利用者端末におけるDMARCを用いたなりすましメール警告システムの設計と実装

田中 俊基[1,a)]　福山 雅深[1,b)]　山井 成良[2,c)]　北川 直哉[2,d)]

概要：公的機関や金融機関等を装ってメールを送信し，個人情報や機密情報の搾取を狙うなりすましメールによる被害は後を絶たず，重大な社会問題となっている．現在，なりすましメールへの対策として，SPFやDKIM等の送信ドメイン認証が利用されている．しかし，なりすましメールの多くはDKIM署名がメッセージに含まれておらず，従来のシステムではDKIM署名がないメールに対し認証を行うことができない．本論文では，DKIM署名がないメールに対し，DMARCを利用してなりすましメールの判定を行う手法を提案する．またこの手法を実装し，DMARCによる判定結果を受信者へ通知するシステムについて報告する．

キーワード：電子メール，送信ドメイン認証，DMARC，DKIM，SPF

Design and Implementation of Spoofed E-mail Alert System Using DMARC for Users' Terminals

Toshiki Tanaka[1,a)]　Masami Fukuyama[1,b)]　Nariyoshi Yamai[2,c)]　Naoya Kitagawa[2,d)]

Abstract: Damages caused by e-mails spoofed from a bank, a public organization and so on become serious social problems. In such e-mails attackers forge the sender address to steal the receivers' personal information and/or secret information. As a countermeasure against spoofed e-mails, sender domain authentication methods such as SPF and DKIM are frequently utilized. However, since most forged e-mails do not include DKIM-signature in their e-mail header, those e-mails cannot be authenticated by the conventional system. In this paper, we propose a method to distinguish spoofed e-mails without DKIM-Signature by using DMARC and implement a system that sends DMARC verification results to receivers.

Keywords: E-Mail, Sender Domain Authentication, DMARC, DKIM，SPF

1. はじめに

インターネットを利用した通信手段の中でも，メールによる通信は広く普及している．しかし，様々な悪意あるメール通信の利用が後を絶たず，社会問題となっている．例えば，通信している二者の間に第三者が割り込んで通信内容の盗聴や通信内容の改ざんを行う中間者攻撃や，受信者に容量の大きいメールや大量のメールを送りつけてサーバやネットワークに過剰な負荷をかけるDDoS(Distributed Denial of Service)攻撃などがある．また，公的機関や金融機関等を装って個人情報を入力するよう促すメールを送信する，フィッシングメールも横行している．これは送信者を詐称してメールを送信することからなりすましメールと呼ばれ，警察庁が注意喚起を呼びかけている[1].

[1] 東京農工大学 工学部　情報工学科
Department of Computer and Information Sciences, Faculty of Engineering, Tokyo University of Agriculture and Technology, Koganei, Tokyo 184–8588, Japan
[2] 東京農工大学 大学院工学研究院 先端情報科学部門
Department of Computer and Information Sciences, The Graduate School of engineering, Tokyo University of Agriculture and Technology, Koganei, Tokyo 184–8588, Japan
[a)] 50012268037@st.tuat.ac.jp
[b)] 50012268052@st.tuat.ac.jp
[c)] nyamai@cc.tuat.ac.jp
[d)] nakit@cc.tuat.ac.jp

なりすましメールへの対策方法として送信ドメイン認証技術がある．代表的な送信ドメイン認証技術として IP アドレスを用いて送信メールサーバの正当性の検査を行う SPF(Sender Policy Framework)[2] と，電子署名によって正当な送信者からの改ざんされていない電子メールであるかどうかを検査する DKIM(DomainKeys Identified Mail)[3] が利用されている．しかし，送信ドメイン認証の結果をメール受信者に通知するシステムは十分に普及していない．また，DKIM 検証では，電子署名がメールヘッダにない場合検証を行うことができない．

本論文では，電子署名がないメールに対し，未認証メールの処理ポリシを送信ドメイン管理者が宣言する仕組みである DMARC(Domain-based Message Authentication, Reporting, and Conformance)[4] を活用することで，なりすましメールの判定を行う手法を提案する．この手法を実現するため，DMARC を用いて送信ドメイン認証を行い，その結果を DMARC ポリシの内容に応じて通知するシステムの設計と実装について述べる．

2. 関連研究

2.1 送信ドメイン認証

現在，よく用いられている送信ドメイン認証技術として，SPF と DKIM がある．

SPF は，送信メールサーバが SMTP(Simple Mail Transfer Protocol) 通信を行う際に用いる IP アドレスとエンベロープ From アドレスのドメインを利用して送信ドメイン認証を行う技術である．この認証を利用するには，送信側はあらかじめ，自らのドメインの DNS(Domain Name System) 権威サーバの TXT レコードに SPF レコードを記述して公開する．SPF レコードには，当該ドメインのメールアドレスを使って送信する可能性のあるサーバを宣言する．受信側は SPF レコードを取得し，接続元であるメールサーバの IP アドレスが含まれていれば認証成功とする．しかし，一般には SPF は転送メールの認証を正しく行うことができない．これは転送メールでは，SMTP 通信の接続元の IP アドレスが，中継サーバの IP アドレスとなるため，エンベロープ From アドレスのドメインで指定された SPF レコードを用いた認証に失敗するためである．

一方 DKIM は電子署名を用いて送信ドメイン認証を行う技術である．この認証を利用する場合，送信側はあらかじめ秘密鍵と公開鍵を用意する．そのうち公開鍵は自らのドメインの DNS 権威サーバで公開する．メール送信時に，秘密鍵を用いてメールのヘッダと本文から電子署名を作成し，DKIM-Signature ヘッダとして付加する．受信側はメール受信時に，DKIM-Signature ヘッダから取得した送信側ドメインの DNS 権威サーバに公開鍵を問い合わせてこれを取得する．次にその公開鍵を用いて電子署名から取り出したハッシュと，ヘッダと本文から作成したハッシュが同じであれば認証成功となる．このため，DKIM による認証では SPF と異なり，転送メールの認証を正しく行うことができる．

表 1　p タグの値とポリシ適用時の動作
Table 1　Values of p-tag and corresponding action policy

p タグの値	ポリシ適用時の動作
none	送信ドメイン認証に失敗しても何もしない．
quarantine	送信ドメイン認証に失敗したメールを隔離する．
reject	送信ドメイン認証に失敗したメールを受信しない．

2.2 POP プロクシを用いた DKIM 検証

筆者らの研究グループでは POP プロクシを用いて，2.1 節で述べた DKIM による送信ドメイン認証を行う手法を提案している [5]．この手法は，通常はメールサービス提供者の受信サーバにて行われる DKIM 検証処理を，メールサービスの利用者が設置した POP プロクシを用いて検証し，その結果を利用者に通知するものである．これにより，学校や企業の運用する受信メールサーバが DKIM に対応していない場合でも，部局等で導入することで，独自に DKIM 検証を行うことができるようになる．さらに，プロクシを設置することで，学校や企業の IT 管理者が認証に失敗したメールの通知方法などを柔軟に変更したり，統計情報によって利用者への注意喚起を行ったりすることも可能となる．この手法では，プロクシはクライアントからメール受信要求を受けると，メールサーバからメールを取得し，DKIM 検証を実行する．検証結果を取得すると，ヘッダに検証結果を添付する．その検証結果に基づき，プロクシもしくは MUA(Mail User Agent) で通知を実行する．

但し，この手法では署名がないメールに対する認証を行うことができない．

2.3 DMARC

認証に失敗したメールの処理方針を指定する技術として，最近 DMARC が普及つつある．この DMARC を用いれば，送信ドメイン認証を実行し，その結果を送信者ドメインの管理者に通知することも可能である．これによりドメイン管理者はこの通知によって自らが意図した認証がどの程度正しく適用できているかを知り，認証の精度を向上させることができる．この仕組みを利用するには，送信側が自ドメインを SPF と DKIM の両方，もしくは，いずれか一方に対応させる必要がある．さらに，送信側は DMARC レコードを自ドメインの DNS 権威サーバで公開する必要がある．DMARC レコードでは，受信メールが送信ドメイン認証に失敗した際に，そのメールに対する処理動作と，認証結果の報告先を指定する．受信側では，メール受信時に送信ドメイン認証 (SPF と DKIM) を行い，両方の認証に失敗したとき DMARC ポリシを適用する．

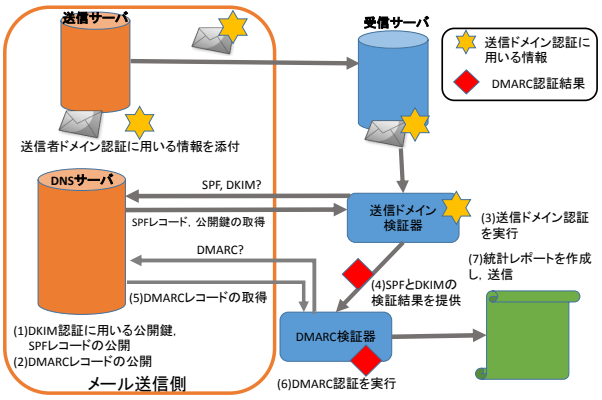

図 1　DMARC 認証の流れ
Fig. 1　Flow of DMARC authentication

DMARC では，メール送信側はあらかじめ自ドメインの DNS 権威サーバの DMARC レコードの “p タグ” で，送信ドメイン認証失敗時の動作ポリシを記述する．p タグの値と，そのポリシ適用時に実行する動作を**表 1** に示す．

例として，メール送信側 (example.com) が DMARC を利用している場合を考える．このとき，メール送信側は DMARC レコードを「_dmarc.example.com」の TXT レコードとして，次のように公開していたとする．

v=DMARC1;p=none;rua=mailto:reports@example.com

この例の場合，p タグの値は “none” であり，example.com からのメールについて，DMARC 認証に失敗した場合でもメール受信側がメールの隔離や受信拒否等の動作を行わないように指示している．また，rua で指定されている reports@example.com というメールアドレス宛に集約したレポートの送信を行うように記述している．

DMARC による検証は，**図 1** に示す手順で行われる．

(1) メール送信側は，自ドメインを SPF と DKIM の両方，もしくはいずれかに対応させる．
(2) メール送信側は，自ドメインの DNS 権威サーバの TXT レコードに DMARC レコードを公開する．
(3) 送信ドメイン検証器は，送信元 DNS 権威サーバに問い合わせ SPF レコードおよび DKIM 公開鍵を取得し，SPF と DKIM の認証を行う．
(4) 送信ドメイン検証器は，SPF と DKIM の認証結果を DMARC 検証器に渡す．
(5) DMARC 検証器はメールのヘッダ From ドメインの DNS 権威サーバへ DMARC レコードの問合せを行う．
(6) DMARC ポリシを取得できたら，DMARC 検証器は SPF と DKIM の認証結果に基づき，DMARC ポリシを適用する．
(7) DMARC 検証器は送信ドメイン認証の結果やポリシの適用状況を集約した統計レポートを作成し，rua で指定されたメールアドレスへ送信する．

3. DMARC による検証を利用したシステムの設計

3.1 本システムの概要

2.2 節のシステムでは，電子署名がないメールに対し DKIM 検証を行うことができない．そこで本システムでは，本来なら電子署名があるはずの送信元からのメールであるにもかかわらず電子署名がないメールに対し，なりすましメールの判定を行い，その判定結果を受信者に通知することを目的とする．このようなメールをなりすましメールであると判定するため DMARC を利用する．DMARC では SPF のみを採用しているドメインでもレコードを公開できるが，現時点では DKIM を採用しているドメインが多いと考えている．また，SPF だけを採用しているドメインであっても，例えば任意の IP アドレスに対して認証に成功するように設定しているドメインは少ないと考えている．このため DMARC レコードを公開しているドメインに対して DMARC 認証を行うとなりすましメールを高い信頼度で判定できると思われる．なお，通常の DMARC による検証で行われる，2.3 節で述べたレポートの作成および送信は本システムでは行わない．

受信したメールの送信ドメイン認証および DMARC による検証と結果の通知はすべて利用者端末にて行われる．利用者端末において検証および通知を行うことで，端末所有者が利用している受信サーバが検証を行っていなくても容易に送信ドメイン認証および DMARC による検証を導入することができる．SPF に必要な情報は，メールヘッダの “Received” フィールドから取得する．自組織と外部組織の境界を決め，その境界にもっとも近い外部組織の IP アドレスと，“Return-Path” のメールアドレスを送信元情報として SPF 認証に利用する．SPF と DKIM 認証は**図 2** に示す送信ドメイン認証モジュールで行う．DMARC 検証モジュールは送信ドメイン認証の結果を受け取り DMARC ポリシを適用するかどうか判断する．DMARC 検証モジュールは検証結果として “pass”，もしくは “fail” を返す．受信者には，成功，失敗に関わらず検証結果を通知し，失敗時には適用されたポリシを通知する．

3.2 システムの動作設計

本システムにおける POP プロクシおよびクライアントの動作を図 2 に示す．

(1) プロクシでは，MUA からメッセージ取得コマンドを受けると，POP サーバへコマンドを中継する．
(2) プロクシで，取得したメールのヘッダから認証に用いる情報を取り出し，送信ドメイン認証モジュールに入力する．
(3) 送信ドメイン認証モジュールは，ヘッダから得た情報を基に送信ドメイン認証 (SPF と DKIM) を実行する．

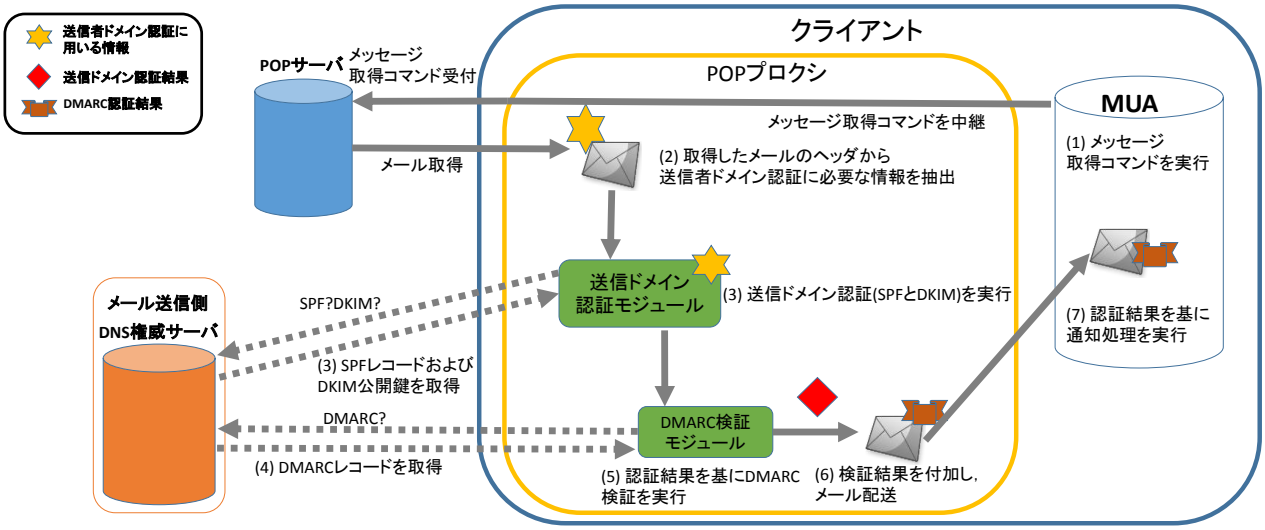

図 2　本システムの DMARC 認証手順
Fig. 2　DMARC authentication flow of proposed system

(4) DMARC 検証モジュールは，DNS 権威サーバに DMARC レコードを問い合わせ，取得する．
(5) DMARC 検証モジュールは送信ドメイン認証の結果を基に，DMARC ポリシを適用する．
(6) プロクシは，認証結果をメールヘッダに付加し，MUA へメールを配送する．
(7) MUA は DMARC 検証結果を基に通知処理を行う．

DMARC 検証は通常，メール送信側が意図した認証が行われているかどうかを知り，認証の精度を向上させるために用いられる．本システムでは，(6)，(7) で示したように，受信側にその結果を通知する．この通知を行うことで，当該送信ドメインをブラックリストに入れたり，メールを隔離したりする等の対策を受信者に促すことができる．

4. DMARC を用いた検証結果通知システムの実装

4.1 システム実装法

3.2 節で述べた設計を基に，Perl 言語によるシステムの実装を行った．本システムは，DMARC 検証を行うため，CPAN 上にある Perl モジュール Mail::DMARC，および Mail::DMARC::PurePerl[6] を使用した．メールヘッダから認証に必要な情報を取得するため，MIME::Parser[7] および，Net::Server::POP3proxy を基にして作成した Net::Server::POP3proxySSL を使用した．また，SPF による認証を行うため Mail::SPF[8] を，DKIM による認証を行うため Mail::DKIM::Verifier[6] を利用した．また，これらはすべて Cygwin 上で実装することにより，利用者の端末上で動作するようにした．

メールヘッダから認証に必要な情報を取得する部分について説明する．認証に必要な部分はメールヘッダ中の次に示す部分である．

```
Return-Path:  <sender@example.com>
略
DKIM-Signature:v=1; a=rsa-sha256;
  c=relaxed/relaxed;
  d=example.com;
  s=20120113;
  h=mime-version:date:message-id:subject:from:to:
   content-type;
  bh=YzODIQzFL5CIwg3H6lYD6ZafgsQR/7HxA6gRkSc7Vvg=;
  b=Jd6cf0fJGsMyekr7dpUL6jxVywqRXhkKeBcdFYdSk/KzuHK
   Zsyg/3iJMNlQq7wtDT6wU9OujAoEnPQirUwCHLFCJHqWklii
   DBva56Ec5nuGXAxsjLCU3XwwMQ1AcBcGSepS+e5kozZFBG7I
   tOoZ5eXBXEyAAvChoLgujjnUHJtS6uYOuSC6pEVlHpyguzm+
   bVk97/wOdxc6W4Z8xaWMneN6KBLod28r7KORNgU8K6WGKkwj
   fcY1lkm1KBuW3X9YkR8nVmhXjsRIyEhz256a3WLYqKbC7cPH
   aBK8xFVHzE1AoZwhsgMRCsewRCR9026OkWSvpuVvk+qN5Csa
   rxWxmA==
略
From:  <sender@example.com>
To:  receiver@example.com
```

図 3　メールヘッダの例
Fig. 3　A sample of e-mail header

図 3 中の情報が，ヘッダ中の DMARC による認証に利用する情報である．Return-Path，DKIM-Signature，ヘッダ From，ヘッダ To を利用する．

図 4 に，ヘッダから情報を取得するコードを示す．メールヘッダから必要な情報を以下に示す手順で取得する．

2-3 行目　MIME::Parser を用いてヘッダ情報を解析する．また，ファイルの一時保存場所を指定する．
5-6 行目　メールヘッダ情報を取得する．本プログラムでは，すでに POP3proxySSL モジュールによって$_[0]

```
 1  #ヘッダ情報取得
 2  my $parse = new MIME::Parser;
 3  $parse->output_to_core(1);
 4  #ヘッダ解析と整形
 5  my $buf = ($_[0]);
 6  my $entity = $parse->parse_data($buf);
 7  my $h_from_org = $entity->head->get('from');
 8  my $e_to_org = $entity->head->get('to');
 9  $_[0] =~/Return-Path:  <(.+)>/;
10  my $e_from_org = $1;
11  $e_from_org =~/@(.+)/;
12  my $e_from = $1;
13  $e_to_org =~/@(.+)/;
14  my $e_to = $1;
15  $e_to =~s/>//g;
16  $h_from_org =~/@(.+)/;
17  my $head_from = $1;
18  $head_from =~s/>//g;
19  my $s_ip = inet_ntoa(inet_aton($head_from));
```

図 4　ソースコード：ヘッダ取得
Fig. 4　Source code: acquisition of e-mail header

にメールヘッダ情報が格納されているため，$_[0] を利用する．

7 行目　メールヘッダに記載されているヘッダ From を取得する．

8 行目　メールヘッダの宛先からヘッダ To を取得する．

9 行目　メールヘッダ中の “Return-Path” で指定されたドメインをエンベロープ From とし取得する．

10-12 行目　取得したエンベロープ From を DMARC 認証モジュールに入力する形に整える．

13-18 行目　エンベロープ From 同様，DMARC 認証モジュールに必要な情報を整形する．

19 行目　ヘッダ From の IP アドレスを取得する．

続いて SPF に用いる情報を取り出す ‘received” フィールドの解析を実行するコードを**図 5** に示す．また，以下の手順で実行する．

2 行目　“Received” フィールドの数を計算する

3-8 行目　利用する変数を初期化する．

9 行目　“Received” フィールドの数だけ次の動作を繰り返す．

10 行目　“Received” フィールドを取得する．

11 行目　フラグを初期化しておく．

12-14 行目　取得した情報に自組織の情報があればフラグを立てる．

15-19 行目　取得中の情報から送信元と思われるドメイン名と IP アドレスを取得する．

20-24 行目　フラグがたっている場合，現在の情報を保存しておく．

```
 1 #received フィールド読み込み解析
 2 my $r_cnt = $entity->head->cnt('received');
 3 my $cnt = 0;
 4 my $spfip = 0;
 5 my $spfip0 = 0;
 6 my $spfflag = 0;
 7 my $ipdecide = 0;
 8 my $spfsenderid = 0;
 9 while($cnt <$r_cnt){
10  my $r_id=$entity->head->get('received', $cnt);
11  $spfflag = 0;
12  if($r_id =~/mta-eaxmple/){
13   $spfflag = 1;
14  }
15  my $r_ip = $r_id;
16  $r_id =~/((.+)[/;
17  my $spfid = $1;
18  $r_ip =~/[(.+)]/;
19  $spfip = $1;
20  if($spfflag == 1){
21   $spfip0 = $1;
22   $spfsenderid = $spfid;
23   $ipdecide = $cnt;
24  }
25  $cnt ++;
26 }
27 $spfip = $spfip0;
28 if($spfip !~ /^(\d+).(\d+).(\d+).(\d+)$/){
29  $spfip =~/[(.+)/;
30 }
```

図 5　ソースコード：“Received” フィールドの解析
Fig. 5　Source code: analysis of “Received” field

25,26 行目　次の情報を取得するため変数を進める．

27 行目　送信元と思われる情報をモジュールに渡す変数に格納する．

28-30 行目　取得した情報を整形する．

送信ドメイン認証 (SPF と DKIM) を実行するコードを**図 6** に示す．また，以下の手順で実行する．

2 行目　DKIM 検証モジュールを実行する．

3 行目　メールヘッダの内容を渡す．

6,7 行目　SPF 認証に用いる変数を初期化する．

8 行目　IP アドレスが取得できている場合認証を行う．

9 行目　SPF 認証モジュールを実行する．

10-15 行目　モジュールに必要な情報を入力する．

16-17 行目　認証結果を取得する．

19-21 行目　IP アドレスが取得できていない場合，SPF 認証結果を “temperror” とする．

続いて DMARC 認証を実行するコードを**図 7** に示す．図 7 で示すコードは以下の処理を実施する．

```
 1 #DKIM 実行
 2 my $dkim = Mail::DKIM::Verifier->new();
 3 $dkim->PRINT("$_[0]");
 4 $dkim->CLOSE;
 5 #SPF 認証実行
 6 my $spf_result = 0;
 7 my $spf_code = 0;
 8 if($spfip ne ""){
 9  my $spf_server = Mail::SPF::Server->new();
10  my $request = Mail::SPF::Request->new(
11   versions => [1],
12   scope => 'mfrom',
13   identity => $e_from_org,
14   ip_address => $spfip
15  );
16  $spf_result = $spf_server->process($request);
17  $spf_code = $spf_result->code;
18 }
19 else{
20  $spf_code = 'temperror';
21 }
```

図 6　ソースコード：送信ドメイン認証の実行
Fig. 6　Source code: sender domain authentication

```
 1 #DMARC 実行
 2 my $dmarc = Mail::DMARC::PurePerl->new;
 3 $dmarc->source_ip($s_ip);
 4 $dmarc->envelope_to($e_to);
 5 $dmarc->envelope_from($e_from);
 6 $dmarc->header_from($head_from);
 7 $dmarc->($dkim);
 8 $dmarc->spf([
 9  {
10   domain => $e_from,
11   scope => 'mfrom',
12   result => $spf_code,
13  }
14 ]);
```

図 7　ソースコード：DMARC 認証の実行
Fig. 7　Source code: DMARC authentication

2 行目　DKIM 検証モジュールを実行する．
3 行目　DKIM 検証モジュールにヘッダ情報を入力する．
6 行目　DMARC 認証モジュールの使用を宣言する．
7-10 行目　DMARC 認証モジュールに，DMARC 認証に必要な情報を入力する．
11 行目　DKIM 検証結果を DMARC 認証モジュールに入力する．
12-16 行目　SPF 検証結果を DMARC 認証モジュールに入力する．

```
 1 #定型作成
 2 my $prefix = "X-Dmarc-Result: ";
 3 # DMARC 認証結果の取得
 4 my $result = $dmarc->validate();
 5 if ($result->result eq 'pass') {
 6  $result = $result->result;
 7  $_[0] = "${prefix}${result}\r\n${_[0]}";
 8  return $_[0];
 9 }
10 if ($result->result eq 'fail' ) {
11  my $reason = $result->disposition;
12  if ($reason eq "none") {
13   my $why1 = $result->reason->[0]{type};
14   my $why2 = $result->reason->[0]{comment};
15   if ($why1 ne '' and $why1 eq 'other') {
16    $_[0] = "${prefix}${why1},${why2}\r\n${_[0]}";
17    return $_[0];
18   }
19  }
20  $result = $result->result;
21  $_[0]="${prefix}${result}($reason)\r\n${_[0]}";
22  return $_[0];
23 }
```

図 8　ソースコード：ヘッダへの DMARC 認証結果の付加
Fig. 8　Source code: addition of DMARC authentication result into e-mail header

続いて，DMARC 認証結果をメールヘッダに付加するコードを**図 8** に示す．図 8 で示すコードは以下の処理を実施する．

2 行目　メールヘッダに結果を付加するためのプリフィックスを作成する．
4 行目　DMARC 認証モジュールの validate メソッドを用いて，DMARC 認証結果を取得する．
5-8 行目　DMARC 認証結果が “pass” の時，その結果をメールヘッダに付加し，メールを配送する．
10-11 行目　認証結果が “fail” の時，適用されたポリシ内容を取得する．適用されたポリシは，パラメータ “disposition” に記載される．
12-14 行目　適用されたポリシが “none” である時，DMARC による認証が行われていない場合がある．その場合，validate メソッドのパラメータ “reason” にその原因が記載されているため，その原因によって処理を変更する．
15-17 行目　DMARC 認証が行われていない場合，なぜ行われなかったかをメールヘッダに付加し，メールを配送する．
20-22 行目　DMARC 認証が行われ，認証に失敗した時，ポリシ適用内容と失敗した結果をメールヘッダに付加し，メールを配送する．

4.2 システムの動作確認方法

本システムの動作を確認するため，DMARC に対応しているドメインとして，DMARC.org に参加しているフリーメールの Gmail，Yahoo! Mail[9]，DMARC に対応していないドメインとして，DMARC.org に参加していない Yahoo!Japan Mail を使用した．これらのメールアドレスから本システムに対してメールを送信することで DMARC ポリシの適用と，受信者への警告の通知の確認も行った．この警告を確認するため，DMARC モジュールに SPF と DKIM の認証結果を “fail” として渡し，ポリシの適用がされ，結果通知がされているか確認した．

4.3 DMARC 認証結果の通知

DMARC 認証の結果を受信者に通知する．DMARC 認証の結果は，認証をパスした際の “pass” に，表 1 で示した 3 つのポリシを加えた 4 種類とした．

また，通知方法は 2 種類とした．1 つは MUA の「ラベル」や「フィルタ」の機能による分類，もう 1 つは，ポップアップによる警告である．

MUA の機能による分類では，あらかじめルールやフィルタを MUA に設定しておき，ルールに一致したメールを受信すると，ルールに応じた分類が行われるようにした．本システムでは，「Microsoft Outlook 2013」(以下 Outlook) の仕分けルールを用いてこの分類を実装した．この Outlook の機能により，次に示すように，認証成功時には青色 (**図 9**)，“none” と “quarantine” を含む認証失敗時には黄色 (**図 10**)，“reject” が適用された場合は，赤色を設定した (**図 11**)．また，ポリシが見つからず，DMARC 認証を実行できなかった場合は，橙色のラベルを設定した (**図 12**)．

ポップアップによる警告では，適用されたポリシが “reject” のときのみ，警告が行われるようにした．この警告は，MUA の機能拡張 (アドイン) を用いたものである．本システムで作成したアドインを利用すると，プロクシにて付加した DMARC 認証結果が “reject” を適用したことを示すとき，**図 13** に示すように通知ウィンドウが表示されるようにした．

適用されたポリシが “reject” の場合，その送信元ドメインから送信される正当なメールにはすべて DKIM 署名が付与されるはずであるにもかかわらず，当該メールには署名がないか認証に失敗したことを意味する．そのため，当該メールはなりすましや改ざんの可能性が非常に高いと考えられる．そこで，“reject” が適用されたメールに対しては分類による通知に加えてポップアップによる警告も行うようにした．

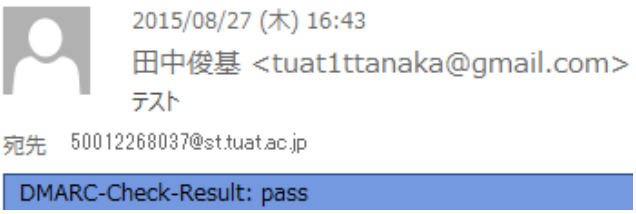

図 9 ラベル付加 (pass)
Fig. 9 Addition of “pass” label

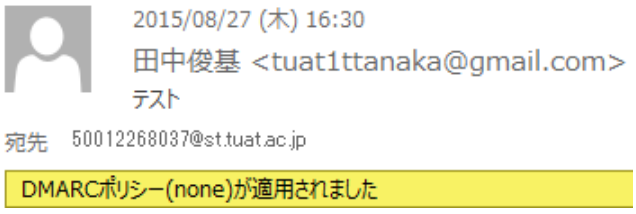

図 10 ラベル付加 (none)
Fig. 10 Addition of “none” label

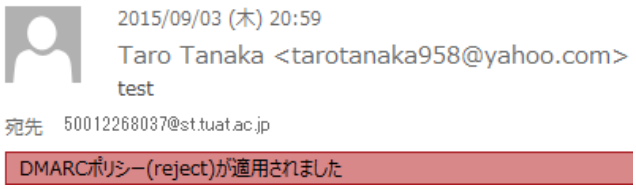

図 11 ラベル付加 (reject)
Fig. 11 Addition of “reject” label

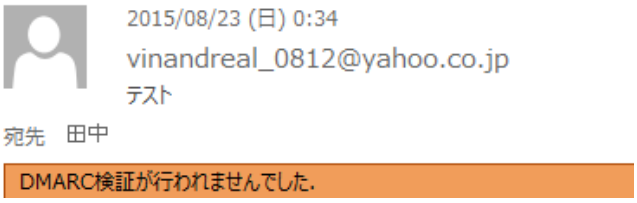

図 12 ラベル付加 (other)
Fig. 12 Addition of “no-DMARC-Check” label

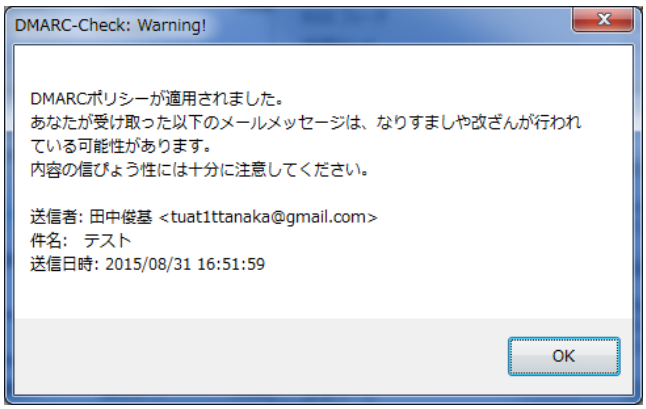

図 13 ポップアップ通知
Fig. 13 Pop-up notification window

5. おわりに

本論文では，SPF と DKIM による検証と，その認証結果を利用した DMARC による検証を行い，その結果を受信者に通知するシステムについて述べた．受信者は，DMARC ポリシの適用結果を得ることにより，当該メールの送信ド

メインをブラックリストに登録したり，当該メールを隔離する等，各々のポリシに応じて対処することが可能となる．これにより，送信ドメインが指定した処理以外のなりすましメール対策も可能となり，DMARC の普及促進につなげることが期待できる．

今後の課題として，SPF に用いる送信元情報の特定方法をより汎用化することが挙げられる．本システムでは，自組織の “Received” フィールドの特徴がわかっているため自組織と外部組織の境界線が特定することが可能である．しかし，大きな組織になると利用者が自組織の “Received” フィールドの特徴がわからず送信元を特定することが難しくなってしまう．したがって送信元情報特定のためには，本システムの導入前に多数のメール情報を収集し自組織の “Received” フィールドの特徴を把握する仕組みが必要である．加えて，本システムは POP プロクシを利用しており，IMAP には対応していないため，IMAP への対応も今後の課題となる．

参考文献

[1] 警察庁：銀行を装うフィッシングメールについて (注意喚起)(online): `http://www.npa.go.jp/cyber/warning/chuikanki/bankphish.htm` (2015 年 9 月 10 日参照)

[2] M. Wong and W. Schlitt: *“Sender Policy Framework (SPF) for Authorizing Use of Domains in E-Mail, Version1”*, RFC4408, IETF，2006.

[3] D. Crocker, Ed., Brandenburg InternetWorking, T.Hansen, Ed., AT&T Laboratories, M. Kucherawy, Ed.and Cloudmark: *“DomainKeys Identified Mail(DKIM) Signatures”*, RFC6376, IETF，2011.

[4] M. Kucherawy, e. Zwicky: *“Domain-based Message Authentication, Reporting, and Conformance (DMARC)”*, RFC7489, IETF，2015.

[5] 福山 雅深，大岩 美春，山井 成良，北川 直哉：POP プロクシを用いた DKIM 検証システムの実装，研究報告インターネットと運用技術 (IOT)，Vol.2015-IOT-28，NO.2，pp.1-6，2015.

[6] CPAN:Mail::DMARC.pm(online)，`http://search.cpan.org/~msimerson/Mail-DMARC-1.20150527/lib/Mail/DMARC.pm` (2015 年 9 月 10 日参照)

[7] CPAN:MIME::Parser.pm(online)，`http://search.cpan.org/~dskoll/MIME-tools-5.506/lib/MIME/Parser.pm` (2015 年 9 月 10 日参照)

[8] CPAN:Mail::SPF.pm(online)，`http://search.cpan.org/~jmehnle/Mail-SPF-v2.9.0/lib/Mail/SPF.pm` (2015 年 10 月 28 日参照)

[9] Leading Email Senders and Providers to Combat Email Phishing through DMARC.org(online)，`https://dmarc.org/press/release-20120130/` (2015 年 10 月 28 日参照)