CLIENTID
C: CLIENTID UUID 23bf83be-aad7-46aa-9e0f-39191ccf402f
S: 250 OK
C: AUTH LOGIN dGVzdAB0ZXN0ADEyMzQ=
S: 235 Authentication successful
C: MAIL FROM:<sender@example.net>
・・・

CLIENTIDを広告

一意なIDを送信



The LIMITS SMTP Service Extension

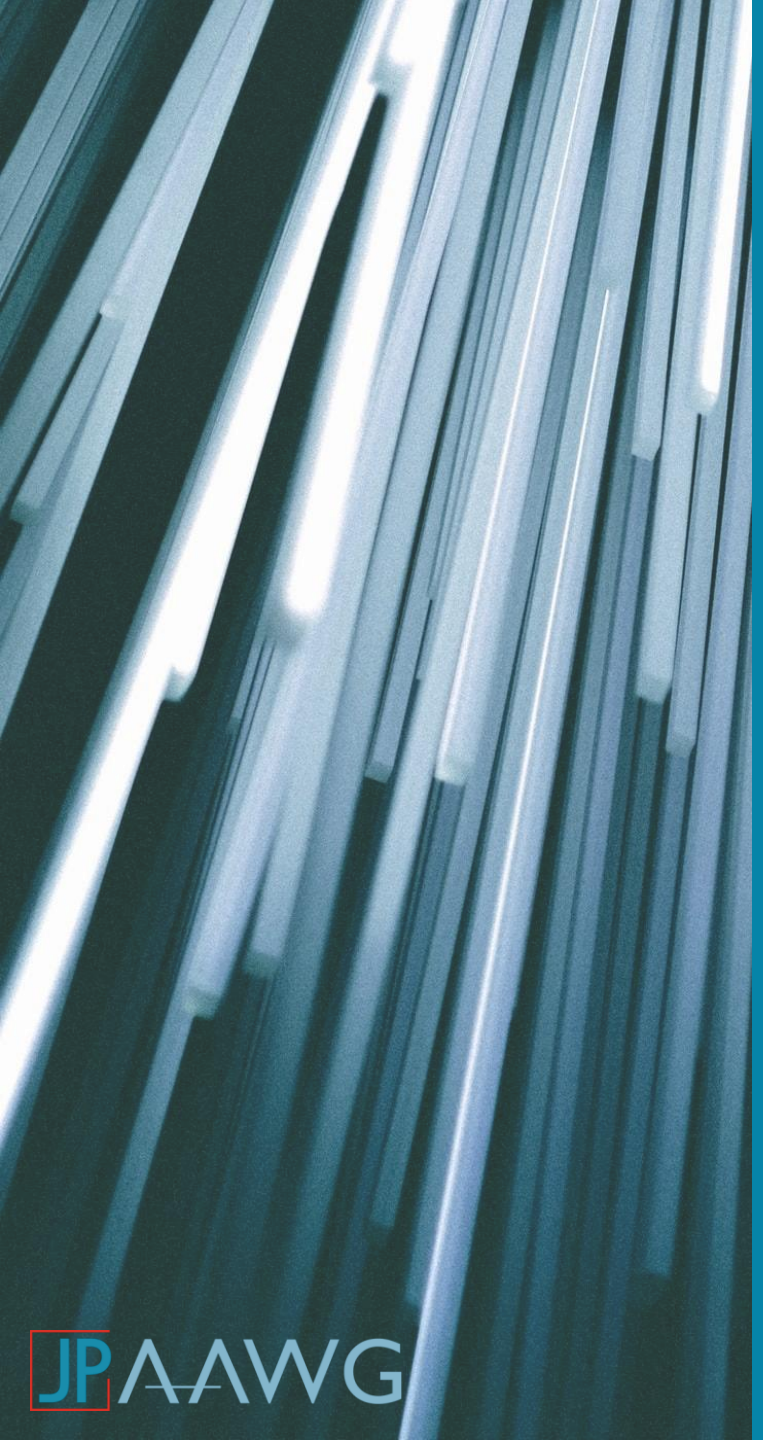
RFC 9422 (2024-2-7)

JPAAWG

EHLOでLimitを返す拡張

- EHLOの応答: LIMITS
- 制限の種類
  - MAILMAX:
    - 1トランザクションで処理できるメール数 (MAIL FROMの回数)
  - RCPTMAX:
    - RCPT TOコマンドの回数
  - RCPTDOMAINMAX:
    - RCPT TOコマンドで渡せるユニークなドメイン数

```
C: EHLO example.org
  S: 250-mail.example.com
  S: 250-LIMITS RCPTMAX=20 MAILMAX=5
  S: 250-SIZE 100000000
  S: 250-8BITMIME
  S: 250-PIPELINING
  S: 250-CHUNKING
  S: 250 STARTTLS
```



Deploying and Using Email in Deep Space

draft-many-tiptop-email-00 (2025-7-21)

JP AAWG

宇宙空間の一連のドラフト

- IP in Deep Space: Key Characteristics, Use Cases and Requirements
  - (draft-ietf-tiptop-usecase-00)
- Deployment and Use of the Domain Name System(DNS) in Deep Space
  - (draft-many-tiptop-dns-01)
- An Architecture for IP in Deep Space
  - (draft-many-tiptop-ip-architecture-02)
- Deploying and Using Email in Deep Space
  - (draft-many-tiptop-email-00)

宇宙空間でのEmail利用

- IMAP, POPやWebメールなどTCPは深宇宙通信には不適切
- 深宇宙ではQUICを使うのがよい
- メールについてはJMAPをREST over HTTP3 over QUICで使うのがよい

地球から深宇宙へメールを送る

1. Earth Mail GatewayがSMTPでメールを受け取る
2. GatewayはメールをBatch SMTP(BSMTP) MIMEオブジェクトに変換する (RFC2442)
3. GatewayはこのBSMTPオブジェクトをHTTP/3 QUICを使って対応する天体側のGatewayに送信する
4. 天体側Gatewayは受信したBSMTPオブジェクトをデコードし、天体側の最終メールサーバーにSMTPで送信する

相手サーバーの見つけ方

- HTTPではURIが必要なので、MXレコードが使えない
- 代わりにURIレコードを使用する

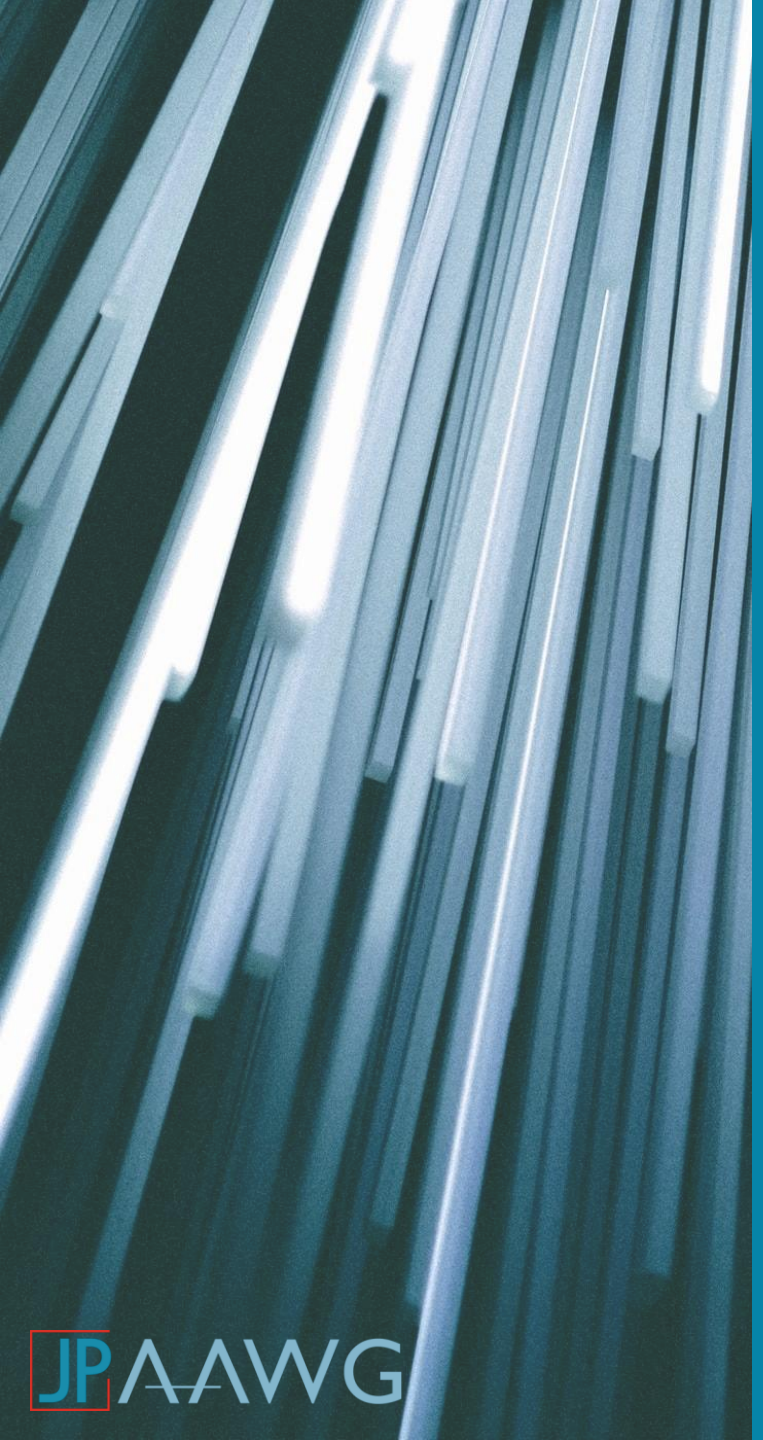
```
_bsmtp._https IN URI 10 1 "https://moonmailgateway1.spaceagency.int/inboundmail"
_bsmtp._https IN URI 20 1 "https://moonmailgateway2.spaceagency.int/inboundmail"
```

- 送り先は非常に少ないので、静的なマップを作ってもよい

その他

- 地球から深宇宙へ送信する場合、URIレコードやそれに対応するIPアドレスは地球上に配置されるか、キャッシュする
- 天体側では事前に同期されたDNSキャッシュを使用する
- メールヘッダ中のドメインは天体側では解決すべきではない

- 不要なメールを減らすために強力なスパムフィルタを地球から出る前に適用する
- 天体側でのスパムフィルタは地球上のDNSを参照しないようにする



BIMI on an Independent MUA

draft-brotman-bimi-mua-00 (2025-6-20)

JP AAWG

概要

- BIMIはDMARC, SPF, DKIMの検証が必要で、特にSPFはMTAしかできない → BIMIの検証はMTAでやる
- 表示はMUA
- MTAとサードパーティーのMUAとの間の相互運用性について考える

仕組み

- MTAはBIMI-Locationヘッダ、BIMI-Indicatorヘッダなどの他に、BIMI-Receiver-Informationヘッダを付ける
- RFC5322形式の日時と、RCPT TOのlocal partをsha256でhashしたメールアドレスを含む

```
BIMI-Receiver-Information: date: Tue, 25 Feb 2023 01:05:55 +0000 ;
  rcpt: 6d9010b2b7a1483b256ae7477738dba7c530bd9ba53db1d6691441e74b83608a@isp.net
```

改ざん防止の仕組み

- DKIMと同じ方法で、BIMI-Receiver-Signatureヘッダを付ける
- 必須の署名対象は、BIMI-Location, BIMI-Selector, BIMI-Receiver-Information

```
BIMI-Receiver-Information: date: Tue, 25 Feb 2023 01:05:55 +0000 ;
  rcpt: 6d9010b2b7a1483b256ae7477738dba7c530bd9ba53db1d6691441e74b83608a@isp.net
```

```
BIMI-Receiver-Signature: v=BIMI1; d=isp.net; s=marketing.example.org.sel_sign;
c=canonicalization; h=BIMI-Location:BIMI-Selector:BIMI-Receiver-Information;
b=<SIGNATURE_BLOB>; t=timestamp
```

- この場合の公開鍵は
  marketing.example.org.sel_sign._local._bimi.isp.net
  に保管

おわり

今日の資料

JP AAWG