Spoofer on Attack: The Exploration of Email Sender Authentication

Author	ID	Major
RUiN	*	Digital Media Technology
SA	*	Information Security
XJC	*	Information Security
XHY	*	Information Security

1. Introduction and Relative Work

1.1 Background

Component-based software design is a primary engineering approach for building modern software systems. When various components work together, because of different interpretations of messages or working protocols, the inconsistence between components may lead to insecurity.

As the well-known mail transfer protocol, SMTP lacked mechanisms to protect the process of email sending and receiving. In an email, there are many identities, such as MAIL FROM, HELO, Sender and so on. But SMTP does not enforce consistencies among these which may lead to insecurity. To prevent email spoofing and other security issue, three protocols —— SPF, DKIM, and DMARC are put into use. But inconsistencies between different components could lead to security vulnerabilities. Because the verification of each step is separate. The attackers can exploit the inconsistencies between different internal components in an e-mail system.

To use the mail system freely, we aim to achieve some security requirement, that is the end-user Bob who uses email client C to receive an email from receiving server R can determine that the message is indeed from user Alice of sending server S. Under this expectation, we are supposed to prove or ensure that (1) The From header of the email that S sends matches the authenticated username (other users of S cannot spoof Alice's address); (2) SPF/DKIM/DMARC components in R can obtain S's DNS correct policy; (3) SPF/DKIM and DMARC components in R consistently authenticate the same identifier; (4) the identifier that R authenticates is consistent with the identifier that C shows to Bob.

But in reality, there exist packs of challenges. The first is working between different components. Although each specific system is fully functional, but when they work at the same time, something terrible may happen. The second is tensions with the robust principle, when facing ambiguous circumstance or potentially dangerous client, the actions different components take can lead to numerous email spoofing attacks. The last one is the danger of feature composition which is similar to the first one.

Because of the hidden problem, we are still under threat. Our work is to focus on such problem and look into several related paper and do a study on this topic about e-mail security.

1.1.1 SMTP

SMTP is the standard protocol for email services on a TCP/IP network.it provides the ability to send and receive email messages. However, SMTP lacks authentication. In practice, attackers usually exploit SMTP by running their own email servers or clients. SMTP's design includes multiple "identities" when handling messages.it also introduces multiple other sender identities, thus leading to the question: which identity to authenticate.

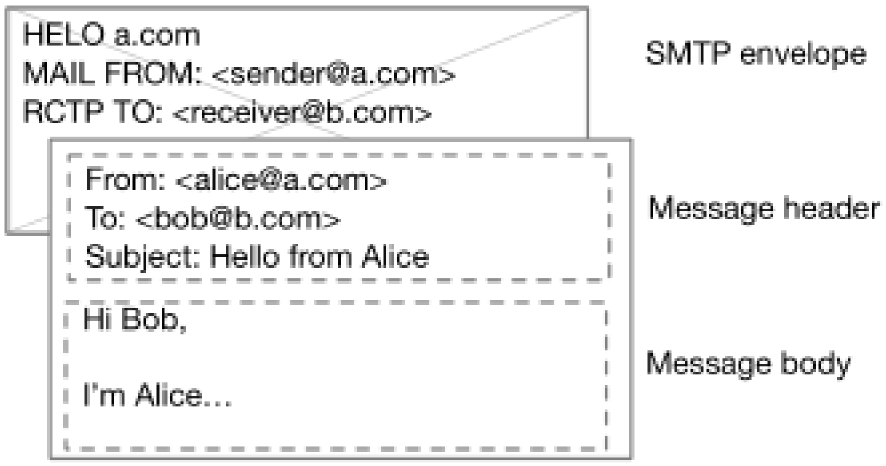


figure 1.1: An example of an SMTP message

1.1.2 SPF/DKIM/DMARC

To combat email forgery, various email authentication mechanisms have been developed, including SPF, DKIM, and DMARC.

Sender Policy Framework (SPF) allows a domain owner to publish DNS records to specify which servers are allowed to send emails for the domain. A major problem of SPF is incompatibility with mail forwarders.

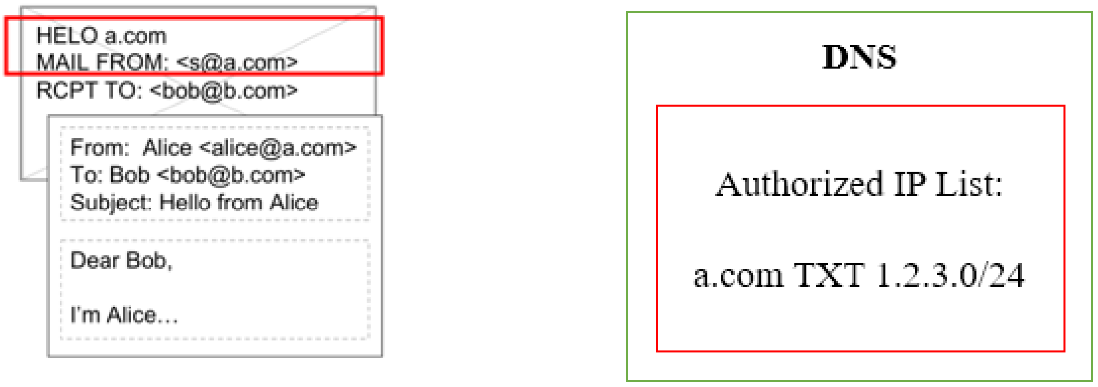


figure 1.2: An example of SPF work process

Domain Keys Identified Mail (DKIM) uses cryptography to authenticate senders and protect email integrity. The general idea behind DKIM is to let senders sign parts of messages so that receivers can validate them.

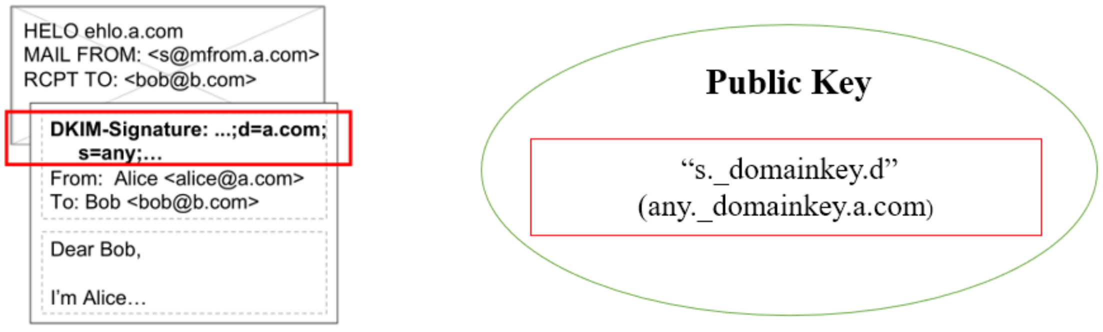


figure 1.3: An example of DKIM work process

However,neither SPF nor DKIM provides a complete solution for preventing email spoofing. Neither of them authenticates the From header displayed to the end-user, so that the FROM address of emails can still be forged.

Domain-based Message Authentication, Reporting & Conformance (DMARC) is designed to fix the final trust problem by building on SPF and DKIM. The target is to ensure that the address in the FROM header cannot be forged and prevents email forgery.

1.2 Threat Model

We assume three types of attackers who would be able to exploit the vulnerability of mail server:

Forgery attacker. They may spoof the email's sender in the From header and craft a fake message to attack the victim's mailbox.

Replay attacker. They may use the valid SPF record and DKIM signatures to bypass the security system and exploit modifications to email header and send the fake email.

Malicious mailbox owner. They may own a legitimate email address to pass all the authentication. However they also have the ability to cheat the security system and exploit the failure of email providers to spoof the email header.

Based on the previous types of attackers, various of attacking methods could be raised to cheat the SPF/DKIM/DMARC security system.

2. Discussion on Paper Work

We choose the paper *Composition Kills: A Case Study of Email Sender Authentication* as the main reference. In this section, we'll do the basic study on this awarded paper.

2.1 Researching method

To investigate how real-world systems handle these challenges, the research team conducted a security analysis of popular email providers and MUAs. In total, the team tested 10 email providers and 19 MUAs, including 9 local email clients and 10 web interfaces. The team defined an email authentication mechanism as broken when the following both hold: 1) the email server erroneously verifies the test email's sender as not spoofed, for example, DMARC authentication produces a "pass" result; 2) the MUA erroneously indicates that the sender address is from a (legitimate) target domain rather than the attacker's sending domain.

Next, we will discover the way to attack the email system introduced in the paper, and show the test results of popular email server and MUAs facing many kinds of attacking techniques. Each technique is marked by letters such as A1 or B1.

2.2 Attacking Methods Raised by Paper

2.2.1 Intra-server attacks

Intra-server attacks exploit inconsistencies between different internal components of a single implementation. The team discovered three techniques to exploit their inconsistencies: (1) HELO/MAIL FROM confusion (A1, A2); (2) ambiguous domains (A3); and (3) authentication results injection (A4, A5).

HELO/MAIL FROM confusion (A1, A2). SMTP employs two different identifiers—HELO and MAIL FROM—to represent the email sender who transmits a message. And when DMARK and SPF work, SPF is supposed to check the both, but actually HELO is recommended instead of enforced, while DMARK use MAIL FROM identities to perform alignment test. This design introduces the possibility that different components might authenticate different identifiers.

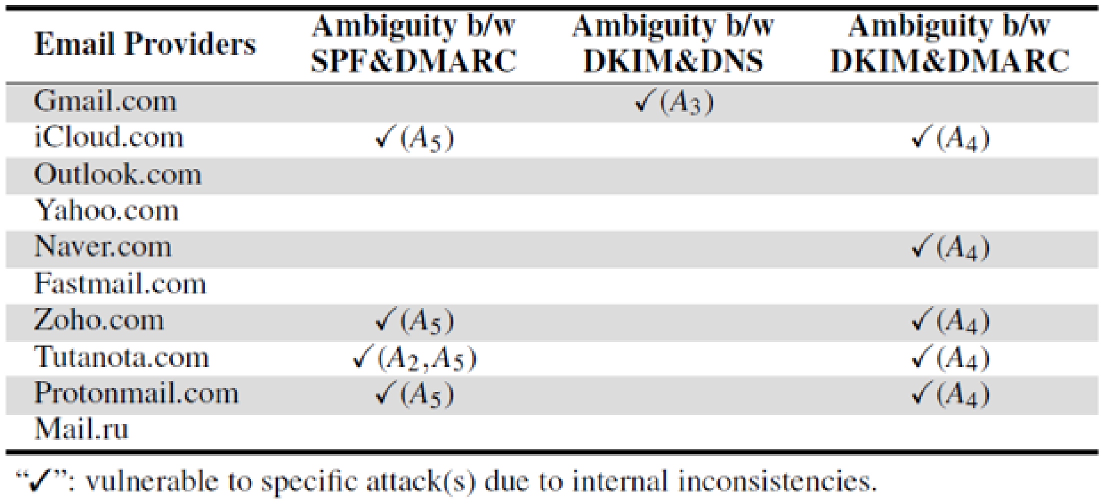
Non-existent subdomains (A1). The first technique crafts a MAIL FROM domain as a non-existent subdomain of a legitimate domain. That means the domain in MAIL FROM is not exist at all, SPF can only test the HELO, which is under the control of attackers, and forwards a "pass" result. However, still use the MAIL FROM domain to perform the alignment test with the FROM header.

"Empty" MAIL FROM addresses (A2). Some SPF implementations treat "(any@legitimate.com" as an empty MAIL FROM address, and leads to the same result as A1.

Ambiguous domains (A3), NUL ambiguity (A3). DKIM and DNS components differ in interpreting NULs in domain name.

Authentication results injection. To explain this attack technique, firstly we will simply introduce authentication result header syntax which is passed by SPF and DKIM to DMARK to show the verification result. In the authentication result header, "smtp.mailfrom" represents the domain verified by the SPF component, and "header.d" represents the domain verified by the DKIM component. A vulnerability arises because attackers can modify the domain name embedded in the "smtp.mailfrom" and "header.d". Particularly, attackers can introduce malformed domains including meta-characters, which will be treated as data by SPF and DKIM, and be parsed as control information by DMARC. The authentication results injection can be divided into DKIM authentication results injection (A4) and SPF authentication results injection (A5) which is similar.

The paper issues the test results of their work. It shows that many email providers are proofed to have such vulnerability to Intra-server attacks up to the publication date.



Email Providers	Ambiguity b/w SPF&DMARC	Ambiguity b/w DKIM&DNS	Ambiguity b/w DKIM&DMARC
Gmail.com		✓(A_3)	
iCloud.com	✓(A_5)		✓(A_4)
Outlook.com			
Yahoo.com			
Naver.com			✓(A_4)
Fastmail.com			
Zoho.com	✓(A_5)		✓(A_4)
Tutanota.com	✓(A_2,A_5)		✓(A_4)
Protonmail.com	✓(A_5)		✓(A_4)
Mail.ru			

"✓": vulnerable to specific attack(s) due to internal inconsistencies.

figure 2.1: The test on main mailboxes provided by the paper author.

2.2.2 UI-mismatch

Generally, UI-mismatch attacks exploit the inconsistencies between how an email server validates a message versus how the MUA ultimately indicates its validity. We divide the UI-mismatch attacks into two categories: ambiguous FROM headers and ambiguous email addresses.

Ambiguous FROM headers exploits the inconsistencies on the display and authenticate rules on FROM header of a certain email.

Multiple From headers (A6). When an email has multiple FROM headers, the email massage should be rejected by receiving services. For example, some of the MUAs show last FROM header while others use the first one.

Space-surrounded From headers (A7). The attackers violate the email header syntax by inserting whitespace around the FROM headers which will be handled by email servers in different way. First, use of whitespace can bypass the email server's validation. Second, inconsistent interpretation of whitespace can lead to ambiguities.

From alternative headers (A8). If an attacker crafts an email with no From header or an unrecognized From header, some implementations will use alternative headers to identify the message sender.

Next we will discuss **ambiguous email addresses**. A valid From header consists of four elements, with a single mailbox address. The four elements are Display name, Real address, Route portion and comment. Attackers can exploit the complexity of FROM header syntax.

Attacks leveraging complex From headers. The team find that implementations vary in parsing and interpreting From headers. Here we show five attacks that exploit these inconsistencies.

Multiple email addresses (A9). The team observe 5 distinct behaviors in processing From headers listing multiple addresses. Gmail.com (Server) and Mail.ru (Server) reject the messages; Tutanota.com (Web) displays the last address; Zoho.com (Server) and iCloud.com (Web) don't verify or display any address; 2 mail servers and 4 MUAs verify or display all of the addresses; all
the others take the first address.

Email address encoding (A10). In the experiments, Yahoo.com (Server), Outlook.com (Server), iCloud.com (Server), Fastmail (Server), Zoho.com (Server) and Tutanota.com (Server) don't recognize the encoded address, and use attack.com for DMARC testing; but Gmail.com (Web), Outlook.com (Web), Yahoo.com (Web), Naver.com (Web), Mail (MacOS), Mail (Windows) and Mail (iOS) support this encoding feature, and only display the first address.

Route portion (A11). Fastmail.com (Server) does not recognize the route portion, and treats attack.com as a real address to use for DMARC verification; while 10

MUAs, including Fastmail.com (Web), ignore the route portion, and only show admin@legitimate.com.

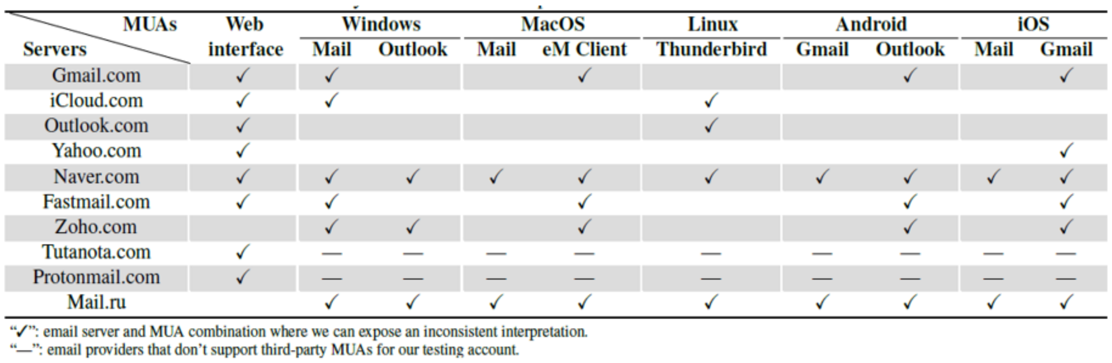
Quoted-pairs (A12). Gmail.com (Server) and iCloud.com (Web) recognize the second address; but Mail (Windows), iCloud (Server) and 12 other implementations only use the first one.

Parsing inconsistencies (A13). Thunderbird (Linux), Mail.ru (Web), Gmail.com (Server) and Mail.ru (Server) mistakenly validate or display admin@legitimate.com as the real sender but Outlook.com (Server), iCloud.com (Server), Protonmail.com (Server) and 9 other implementations recognize it as the wrong one.

SPF, DKIM, and DMARC rely on domain queries for sender authentication. When failing to obtain the domain record, the mail service providers may decide that the domain doesn't deploy the corresponding security mechanisms, and allow the message into the user's inbox.

The attackers can use *Invisible characters* (B1), *Encoding* (B2), *From alternative headers* (B3) to exploit the vulnerability.

The test results in the paper implies a more serious and general problem on most mailboxes.



MUAs / Servers	Web	Windows		MacOS		Linux	Android		iOS	
	interface	Mail	Outlook	Mail	eM Client	Thunderbird	Gmail	Outlook	Mail	Gmail
Gmail.com	✓	✓			✓			✓		✓
iCloud.com	✓	✓				✓				
Outlook.com	✓					✓				
Yahoo.com	✓									✓
Naver.com	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Fastmail.com	✓	✓			✓			✓		✓
Zoho.com		✓	✓		✓			✓		✓
Tutanota.com	✓	—	—	—	—	—	—	—	—	—
Protonmail.com	✓	—	—	—	—	—	—	—	—	—
Mail.ru		✓	✓	✓	✓	✓	✓	✓	✓	✓

"✓": email server and MUA combination where we can expose an inconsistent interpretation.
"—": email providers that don't support third-party MUAs for our testing account.

figure 2.2: The test on main mailboxes focuses on the UI-mismatch attacks provided by the paper author

2.2.3 Ambiguous-replay Attacks

This is similar to the replay attack, the attackers spoof emails with seemingly valid DKIM signatures from legitimate domains. Two DKIM mechanisms make signature spoofing possible. First, DKIM doesn't prevent replay attacks. Second, DKIM allows attackers to append additional email headers— or even body contents, in some cases—to the original message.

Next, we will discuss DKIM signature replay attacks. DKIM signatures protect both email headers and bodies. The latter is always signed. Signing headers, however, is optional, and specified by the "h=" tag of the DKIM-Signature header.

Header spoofing (A14 and A15). Among the various identities, only the From header must be signed. A replay attacker can modify these unprotected fields in signed messages without invalidating DKIM signatures. While including all necessary headers in the signature can prevent attackers from tampering them, an attacker can craft ambiguous emails by adding a new header (e.g., Subject) to the signed mail, if two parties in the email process chain parse and interpret the extra header differently.

Body spoofing (A16). The attack technique can modify the email body without destroying the DKIM signature. The DKIM-Signature header has the option "l="tag which indicates the length of the email body included in the signature. By modify the tag and email body, attackers can change the content of email without being detected by email servers.

The problem of *spoofing via an email account* (A17) arises when an email provider does not perform sufficient checks on the From header, enabling an attacker to send a signed message with another user's address (e.g., administrator). As the message has the email provider's DKIM signature attached, it will pass the receiver's DKIM and DMARC validation.

The spoofing attack (A18). First the attackers craft normal email to the email servers to get the DKIM signature and reuse the signature by adding an extra From header with another user's address to the DKIM-signed message and resends it to a victim.

2.3 More Discussion

The team mainly discuss about the vulnerability arise from the inconsistence between different parts of authentication. The exploration pay attention to the mechanism with defective and unreasonable design. So, we can also pay close attention to the users' manner and using habit. Because with a large number of users, millions of mail transfer may present specific behaviors which may lead to the insecurity. Based on the big data analysis, we may find other vulnerabilities which could be exploited by the attackers.

3. Spoofer Demo

After looking into the paper and understand the learning method, we constructed an email spoofer demo to illustrated our research results. The demo has two core part. One is the SMTP-based email sender and the other is the spoofing parameters. By setting the spoofing parameters, it is possible to send a sender-spoofed email.

The demo is written in Python. Several open-sourced library are used, such as *dns*, *pydkim* and *socket*.

One matter to be issued: All the spoofing cases are tested based on personal domain and personal email address. Besides, the email systems are all strengthened, so it cannot reach the target that achieve a real spoofing. The test is a way to proof that such technique is able to crack SPF/DKIM/DMARC authentication.

3.1 SMTP Framework

The implementation of SMTP is based on TCP and socket. The logic could be simplified as the figure below.

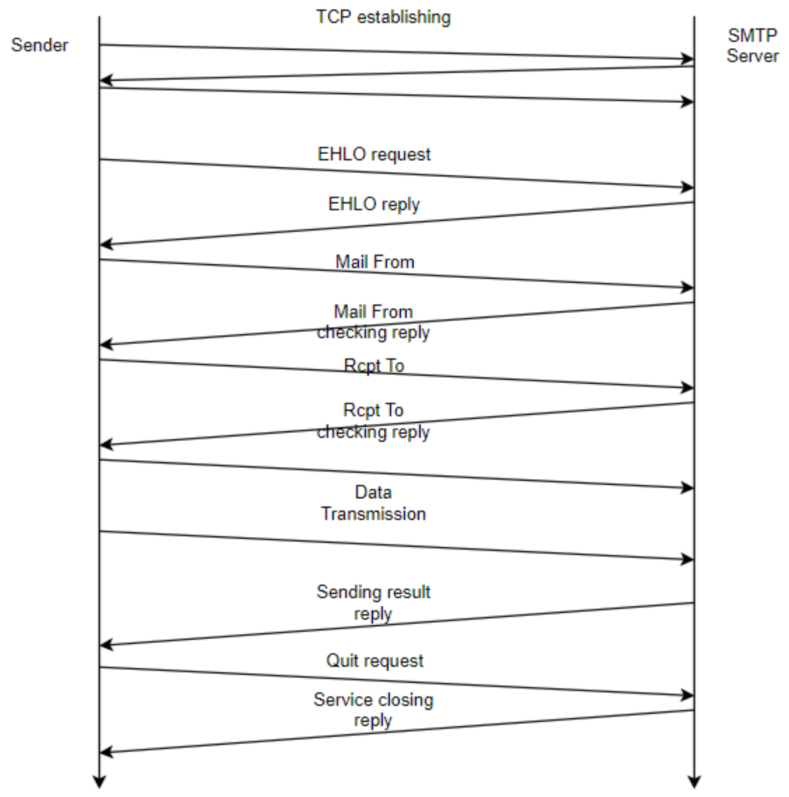


figure 3.1: SMTP workflow

The first step is common TCP three-way handshake. After the connection was established between sender and the mail server, the sender will send a EHLO request packet to ask for the server response. EHLO is an extended version of HELO, which supports user authentication. EHLO request includes the login information. A typical EHLO section can be listed as follow:

```
<<< ehlo 126.com  // inform the server to do the identity authenticatio
n
>>> 250-mail
      250-PIPELINING
      250-AUTH LOGIN PLAIN
      250-AUTH=LOGIN PLAIN
      250-coremail  1Uxr2xKj7kG0xkI17xGrU7I0s8FY2U3Uj8Cz28x1UUUUU7Ic2I0
Y2UrTiPGSUCa0xDrUUUUj
      250-STARTTLS
      250 8BITMIME

<<< AUTH LOGIN aml*****29t          // start identifying, sending the u
ser mail address with base64 encoding
>>> 334 UGFzc3dvcmQ6                // asking for password

<<< ZW**********Mz                        // the password with base64 e
ncoding
>>> 235 Authentication successful
```

After the identity authentication, SMTP server will check the MAIL FROM header and the RCPT TO header. This will be the last security defense of SMTP.

3.2 Key Techniques

3.2.1 Domain Setting

In order to perform well both on authenticated email address and the invalid domain (which refers to the fake email address), we used a personal domain name *ruin.net.cn* to deploy several configuration.

Considering the security of email system, most of the servers shut down the 25 port which is the default forwarding port of SMTP. Thus, this domain cannot be set as an email sender. However, by changing the parameter of the email message and configuring the DNS records, a message with invalid domain name could be sent to the victim email.

3.2.2 SPF & DKIM Signature Configuration

By setting SPF record and DKIM signature to a personal domain, the certain domain would be able to pass the SPF and DKIM checking. Generally, simple DNS resolver could achieve this goal.

An SPF record can be noted as follow:

```
@ TXT v=spf1 a mx ip4:222.205.46.0/24 -all
```

A DKIM signature record can be noted as follow:

```
selector._domainkey TXT v=DKIM1; k=rsa; p=MIGfMA0GCSqGSIb3DQEBAQUAA4GNA
DCBiQKBgQCMK4X+VibGhNWBGmxwvNgTKqyyLHuOWlomLo0vG9nhy97w8BLUz6fD6yLYdVhm
rWW8hy8j0rcPxr40+L6P9yZ5yDpYwTwcarEky8oj4Jb2XlqWE9OcYgIOjpoN5d2xGqJau+S
0bmeuneuMRPh4cx/Uo1qEovsxnMAAiKPgmze6nwIDAQAB
```

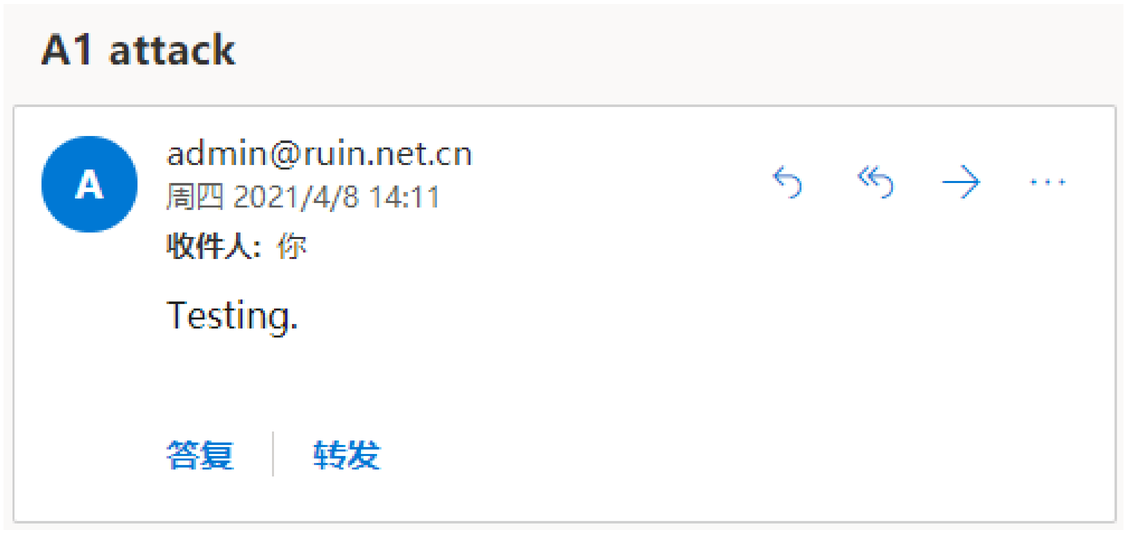
The p domain in DKIM record refers to the public key. The private key is attached to the message while sending to the SMTP receiver.

3.3 Typical Spoofing Methods

Since most of the mail service providers have already strengthened their guarding system, it's hard to recover the results presented by the author. The following tests are performed on a personal Outlook mail address, and all tests only proof that such a spoofing method may be effective to pass the DMARC checking.

3.3.1 A1 Attack: Fake subdomain invasion

The case crafts a fake MAIL FROM header with non-existent subdomain to cheat the receiver. Since some SPF implementations only check the HELO header, the MAIL FROM header is able to be a spoofing point. As long as the MAIL FROM address is not empty, OpenDMARC authentication will return a positive result because of the pass of SPF checking.

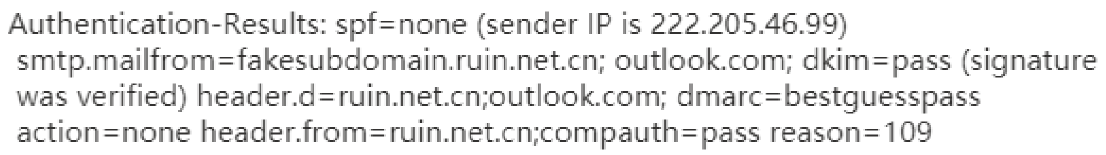


A1 attack

admin@ruin.net.cn
周四 2021/4/8 14:11
收件人: 你

Testing.

figure 3.2: A1 Attack mail



```
Authentication-Results: spf=none (sender IP is 222.205.46.99)
 smtp.mailfrom=fakesubdomain.ruin.net.cn; outlook.com; dkim=pass (signature
 was verified) header.d=ruin.net.cn;outlook.com; dmarc=bestguesspass
 action=none header.from=ruin.net.cn;compauth=pass reason=109
```

figure 3.3: A1 Attack mail check

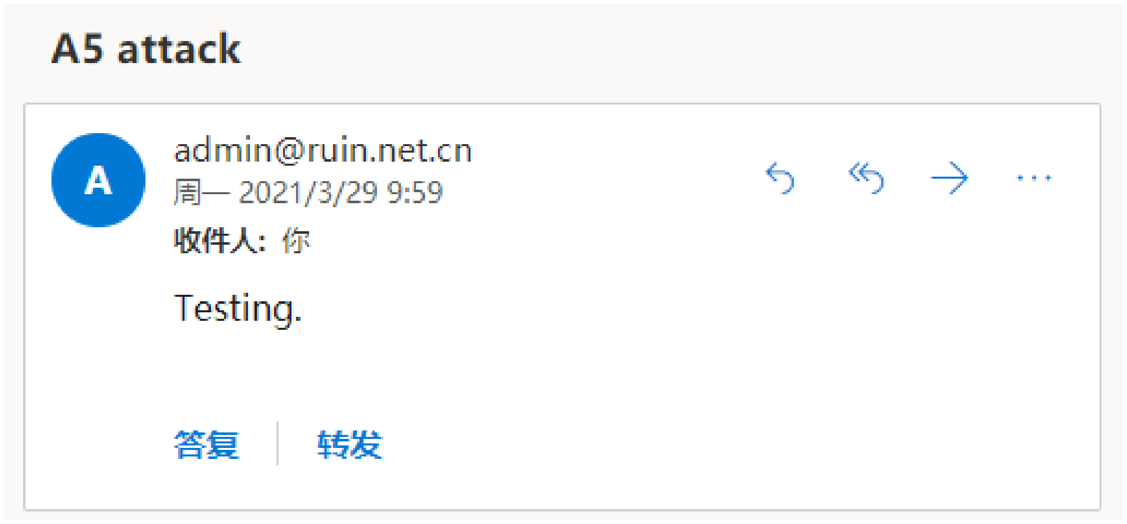
In our test, the fake subdomain can still pass the verify of DKIM test while the SPF check shows *none* because of the MAIL FROM is set to a fake subdomain name *fakedoain.ruin.net.cn*. However the DKIM check still go through the genuine domain name *ruin.net.cn*.

The DMARC check result is bestguesspass for there's no DMARC result setting at *outlook.com*. While no error reported from both SPF and DKIM, it gave a reasonable infer that the mail could pass the DMARC testing.

3.3.2 A5 Attack: The variation of authentication results injection

This case is a variation of the A5 Attack in the paper. This case exploit the vulnerability of the attacker's accessibility and the character's non-consistent. In particular, an attacker can introduce malformed domains that include meta-characters. Different secure protocol and mail servers may treat those characters differently.

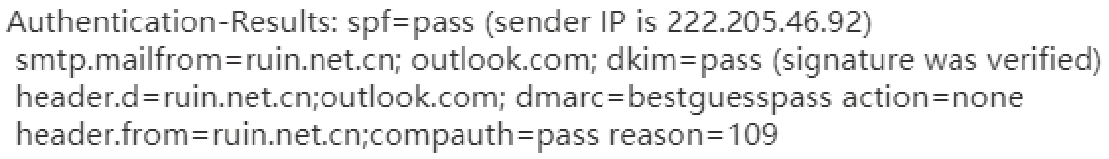
In this spoofing case, the address in MAIL FROM is routed to another address. For example, a MAIL FROM header <*@legitimate.com,@any.com:'any@attack.com*> will lead to an inconsistence display in some mail server. Take *outlook.com* as an example, the receiver will be shown as *any@legitimate.com*, while the spoofed email will pass the security by the domain name *attack.com*.



admin@ruin.net.cn
周一 2021/3/29 9:59
收件人: 你

Testing.

figure 3.4: A5 Attack mail



```
Authentication-Results: spf=pass (sender IP is 222.205.46.92)
 smtp.mailfrom=ruin.net.cn; outlook.com; dkim=pass (signature was verified)
 header.d=ruin.net.cn;outlook.com; dmarc=bestguesspass action=none
 header.from=ruin.net.cn;compauth=pass reason=109
```

figure 3.5: A5 Attack mail check

The result shows that such a spoofed email with a fake MAIL FROM header will pass both SPF and DKIM. The verify system check only the legitimate domain, while the showing result is the attack site.

3.3.3 A10 Attack: Email address encoding

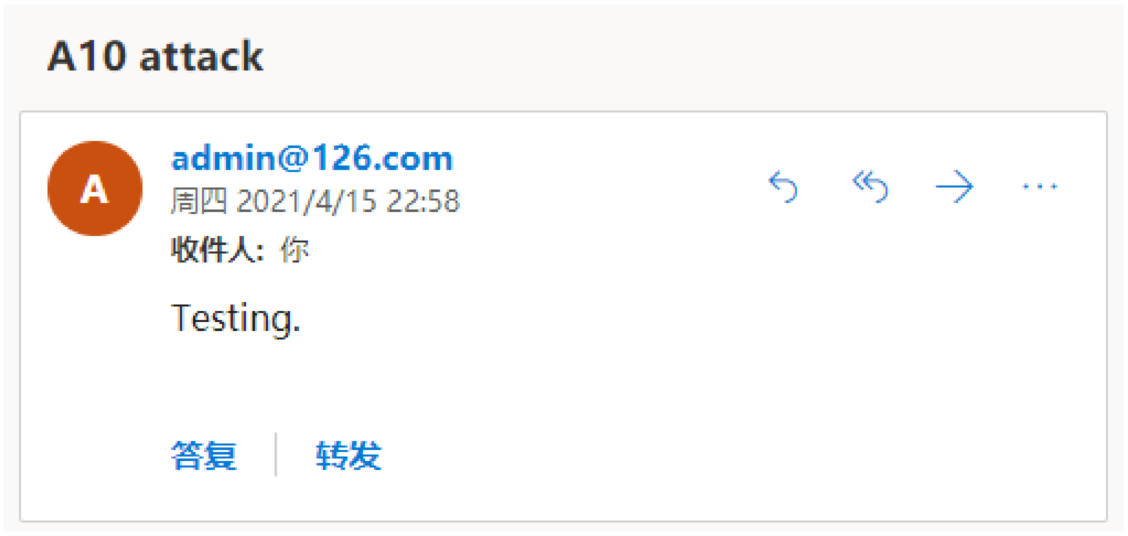
This case exploits the difference of coding rules between different part. The basic coding rule of a SMTP email is US-ASCII characters. To support non-ASCII characters, RFC 2047 defined two encoding approaches: Base64 encoding and quoted-printable encoding with following syntax:

```
=?charset?encoding?encoded-text?=
```

Thus a hidden From header can be added to the mail, for example:

```
=?utf-8?B? +base64.b64encode(b"admin@126.com")+ ?=
```

Inside which is a base64 encoding of an email address, which will be displayed on MUA.

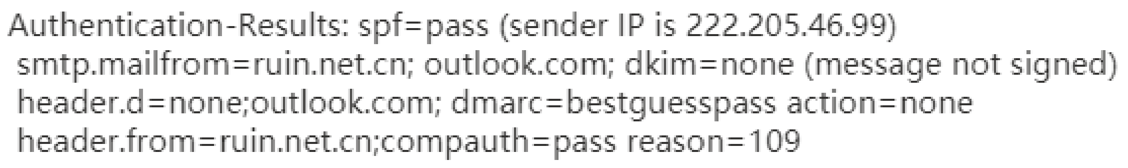


A10 attack

admin@126.com
周四 2021/4/15 22:58
收件人: 你

Testing.

figure 3.6: A10 Attack mail



```
Authentication-Results: spf=pass (sender IP is 222.205.46.99)
 smtp.mailfrom=ruin.net.cn; outlook.com; dkim=none (message not signed)
 header.d=none;outlook.com; dmarc=bestguesspass action=none
 header.from=ruin.net.cn;compauth=pass reason=109
```

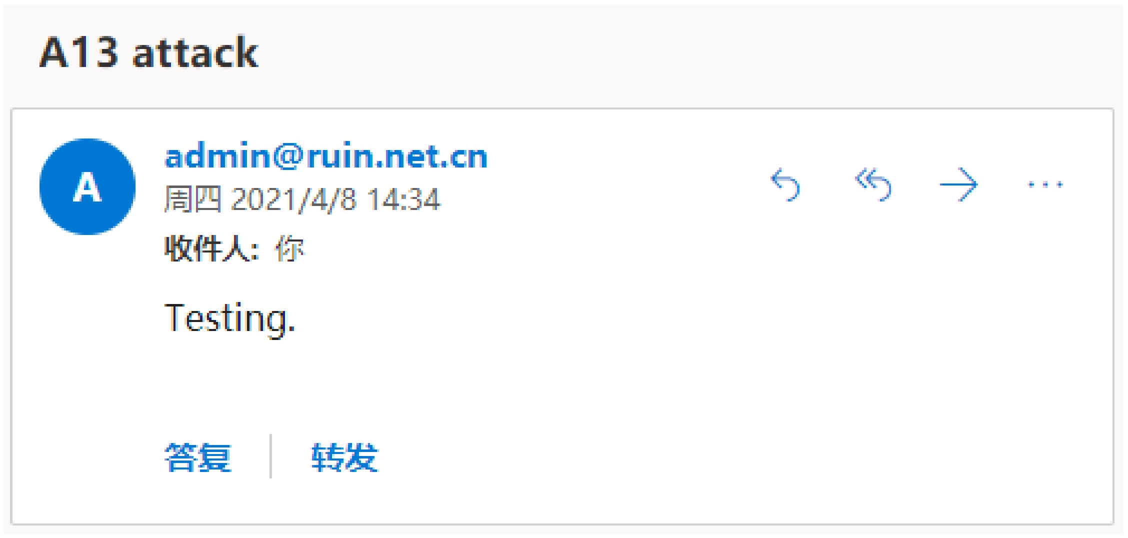
figure 3.7: A10 Attack mail check

With this attacking method, we can even spoof as another domain. The SPF will go pass the sender address, which is outside the base64 encoding. While the displayed one is the fake spoofed sender address which encoded by base64.

3.3.4 A13 Attack: Parsing inconsistencies

This is one of the methods that leverages complex From headers. In paper, the author came up with various of spoofing methods including using multiple email address, encoding the email address and so on.

We implemented the spoofing method which exploits the parsing inconsistencies. This is based on the inconsistencies in recognizing the precedence of different delimiters. For some email server, DMARC component will take '<' character for higher priority and go through the authentication process while MUA will have a different preference and display the other one.


A13 attack

admin@ruin.net.cn
周四 2021/4/8 14:34
收件人: 你

Testing.

figure 3.8: A13 Attack mail

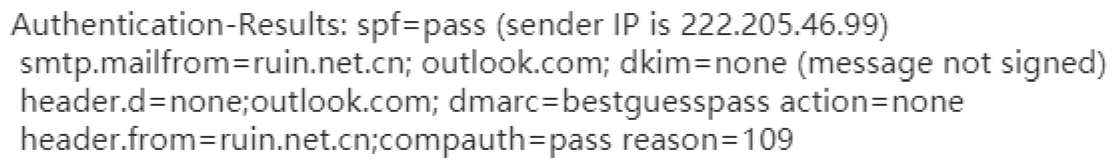

```
Authentication-Results: spf=pass (sender IP is 222.205.46.99)
 smtp.mailfrom=ruin.net.cn; outlook.com; dkim=none (message not signed)
 header.d=none;outlook.com; dmarc=bestguesspass action=none
 header.from=ruin.net.cn;compauth=pass reason=109
```

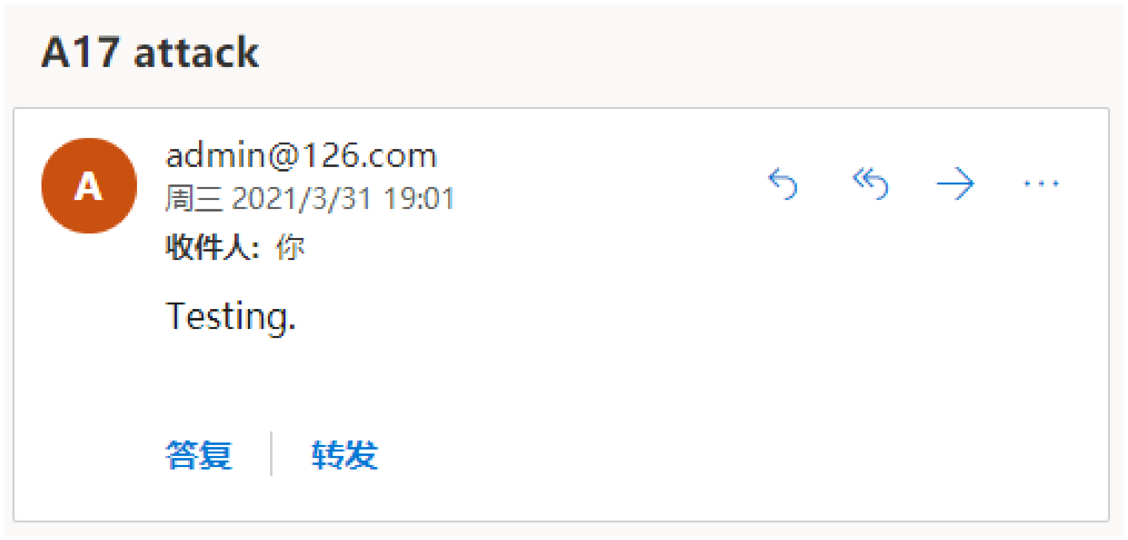
figure 3.9: A13 Attack mail check

The testing result shows that the SPF and DMARC all go through the spoofed email address.

3.3.5 A17 Attack: Spoofing via an email service account

Among all of the test cases, the most reliable attack methods is to spoof via an authenticated account. In this way, the only goal is to forge the sender address.

The most effective way is to forge into another email address of the same email server of the attacker's account.



A17 attack

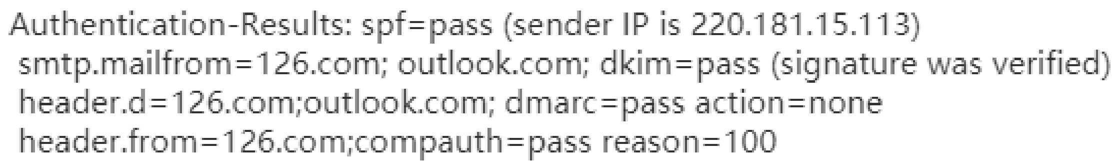
admin@126.com

周三 2021/3/31 19:01

收件人: 你

Testing.

figure 3.10: A17 Attack mail



```
Authentication-Results: spf=pass (sender IP is 220.181.15.113)
 smtp.mailfrom=126.com; outlook.com; dkim=pass (signature was verified)
 header.d=126.com;outlook.com; dmarc=pass action=none
 header.from=126.com;compauth=pass reason=100
```

figure 3.11: A17 Attack mail check

Our test case uses a personal email account in *126.com* and the goal is the result is to cheat the MUA and only display the spoofed email address *admin@126.com*. The email provider does not perform sufficient checks on the From header, enabling an attacker to send a signed message with another user's address. As the message has the email provider's DKIM signature attached, it will pass the receiver's DKIM and DMARC validation.

4. Further Study

4.1 Basic Conclusion

The protocol allows the sender to send a mail with multi-identifier. Although we can manager it with strict definition, implementation of SMTP always cant's fully understand the complicated definition.

Complicated syntax rules based on text leads to the inconsistency between different components and the ambiguity of parsing process. Allowing various types of white space and notes makes the possible positions variable. But different implementations would not obey the rules totally, for the cause of robustness, which makes the situation even worse.

Furthermore, there are too many components in SMTP system, between which two would cause security problems.

4.2 Comprehensive Solutions to Avoid Spoofing

Reject the mails which have ambiguous address problems. But in the real life, many mails are not perfectly right. So we can't do it in the receiving server and receiving MUA. ⇒ We need to begin it in the sending MUA.

Deploy IMAP extension to pass the authentication result to the MUA, instead of passing the result in the "Authentication-Results" header. This solution can also shield the Authentication-Result header to be spoofed.

Add the verify result of identity. To solve "multi-identifier" problems, apart from adding "xxx: pass" to the Authentication-Result, we can also add"xxx*identifier: xxx" to the result. In this way, we can unify the identifiers used in different components. And thus we can decrease the inconsistency in a large scale Authentication-Result header is safe enough We can use Comprehensive*solutions_2 to help get it solved.

Collect suspicious information. For mails that we have found to be abnormal, though we can accept it by courtesy of robustness, it is important to collect the information so that the MUA can display it to the user.

Set detailed configuration of the MUA. To improve the complicated syntax rules which make much contribution to the difficulty of the interacting between different implementation, the service providers need to keep a configuration message, which signify the parts of the "FROM" rules, like white space、portion、notes.

Improve the rules of "FROM" headers. We don't chase restricting the input totally. Instead, we try to reduce the freedom of it. For example, we can request the header to be regular-language、context-free-language… to the lower level in the Chomsky infrastructure. We impose a same parser to both the receiving server and the MUA. In this way, we can extinguish the inconsistency in the parsing procedure. Besides, simply set some restrictions to part of the special characters, which lead to most bugs, like "(".

Realizing the inconsistency between the receiving server and the MUA. The service providers can eliminate the parsing & verifying procedure of the MUA. The receiving server pass the authentication result . This result will not be parsed/verified again. ⇒ RFC standards have considered this solution. But it is not defined well, which can still be spoofed. ⇒ choose some new transfer protocol like IMAP/POP.

4.3 Current Defenses

Aims at the threat of spam spreading into the enterprise network and causing great harm. A large number of potential malicious spam are collected, and the main distinguishing features of dangerous spam are analyzed. The common supervised machine learning classifiers are used for learning test and optimization. Finally, the best performance model is obtained: Support Vector Machine & random forest classifiers. The recall rate is 91.6% and the accuracy rate is 95.2%

Dynamic blacklist based on IP/domain/HELO/FROM. Blacklist is widely used in many places. It protect the user from too many malicious mails. In the practice, it is always combined with reporting. If your mails are reported to be dangerous by several people in a short time, all mails from you may be rejected, even there is spoofing.

Over-sign technique. In a common case, there is only 1 header in the mail, so we will only check 1 header. But under the risk of spoofing with multi-header, we need to change the protocol. We make digital signature on all the possible headers, even on nonexistent headers. So any changes will fail.

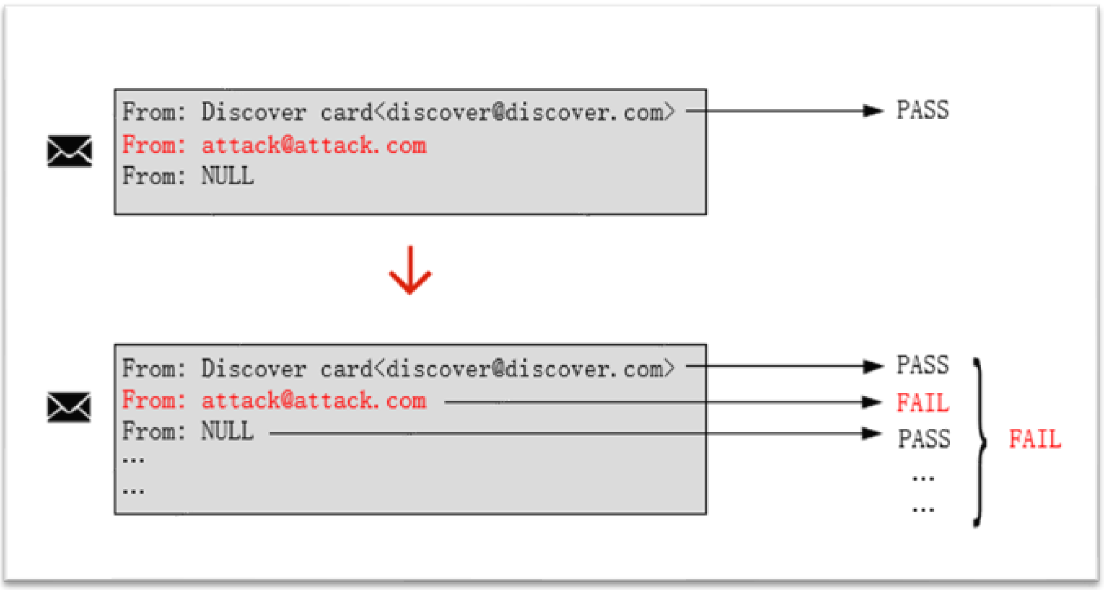


figure 4.1: An example of Over-sign technique working principle.

IMAP/POP extensions. In the past, the receiving server will parse and verify the mail, add the result to the mail and send it to the user. And the inconsistency between the receiving server and the MUA, is one of the major reasons of risks. After adopting the IMAP/POP extensions, your computers are really a screen for displaying. You don't need to do the verifying for the second time.

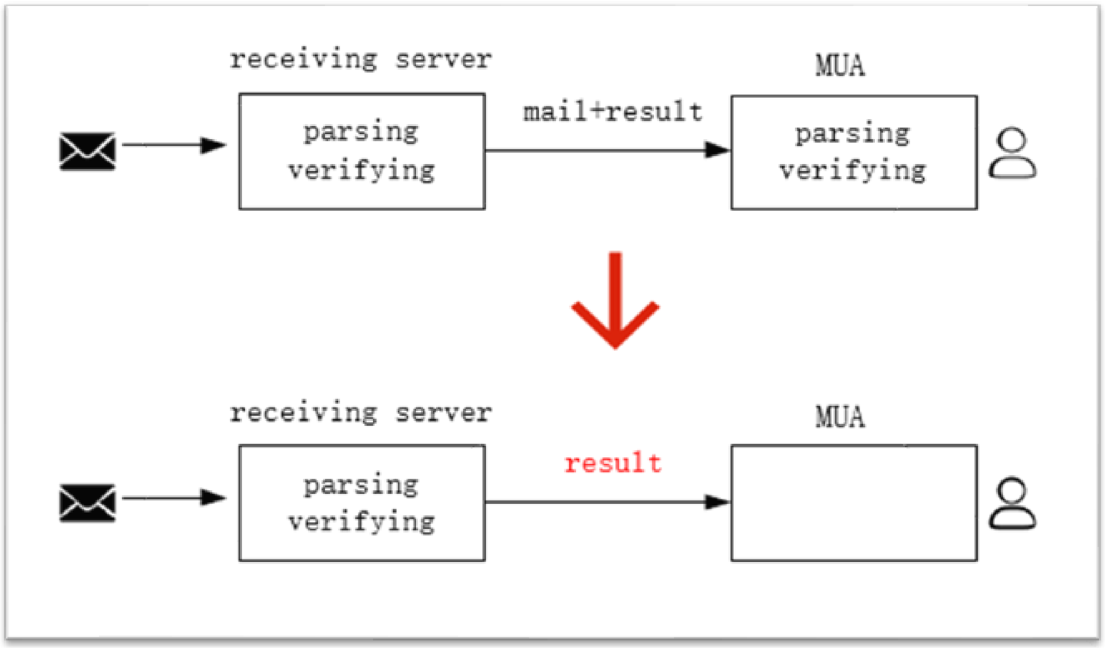


figure 4.2: An example of IMAP/POP extension working principle.

ARC protocol. Authentication Received chain. In this protocol, a mail is not allowed to be changed in the transit. Just like what the picture shows, if a Mail from ZJU get changed in the Yahoo, the user will search for the chain. If the chain is trustworthy, the user will accept the mail.

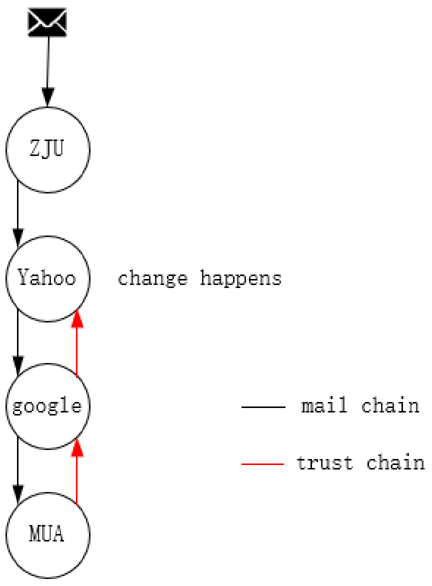


figure 4.3: An example of ARC mail chain and trust chain

4.4 Other Possible Attacks and Solutions

Homograph Impersonation: IDN (international domain name) allows people from all over the world to visit websites in their own national languages. In the field of Web defense, there has always been a way of attacking websites by means of profiling. Characters in many languages may look very similar. The classic one is the official website of apple. A can be replaced by characters in other languages, All the headers that may be presented in the address bar -- from header, may be disguised as characters of other languages. This attack takes advantage of the homonym of human languages, so it is difficult to solve fundamentally

Solutions:

There are similarities and differences between e-mail system and web system. For example, both of them are domain based processing methods, and need to interact with DNS server. Therefore, we can learn from the browser's solution and experience of IDN problem, and build similar filter verification components on MUA. However, we need to pay attention to that web system is often divided into mainstream websites.

However, in the SMTP system, the mailbox name is no mainstream or substream, and the use frequency of individual users may be lower than that of attackers, so it can not be copied. It needs to be combined with the characteristics of the mail system.

Appendix 1: Detail Solutions to Different Attacks

A1+A2 HELO/MAIL_FROM confusion

1. We should regard the SPF result to be "fail" if we can't get the DNS-record for SPF policy.
2. All DMARC implementation should be strict, in which way we can prevent bypassing through non-existent sub-domain.
3. Use unified identifier in SPF and DMARC process. For example, we impose the SPF to verify "HELO" and pass"SPF_identifier: HELO" to the DMARC. And then DMARC can use the same identifier for verifying.
4. Filter the characters in the MAIL_FROM header which may lead to ambiguity like "("

A3 ambiguous domains

1. DMARC not only verify the "d=" label from the DKIM, but also verify the raw MAIL_FROM, and request these two to be unified.

2. Upgrade the DNS server to avoid the “\x00”-cheating. This may lead to incremental deployment problems and cost a lot.

A4+A5 SPF/DKIM injection

1. We call for help from the DNS server just like what we do in A3.2

2. Recognize the special characters that the DNS server may not support before SPF/DKIM pass the content to the DNS server. And tell the exception to the Authentication-Result.

3. Impose a same parser in SPF、DKIM and DMARC

4. A3.1

A6 multiple FROM header

1. Obey the RFC5322 standard and accept only one FROM header.

A7 space-surrounded FORM header

1. Improve the complicated syntax rules of FROM header, and in this way, we can solve the problem while we don’t loose much robustness.

A8 FROM alternative header

1. When we look forward to other headers as substitutes of FROM header, we should notify the user instead of simply putting them in the address.

2. Comprehensive_solutions.2: Passing the identifiers used in the receiving server.

A9 Multiple email address

1. Receiving server don’t insert the Sender to the FROM header, since the sender maybe not the real sender. The insertion may lead to the FROM header’s inconsistency between DMARC and MUA.

A10 Email address encoding

1. Since the problems arise from “Servers can’t distinguish non-ASCII-encoded From header”, we should improve the syntax library so the server can support them.

2. Tell the MUA that the server don’t support other encoding methods, and impose the MUA to use the same encoding methods. ⇒ This way may lead to some decrease in the robustness

A11 Route portion

1. Sending server begin to unable route-portion function. And then the MUA should follow this action.

2. Comprehensive solutions 5

A12 Quoted-pairs

1. Comprehensive solutions 5

A13 Parsing inconsistency

1. Comprehensive solutions 6.2

B1 Invisible characters

1. Comprehensive solutions 4
2. Instead of using open-failure, we use open-reject. Failure of DNS request of SPF、DKIM、DMARC at the same time is impossible in the theory, and at that time ,we should reject the mail. The danger of the mail overwhelms the robustness principle.
3. For domain datas that can't be shown as characters like \u2000, we can display it in the address as byte-string, and then notify the user.

B2 Encoding

1. Comprehensive solutions 4
2. Instead of using open-failure, we use open-reject. Failure of DNS request of SPF、DKIM、DMARC at the same time is impossible in the theory, and at that time ,we should reject the mail. The danger of the mail overwhelms the robustness principle.
3. For domain data that can't be shown as characters like \u2000, we can display it in the address as byte-string, and then notify the user.

B3 From alternative header

1. When we find the FROM header to be null, we should just show it to be null, instead of showing another header as a substitute. And we need to notify the user about the possible senders.
2. Just reject the mail, because FROM header is a very basic header which is so important that forgetting it seems to be impossible.

A14+A15+A16 Header spoofing

1. Get all the headers into the protection of DKIM signature.
2. By Comprehensive solutions 7, we can solve the inconsistency problems between receiving server and the MUA. And based on this, we can still enjoy the robustness brought by the "I=" flag.

combining (1) and (2) is reasonable and useful.

A17 spoofing via an account

1. Combine comprehensive solutions 1,4,5,6. We can mitigate the incremental deployment problem while decrease the complexity of FROM header.

2. Mail service providers always deploy username/password authentication mechanism incompatible with the complex FROM header syntax. So there is no need to verify the previous one based on the latter one. We can extract the message from the username, if the From header is not so standard, we can replace the FROM header with the username, since the previous one can be more safe by courtesy of the service providers.

3. This paper proposes to remove the authentication function of MUA directly and transfer the authentication responsibility to receiving server completely. MUA only exists as the front end and is not responsible for the actual authentication

Therefore, the sender can not fill in the from header, but the sending receiver can fill in according to the user name / password, which can also avoid many security problems caused by the inconsistency between MUA and sending server

A18 Replay attack to subvert DKIM signature

1. Since MUA has verified the DKIM signature of the first from header, it can't display the second from header, otherwise the verification will lose its function. The real address displayed by MUA should be parsed from the result of DKIM verification, not directly from the e-mail

2. Adopt the solution of A17.3, let MUA not directly fill in the from header. In this way, attackers will not be able to directly manipulate the from header of e-mail, and naturally have no chance to exploit the inconsistency of multi from header vulnerability

3. Using over-sign technology, we can avoid multi from header and have certain resistance to various attacks based on from header

Appendix 2: Teamwork

Team Member	Work
RUiN	Raise the topic. Reference researching. Demo implementation. Writing the report.
SA	Reference researching. Relative work analysis. Further study and research. Writing the report.
XJC	Reference researching. Relative work analysis. Writing the report.
XHY	Reference researching. Writing the report.

References

1. Composition Kills: A Case Study of Email Sender Authentication - Jianjun Chen, Vern Paxson, Jian Jiang; 29th USENIX Security Symposium.

2. Towards More Security in Data Exchange: Defining Unparsers with Context-Sensitive Encoders for Context-Free Grammars - Lars Hermerschmidt, Stephen Kugelmann, Bernhard Rumpe; 2015 IEEE Security and Privacy Workshops.
3. IMAP/POP AUTHorize Extension for Simple Challenge/Response - J. Klensin, R. Catoe, P. Krumviede; RFC 2195, 1997.
4. End-to-End Measurements of Email Spoofing Attacks - Hang Hu, Gang Wang; 27th USENIX Security Symposium.
5. Domain-based Message Authentication, Reporting, and Conformance (DMARC) - M. Kucherawy, Ed., E. Zwicky, Ed.; RFC 7489, 2015.
6. Email Authentication Mechanisms: DMARC, SPF and DKIM - Stephen Nightingale.