SMTP smuggling

Gennaio 2024

-- Francesco Gennai –
CNR-ISTI

Sommario

Introduzione. 1
Una semplice descrizione del percorso di una email. 2
Esempio 1: Sessione MUA – MSA. 3
Esempio 2: sessione MTA – MTA. 3
Descrizione. 4
Elementi di rilievo. 4
Terminazione dei comandi SMTP e delle righe della email. 4
Il punto. 4
Trasmissione di più email durante la stessa sessione SMTP. 5
Le autenticazioni. 5
SPF - Sender Policy Framework (SPF) for Authorizing Use of Domains in Email. 6
DKIM - DomainKeys Identified Mail (DKIM) Signatures. 7
DMARC - Domain-based Message Authentication, Reporting, and Conformance (DMARC). 9
Invio di email di phishing nello scenario fin qui descritto. 10
Command Pipelining - SMTP Service Extension for Command Pipelining. 11
SMTP smuggling. 13
C’è di peggio! 16
Conclusioni. 17

Introduzione.

Questa nota tecnica si riferisce alla vulnerabilità SMTP smuggling, annunciata il 18 dicembre 2023 dalla società SEC Consult con sede in Vienna.

Con questa nota intendo descrivere sinteticamente lo scenario tecnico in cui l'SMTP smuggling si potrebbe attuare.
Per fare questo, nella prima parte cerco di riassumere il funzionamento del protocollo SMTP, di alcune sue estensioni e di alcuni "protocolli"/"meccanismi" di autenticazione (quali SPF, DKIM e DMARC) che è necessario prendere in considerazione per una migliore comprensione del SMTP smuggling.
Nella seconda parte passo alla descrizione del funzionamento del SMTP smuggling.

La seguente tabella introduce alcuni termini tecnici e, quando necessario, alcuni termini corrispettivi di uso corrente, utili per la presente nota tecnica.

Tecnico	**Di uso corrente**
MUA = Message User Agent	**Client di posta, mail client**: è un software per la composizione ed invio di email e per l'accesso alla mailbox (Thunderbird, Outlook, Windows Mail, etc.) installato sul proprio dispositivo (desktop, mobile) o una interfaccia web (web mail)
MTA = Message Transfer Agent	**Server di posta, server email, email server, mailer, server SMTP, SMTP server**: è un nodo della rete che tramite un opportuno software (Postfix, Exim, Sendmail, etc..) svolge le funzioni di instradamento (relay) delle email verso la loro destinazione.
MSA = Message Submission Agent	E' una componente di un MTA che svolge anche la funzione di accettazione delle email provenienti da un MUA
SMTP	Simple Mail Transfer Protocol. RFC 5321. E' il protocollo di comunicazione con il quale le email vengono trasferite dal client al server.
Client SMTP	È il processo che, in una comunicazione SMTP, attiva la sessione con la funzione di mittente (sender, transmitter). Questo ruolo è svolto dal MUA o dal MTA.
Server SMTP	È il processo ricevente (receiver) che attende e riceve le richieste di attivazione della sessione dal client SMTP. Questo ruolo è svolto dal MSA o dal MTA.

Header e body	L'email è strutturata in due parti: intestazione (header) e corpo (body), separate da una riga vuota. La definizione completa del suo formato si trova nella RFC 5322 (Internet Message Format) [1]

Una semplice descrizione del percorso di una email.

L'utente, mediante il proprio MUA, consegna l'email che intende inviare al MSA che, tramite l'MTA, provvederà all'inoltro della stessa verso la sua destinazione.
Le comunicazioni tra MUA e MSA avvengono tramite il protocollo SMTP.
La comunicazione tra MUA e MSA potrebbe avvenire anche senza ricorrere al protocollo SMTP (per esempio, consegna della email al MTA attraverso un file system condiviso tra web mail e MTA).
Per lo scopo di questa nota tecnica mi limito a prendere in considerazione solo i MUA che inviano l'email tramite il protocollo SMTP.

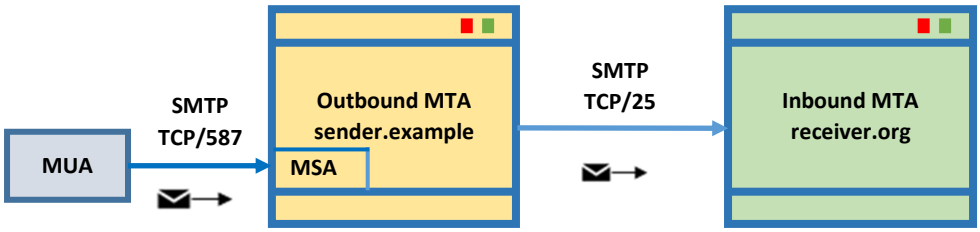

Figura 1: percorso della email dal MUA al MTA di destinazione.

Vediamo adesso cosa avviene durante le due sessioni SMTP, quella tra MUA e MSA e quella tra MTA e MTA.

[1] In questa nota tecnica utilizzo il termine header (non la sua traduzione in intestazione) in quanto risulta di uso comune anche in italiano. L'header della email è composto da una sequenza di "header fields", ognuno dei quali è identificato da un proprio nome (From, To, Subject, etc…). Spesso, per identificare uno di questi header fields, si utilizza la forma contratta header <nome dell'header> (header From, header To, etc..). Nel testo mi riferisco ad un header field sia con la forma contratta (header From, …) che con la sua traduzione in "campo dell'header" (il campo dell'header From contiene… etc..), spero a favore di una maggiore chiarezza del testo.

Esempio 1: Sessione MUA – MSA.

Il client di posta (MUA) apre una sessione SMTP con il submission server (MSA), sulla porta TCP 587.

```
CLIENT (MUA)                TCP (587)                SERVER (MSA)

                                          220 sender.example ESMTP
EHLO mua.example
                                               250-sender.example
                                                     250-8BITMIME
                                                250 AUTH PLAIN LOGIN
AUTH PLAIN
                                                                334
[CREDENTIALS]
                                       235 Authentication succeeded
MAIL FROM:<bob@sender.example>
                             250 2.1.0 <bob@sender.example> Sender ok
RCPT TO:<alice@receiver.org>
                          250 2.1.5 <alice@receiver.org> Recipient ok
DATA
                                                    354 Enter mail
From: Bob Trant <bob@sender.example>
To: alice@receiver.org
Subject: messaggio di test

prova prova
.
                              250 2.0.0 Message accepted for delivery
QUIT
                          221 2.0.0 sender.example closing connection
```

Esempio 2: sessione MTA – MTA.

Il server outbound MTA apre una sessione SMTP con il server inbound MTA sulla porta TCP 25.

```
CLIENT (MTA)                TCP (587)                SERVER (MTA)

                                        220 receiver.org ESMTP
EHLO sender.example
                                             250-receiver.org
                                                 250-8BITMIME
                                               250 PIPELINING
MAIL FROM:<bob@sender.example>
                             250 2.1.0 <bob@sender.example> Sender ok
RCPT TO:<alice@receiver.org>
                          250 2.1.5 <alice@receiver.org> Recipient ok
DATA
                                                    354 Enter mail
From: Bob Trant <bob@sender.example>
To: alice@receiver.org
Subject: messaggio di test

prova prova
.
                              250 2.0.0 Message accepted for delivery
QUIT
                            221 2.0.0 receiver.org closing connection
```

Descrizione.

Si può notare che la sessione SMTP avviene con lo scambio di vari comandi/risposte tra client e server: tra MUA e MSA del outbound MTA (esempio 1) e successivamente tra outbound MTA e inbound MTA (esempio 2).
All'interno di ciascuna sessione possiamo individuare e evidenziare le parti in cui client e server SMTP si scambiano i vari comandi e risposte del protocollo SMTP: comandi del client (righe verdi) e relative risposte del server (righe nere) e possiamo evidenziare la parte in cui avviene l'effettivo trasferimento dell'intera email (comando SMTP DATA – righe azzurre) dal client SMTP (sender), al server SMTP (receiver).
Possiamo anche notare una differenza tra la sessione SMTP MUA-MSA e la sessione MTA-MTA per il trasferimento della stessa email. Nella prima infatti, vi è un punto (comando AUTH) in cui il client invia al server le credenziali di autenticazione dell'utente che sta effettuando l'invio della email. Funzionalità fondamentale mediante la quale il mittente della email viene autenticato.
Nel secondo esempio (MTA-MTA) l'autenticazione non è presente, ma, come vedremo nei prossimi paragrafi, qui, entrano in funzione altri protocolli standard per assolvere ad una simile funzione di autenticazione, non più relativa all'utente mittente, ma al dominio degli indirizzi email presenti nel comando SMTP MAIL FROM e nel campo From: dell'header della email.

Elementi di rilievo.

Nei seguenti paragrafi vengono introdotti alcuni concetti, componenti e modalità di funzionamento del sistema email Internet, con l'utilizzo di esempi, che, anche se analizzati in modo superficiale, dovrebbero essere propedeutici alla comprensione, da un punto di vista tecnico, della vulnerabilità SMTP smuggling e delle sue potenziali conseguenze.

Terminazione dei comandi SMTP e delle righe della email.

Gli standard SMTP (RFC 5321) e Internet Message Format (RFC 5322) [2] richiedono che ogni comando SMTP, le righe che compongono l'intestazione (header) e il corpo (body) della email siano terminati dalla sequenza di caratteri di controllo <CR><LF>.
Una sintassi alternativa per indicare i caratteri <CR> e <LF> è \r per <CR> e \n per <LF>. Nel seguito di questa nota, in base al contesto, utilizzerò l'una o l'altra.

Il punto.

Per lo scopo di questa nota tecnica, è importante osservare che, lo standard SMTP prevede che li comando DATA (trasmissione della email - header + body) termini con la sequenza di caratteri

[2] Presumibilmente entro il primo semestre del prossimo anno, le specifiche del protocollo SMTP e del Internet Message Format (email core protocols) diventeranno Standard Internet dopo un periodo di circa 40 anni, durante i quali, attraverso osservazioni e verifiche sull'evoluzione del sistema email Internet, sono state riviste e perfezionate per la loro versione finale e standard. Per l'occasione saranno pubblicate tre nuove RFC, due che andranno a sostituire rispettivamente la RFC 5321 e la RFC 5322 mentre la terza "Applicability Statement for IETF Core Email Protocols" conterrà osservazioni e raccomandazioni sugli email core protocols e le loro varie estensioni.

<CR><LF>.<CR><LF> (la sequenza contiene il carattere punto), dove il primo <CR><LF> è in realtà il terminatore della riga precedente.

Da un punto di vista operativo, abbiamo che, con la trasmissione della sequenza <CR><LF>.<CR><LF>, il client SMTP comunica al server SMTP la fine della email. Il server SMTP deve considerare la sequenza finale di caratteri **.**<CR><LF> come conclusione del comando DATA, quindi questi caratteri non appartengono al testo della email trasmessa.

Trasmissione di più email durante la stessa sessione SMTP.

All'interno della stessa sessione SMTP è possibile la trasmissione di più email, semplicemente iniziando una nuova sequenza di comandi dopo la terminazione del comando DATA della email precedente.
La seguente è una sintetica rappresentazione della sequenza di comandi che il client SMTP invia al server SMTP (per brevità non sono riportate le risposte del server SMTP e i caratteri <CR><LF> al termine di ogni riga, eccetto che quelli necessari ad evidenziare la riga contenente il punto):

```
MAIL FROM:<…..>
RCPT TO:<… . .>
DATA
… …
… …
<CR><LF>.<CR><LF>      termine della prima email
MAIL FROM:<…..>        primo comando della seconda email
RCPT TO:<… . .>
DATA
… …
… …
<CR><LF>.<CR><LF>
QUIT
```

Le autenticazioni.

Come precedentemente osservato, durante la sessione tra MUA e MSA, avviene l'autenticazione del mittente della email. Con questa autenticazione l'MSA può verificare che l'indirizzo email presente nel comando SMTP MAIL FROM e/o nel campo From dell'header della email corrisponda a quello assegnato all'utente autenticato.
L'autenticazione SMTP favorisce una corretta amministrazione del servizio email da parte del gestore del outbound MTA, evitando falsificazioni del mittente di una email.
Cosa può fare l'inbound MTA per autenticare gli indirizzi email presenti nel comando SMTP MAIL FROM e nel campo From dell'header di una email che sta ricevendo dall'outbound MTA?
Per quanto riguarda l'utente mittente non può fare niente, del resto, non è suo compito in quanto la verifica sull'utente mittente la può tecnicamente fare solo l'outbound MTA, durante la fase di accettazione della email da parte del proprio MSA, essendo questa una relazione tra utente e gestore della mailbox. È compito del gestore mittente assicurarsi che l'indirizzo presente nel campo From dell'header delle email inviate corrisponda effettivamente alla persona o al ruolo autenticati sul proprio server (esempi: direttore@dominioazienda, segretaria@dominoazienda, bob.rossi@dominioazienda).

L'inbound MTA può però verificare che l'email in ricezione provenga effettivamente dal MTA (outbound) che è autoritativo per il dominio presente nell'indirizzo del mittente. Con queste verifiche l'inbound MTA può evitare la ricezione di email con indirizzi falsi provenienti da MTA malevoli, tipicamente utilizzati dagli spammers.
Questa operazione di verifica è identificabile come *autenticazione del dominio mittente* o *autenticazione della sorgente di una email*.
Nel corso degli anni, sono stati definiti alcuni "protocolli"/"meccanismi" per effettuare l'autenticazione della sorgente di una email. Questi sono: SPF (Sender Policy Framework), DKIM (DomainKeys Identified Mail) e, più recentemente, DMARC (Domain-based Message Authentication, Reporting, and Conformance).
In modo molto sintetico li descrivo qui.

SPF - Sender Policy Framework (SPF) for Authorizing Use of Domains in Email.

Mediante il protocollo SPF (RFC 7208), l'amministratore di un dominio [3], pubblica gli indirizzi IP degli outbound MTA autorizzati ad inviare email per tale dominio.
Questo avviene mediante la pubblicazione di un opportuno record TXT nel DNS. Esempio:
per il dominio sender.example potrei avere il seguente record:

```
sender.example TXT "v=spf1 include:spf.protection.outlook.com ~all"
```

dove spf.protection.outlook.com corrisponde a:

```
spf.protection.outlook.com TXT "v=spf1 ip4:40.92.0.0/15 ip4:40.107.0.0/16
ip4:52.100.0.0/14 ip4:104.47.0.0/17 ip6:2a01:111:f400::/48
ip6:2a01:111:f403::/49 ip6:2a01:111:f403:8000::/50
ip6:2a01:111:f403:c000::/51 ip6:2a01:111:f403:f000::/52 -all"
```

I record TXT dell'esempio dichiarano gli indirizzi IP autorizzati ad inviare email con dominio sender.example nel comando MAIL FROM (in questo esempio, gli indirizzi IP, sono quelli del provider email Microsoft Outlook).
Operativamente, il server SMTP di inbound MTA, estrae il dominio dall'indirizzo email presente nel comando MAIL FROM ricevuto dal client SMTP di outbound MTA e utilizza il dominio estratto (per esempio: sender.example) per interrogare il DNS ed ottenere i record TXT-SPF (appena descritti).
L'autenticazione dell'outbound MTA avviene verificando se l'indirizzo IP della sessione SMTP in corso è presente nella lista degli IP del record TXT:
Per esempio, se l'indirizzo IP è presente nella lista può accettare ed inoltrare l'email verso la sua destinazione, se assente può rifiutare la sessione SMTP oppure accettare l'email aggiungendo un flag che indichi che si tratta potenzialmente di una email di spam.
SPF si applica al dominio dell'indirizzo email presente nel comando SMTP MAIL FROM. Il From dell'header, che è quello visualizzabile dall'utente destinatario all'apertura della email nel proprio MUA, può avere un valore arbitrario. Per esempio:

Con il seguente record SPF, inserito nel name server di attacker.net, vengono pubblicati gli indirizzi IP dei due outbound MTA.

```
attacker.net. TXT "v=spf1 ip4:192.12.192.10/32 ip4:192.12.192.11/32 ~all"
```

[3] L'amministratore di dominio è colui che gestisce il name server dell'organizzazione (gestisce i vari record A, TXT, CNAME, …)

L'outbound MTA riceve da uno di questi indirizzi IP la seguente email:
Nota: nei prossimi due esempi, riduco la rappresentazione delle sessioni SMTP riportando solo i comandi e i dati che il client invia al server.

EHLO attacker.net\r\n **MAIL FROM:<utente@attacker.net>\r\n** **RCPT TO:<alice@receiver.org>\r\n** **DATA\r\n**	**comandi SMTP**
From: Bob Trant <bob@sender.example>\r\n **To: alice@receiver.org\r\n** **Subject: messaggio di test\r\n** **\r\n** **prova prova\r\n** **.\r\n**	**email trasmessa**

Per comprendere questo esempio:

- il dominio presente nel comando MAIL FROM *attacker.net* è regolarmente autenticato dalla presenza del relativo record SPF nel DNS (la sessione SMTP proviene da uno degli IP listati nel record SPF).
- l'email passa l'autenticazione SPF e si presenta al destinatario come proveniente da *Bob Trant <bob@sender.example>*. Potrebbe essere email di phishing [4].

DKIM - DomainKeys Identified Mail (DKIM) Signatures.

Il protocollo DKIM (RFC 6376) si può considerare complementare a SPF, in quanto permette l'autenticazione del dominio dell'indirizzo email presente nel campo From dell'header.
DKIM si basa sulla firma elettronica di una parte selezionabile dei campi presenti nell'header della email e di tutto o parte il contenuto (body) della email.
La firma viene applicata dal outbound MTA e la chiave pubblica per la verifica della validità della firma è pubblicata dall'amministratore del dominio mittente (o altra entità come, per esempio, l'amministratore dell'oubound MTA) nel DNS.
Le informazioni necessarie all'inbound MTA per la verifica della firma vengono in parte prelevate dal campo dell'header DKIM-Signature, che l'outbound MTA aggiunge all'header della email ed in parte prelevate dal DNS
Un esempio.
Con il record TXT del DNS viene pubblicata la chiave pubblica che l'inbound MTA dovrà utilizzare per la verifica della firma.
Esempio:
dkim._domainkey.dom.com. TXT v=DKIM1; h=sha256; k=rsa; p=<-chiave pubblica->

[4] Per cercare di rilevare questo tipo di attacco con l'uso di SPF, il sistema antispam installato sul inbound MTA, può imporre che l'indirizzo presente nel comando SMTP MAIL FROM sia identico a quello presente nel campo From dell'header (*identifier alignment*). Questa rappresenta una potenziale violazione al funzionamento del sistema email Internet che non richiede l'allineamento dei due valori. Va detto che gli outbound MTA tendono oramai, quando possibile, a trasmettere email con tale allineamento. Il "meccanismo" DMARC, attualmente in fase di revisione da parte del relativo gruppo di lavoro IETF, è stato introdotto per consentire più efficienti policy che includono anche gli allineamenti tra MAIL FROM e header From.

L'outbound MTA aggiunge all'header della email in uscita il campo DKIM-Signature:

```
DKIM-Signature: v=1; a=rsa-sha256; d=dom.com; s=dkim;
c=relaxed/relaxed; q=dns/txt; t=1703868957;
h=from:to:subject:date;
bh=MHCzUDU2Nrf9MDFyWzR1Wjc5OTAyMjD0dUY3ODlqART=;
b=hyjCnOfCKDdLZdKIa9G1q7LoLWlEniSbz+cyuU2zGrtruF002dcFVoG3WUNAtNzG
```

Nel campo DKIM-Signature possiamo evidenziare i seguenti tag:

d=`dom.com` - dichiarazione del dominio appartenente all'organizzazione che applica la firma DKIM alla email. Non necessariamente è l'organizzazione a cui appartiene il dominio dell'indirizzo email presente nel header From.

h=`from:to:subject:date` – lista dei campi dell'header della email che sono stati firmati.

bh=`MHCzUDU2Nrf9MDFyWz`… hash del corpo (body) della email.

b=`hyjCnOfCKDdLZdKIa9G`… risultato della firma.

s=`dkim` – è un selettore che, concatenato alla costante _domainkey e al dominio presente in **d**, permette di creare il nome del record TXT contenente la chiave pubblica della firma.
L'inbound MTA dovrà interrogare il DNS per ottenere da tale record TXT (record DKIM) la chiave pubblica per la verifica della firma DKIM.
Esempio:
Record TXT pubblicato nel DNS ed email ricevuta dal inbound MTA:

```
dkim._domainkey.attacker.net. TXT v=DKIM1; h=sha256; k=rsa; p=<-chiave
pubblica->
```

`EHLO attacker.net\r\n`
`MAIL FROM:<utente@attacker.net>\r\n`
`RCPT TO:<alice@receiver.org>\r\n`
`DATA\r\n`
	comandi SMTP

`DKIM-Signature: v=1; a=rsa-sha256;\r\n`
` d=attacker.net; s=dkim;\r\n`
` c=relaxed/relaxed; q=dns/txt; t=1703931604;\r\n`
` h=from:to:subject:date;\r\n`
` bh=MHCzUDU2N … … dUY3ODlqART=;\r\n`
` b=hyjCnOfCKDdLZdKI … … F002dcFVoG3WUNAtNzG\r\n`
`From: Bob Trant <bob@sender.example>\r\n`
`To: alice@receiver.org\r\n`
`Subject: messaggio di test\r\n`
`\r\n`
`prova prova\r\n`
`.\r\n`
	email trasmessa

Sintetica analisi della email firmata con DKIM.

L'inbound MTA ottiene la chiave pubblica per la verifica della firma DKIM della email interrogando il DNS per il record TXT:

dkim._domainkey.attacker.net.

dove il selettore è dkim, definito con il tag **s** dell'header DKIM-Signature e _domainkey è una parte costante predefinita dallo standard.
L'eventuale successo della verifica della firma conferma che l'email è stata effettivamente autenticata da un outbound MTA appartenente all'organizzazione che ha il controllo del dominio attacker.net (dominio del tag **d**).
Occorre però notare che il dominio attacker.net, presente nel tag **d,** è diverso da quello presente nel campo From del header (bob@sender.example).

SPF e DKIM sono protocolli di autenticazione, quindi servono ad autenticare la sorgente di una email: per SPF viene autenticato l'indirizzo IP sorgente, per DKIM, mediante il dominio presente nel tag **d**, può essere individuata l'organizzazione che ha firmato l'email immessa nel sistema email Internet.
Ogni inbound MTA ha le proprie "policy" di controllo del traffico email in ingresso, che, sulla base del successo o meno di queste autenticazioni, possono determinare le successive azioni, quali rifiuto della sessione SMTP, tag della email come spam email in quarantena, etc…
Bisogna anche osservare che:

- Il fallimento di una verifica SPF o DKIM non necessariamente significa che l'email sia da scartare. Sia per SPF che per DKIM vi sono casi in cui la verifica può fallire anche se l'email proviene dalla sua valida sorgente.
- In assenza di questi protocolli di autenticazione l'email deve essere trattata come una normale email Internet, eventualmente filtrata in base alle policy locali all'inbound MTA che potranno tenere conto di altri parametri per verificarne la validità.

DMARC - Domain-based Message Authentication, Reporting, and Conformance (DMARC).

DMARC, che si basa sull'Informational RFC 7489, introduce il cosiddetto *identifier alignment* per SPF e DKIM.
Anche DMARC prevede la pubblicazione, da parte dell'amministratore del dominio, di un record TXT nel DNS il cui nome si compone dal suffisso _dmarc seguito dal dominio
Un esempio:

```
_dmarc.sender.example. TXT v=DMARC1; p=reject;
      rua=mailto:postmaster@sender.example; pct=100; adkim=s; aspf=s
```

(l'intero record è visualizzato qui su 2 righe)

Per lo scopo di questa nota tecnica è utile sapere che con DMARC l'amministratore del dominio (in questo caso sender.example) definisce le policy che gli inbound MTA dovrebbero applicare per l'autenticazione delle possibili sorgenti (outbound MTA) delle email ricevute con MAIL FROM o header FROM contenenti il dominio sender.example.
In particolare con il tag adkim e aspf si possono richiedere due tipi di allineamento del dominio:
s = strict alignment - significa che deve esserci la corrispondenza esatta (vedi sotto).
r = relaxed alignment (che è il default, in assenza del tag) – significa che la corrispondenza è estendibile anche ai sottodomini (esempio: sub.sender.example è un sottodominio di sender.example) (vedi sotto).

Per SPF l'allineamento si deve verificare tra il dominio dell'indirizzo email presente nel comando SMTP MAIL FROM e il dominio dell'indirizzo email presente nel campo From dell'header.
Per DKIM l'allineamento si deve verificare tra il dominio presente nel tag **d** del campo DKIM-Signature (inserito nell'header dal MTA dove avviene la firma) e il dominio dell'indirizzo email presente nel campo From dell'header.
Altri tag presenti in questo esempio, meno interessanti per lo scopo di questa nota tecnica sono:
p= (none, quarantine o reject) indica l'azione che l'inbound MTA dovrebbe intraprendere quando sia l'autenticazione SPF che l'autenticazione DKIM falliscono.
pct= (percentuale) indica la percentuale di email alla quale l'inbound MTA dovrebbe applicare la policy DMARC. È utile per consentire una fase di attivazione controllata di DMARC, durante la quale l'amministratore di sender.example può effettuare le verifiche che ritiene opportune.

DMARC non richiede la presenza di entrambi i protocolli di autenticazione SPF e DKIM e non richiede nemmeno che, se presenti, l'autenticazione di entrambi abbia successo (può essere sufficiente il successo di uno dei due).
In generale, anche in assenza di DMARC, una email che passa la verifica di autenticazione di almeno uno dei due tra SPF e DKIM ed ha l'allineamento del dominio, ha buone probabilità di essere consegnata al destinatario.

Invio di email di phishing nello scenario fin qui descritto.

La sintassi del campo From dell'header di una email è descritta dalla RFC 5322. In particolare il campo From è composto da due sottocampi, che sono il Display Name e l'Indirizzo Email.
Per esempio:
Francesco Gennai <francesco.gennai@isti.cnr.it>
dove:
Francesco Gennai è il Display Name
francesco.gennai@isti.cnr.it è l'indirizzo email.
Il contenuto del Display Name è libero. Esempio:
FraGen <francesco.gennai@isti.cnr.it> è comunque un campo From valido.

Ad oggi le email di phishing, per passare i controlli di autenticità della loro sorgente, devono essere spedite con un dominio mittente autenticato/autenticabile.
Considerando la diffusione della autenticazione della sorgente di una email, il mittente del phishing (spammer) è quasi sempre costretto ad inviare email con domini autenticabili se vuole che l'email abbia qualche probabilità di essere consegnata nella mailbox del destinatario.
Allo "spammer" mittente del phishing non resta che inserire nel Display Name del header From un identificativo che possa rappresentare il mittente che intende falsificare. Azione che può avere successo in tutti quei casi (la maggioranza) in cui il client di posta del destinatario visualizza solo il Display Name.
Infatti, se chiediamo al nostro MUA (client di posta) di visualizzare il sorgente dell'header o di un'intera email di phishing che abbiamo ricevuto, noteremo che frequentemente l'indirizzo email presente nel header From non ha alcuna relazione con ciò che lascerebbe intendere il Display Name.

Esempi di header From di due classiche email di phishing.

Immaginiamo che ACME sia il nome di una organizzazione e che il proprio dominio email sia acme.it

Un dipendente di tale organizzazione potrebbe ricevere una email di phishing con il seguente campo From:
From: ACME Email Admin <info@**accel-tw.com**>
oppure, in un'altra email di phishing
From: DHL Express <troy@**devochki.top**>

Dove, nel primo esempio:
ACME Email Admin è il display name e info@accel-tw.com è l'indirizzo email.
I più diffusi MUA visualizzano l'email come proveniente da **ACME Email Admin**, senza rendere evidente l'effettivo indirizzo email del mittente, che sarebbe info@accel-tw.com.
Visto il "Display Name", che vorrebbe identificare l'amministratore email di ACME, il dominio che ci dovremmo presumibilmente aspettare al posto di accel-tw.com è **acme.it**.

Nel secondo esempio, **DHL Express** è il display name e troy@devochki.top l'indirizzo email.
Il MUA visualizza l'email come proveniente da **DHL Express** senza evidenziare l'effettivo indirizzo email dell'header From, che faciliterebbe il riconoscimento del phishing.
Presumibilmente, in questo esempio, validi domini per l'indirizzo email dell'header From potrebbero essere **dhl.com** o **dhl.it**. Il dominio **devochki.top,** che non sembra avere alcuna relazione con un azienda dal nome DHL, dovrebbe generare allarme nel destinatario, ma l'indirizzo email che lo contiene generalmente non viene visualizzato dal MUA [5].

I domini accel-tw.com e devochki.top sono regolarmente autenticati tramite SPF e/o DKIM.

Cosa accade con la vulnerabilità SMTP Smuggling?

Il mittente del phishing, venuto a conoscenza della vulnerabilità, potrebbe inviare l'email con un indirizzo email valido, istituzionale! che verrebbe regolarmente consegnato nella mailbox del destinatario.
Richiamando gli esempi sopra, potremmo avere:
From: ACME Mail Admin <admin@**acme.it**>
From: DHL Express <dhlexpress@**dhl.com**>

In questo caso l'eventuale visualizzazione dell'indirizzo email del From porterebbe a ritenere valida l'email.

Command Pipelining - SMTP Service Extension for Command Pipelining.

Prima di arrivare alla descrizione della vulnerabilità SMTP smuggling è necessario introdurre l'estensione SMTP Command Pipelining (RFC 2920).

[5] In ogni MUA esiste la possibilità di richiedere la visualizzazione dell'intero header della email, dove si trova l'header From o, alternativamente, dell'intero sorgente del messaggio. Per alcuni MUA è anche possibile modificare le impostazioni in modo da visualizzare sempre l'intero header From.

Questa estensione SMTP, supportata dalla maggioranza dei server email Internet, consente al client SMTP di inviare più comandi senza dover aspettare la risposta per ognuno di essi, risposte che il server darà al termine della sequenza di comandi inviata dal client.
Riprendendo l'esempio 2 della sessione SMTP tra outbound MTA e inbound MTA, che già si presentava senza l'uso dell'estensione Command Pipelining:

```
CLIENT (MTA)                TCP (587)                SERVER (MTA)

                                    220 receiver.org ESMTP\r\n
EHLO sender.example\r\n
                                          250-receiver.org\r\n
                                             250-8BITMIME\r\n
                                           250 PIPELINING\r\n
MAIL FROM:<bob@sender.example>\r\n
                 250 2.1.0 <bob@sender.example> Sender ok\r\n
RCPT TO:<alice@receiver.org>\r\n
              250 2.1.5 <alice@receiver.org> Recipient ok\r\n
DATA\r\n
                                          354 Enter mail\r\n
From: Bob Trant <bob@sender.example>\r\n
To: alice@receiver.org\r\n
Subject: messaggio di test\r\n
\r\n
prova prova\r\n
.\r\n
                  250 2.0.0 Message accepted for delivery\r\n
QUIT\r\n
                221 2.0.0 receiver.org closing connection\r\n
```

con l'uso del Command Pipelining il dialogo tra client e server SMTP si può trasformare nel modo seguente:

```
CLIENT (MTA)                TCP (587)                SERVER (MTA)

                                    220 receiver.org ESMTP\r\n
EHLO sender.example\r\n
                                          250-receiver.org\r\n
                                             250-8BITMIME\r\n
                                           250 PIPELINING\r\n
MAIL FROM:<bob@sender.example>\r\n
RCPT TO:<alice@receiver.org>\r\n
DATA\r\n
                 250 2.1.0 <bob@sender.example> Sender ok\r\n
              250 2.1.5 <alice@receiver.org> Recipient ok\r\n
                                          354 Enter mail\r\n
From: Bob Trant <bob@sender.example>\r\n
To: alice@receiver.org\r\n
Subject: messaggio di test\r\n
\r\n
prova prova\r\n
.\r\n
                  250 2.0.0 Message accepted for delivery\r\n
QUIT\r\n
                221 2.0.0 receiver.org closing connection\r\n
```

Come si può notare negli esempi riportati sopra, senza l'uso dell'estensione Command Pipelining il client resta in attesa della risposta del server 6 volte, mentre con l'uso dell'estensione resta in attesa 4 volte, rendendo il colloquio client-server più efficiente.

L'ottimizzazione si può rilevare anche osservando che nel caso del PIPELINING la trasmissione di più comandi (e la ricezione delle relative risposte) avviene come una sequenza di caratteri che può essere contenuta all'interno di un unico segmento TCP:

```
MAIL FROM:<bob@sender.example>\r\nRCPT TO:<alice@receiver.org>\r\nDATA\r\n
```

SMTP smuggling.

L'email prima di essere trasmessa o dopo la sua ricezione si trova memorizzata, come file di testo, rispettivamente sui file system dei sistemi operativi del client o del server SMTP.
I file system di sistemi operativi diversi possono utilizzare diverse sequenze di caratteri per identificare la fine delle righe di un file di testo, quindi di una email. Per esempio, Windows e OpenVMS utilizzano <CR><LF>, mentre Linux utilizza <LF>.
Nella evoluzione del sistema email Internet, questa discrepanza tra i diversi file system ha avuto come conseguenza l'implementazione di mailer il cui client SMTP trasmette email non conformi allo standard, utilizzando <LF> come fine riga, al posto di <CR><LF> e, conseguentemente allo sviluppo di mailer che, in ricezione, oltre al formato standard (<CR><LF>), accettano anche comandi o email il cui terminatore di riga è <LF>. Storicamente l'accettazione di email non conformi allo standard è un aspetto che ha favorito la massima interoperabilità tra i vari mailer.
In conclusione è risultato possibile che varie combinazioni della sequenza che identifica il termine del comando DATA siano riconosciute come valide (condizione che può variare tra mailer e mailer non perfettamente conformi agli standard):
<LF>.<LF>
<LF>.<CR><LF>
<CR><LF>.<CR>
<CR>.<CR>

Dato che la sequenza <CR><LF>.<CR><LF>, che rappresenta una riga contenente solo un punto, ha una sua particolare semantica (fine della email trasmessa), il protocollo SMTP prevede il cosiddetto "*dot stuffing*" per evitare che righe costituite da un solo punto, eventualmente presenti nel testo della email, possano provocare un troncamento indesiderato della stessa durante la sua trasmissione.
Sinteticamente, il dot stuffing prevede che il client SMTP trasmetta una riga contenente un doppio punto, per ogni riga del testo della email composta da un solo punto, e che il server provveda a rimuovere un punto dalle righe con doppio punto ricevute [6].
Anche l'implementazione dell'estensione PIPELINING non è sempre strettamente conforme allo standard, che prevede, per esempio, che dopo il comando DATA il client SMTP si metta in attesa

[6] Per una descrizione completa del dot stuffing suggerisco la lettura del paragrafo *4.5.2 Transparency* della RFC 5321.

della risposta dal server. Vi sono perciò mailer il cui server SMTP accetta l'intera sequenza dei comandi senza rilevare l'anomalia.

Vediamo adesso come una particolare combinazione tra queste non conformità agli standard tra l'outbound MTA e l'inbound MTA contribuisca al funzionamento del SMTP smuggling.

Prendiamo come esempio la seguente email inviata dall'utente bob@acme.it attraverso il proprio outbound MTA:

```
From: Bob <bob@acme.it>\r\n
To: <alice@receiver.org>\r\n                HEADER
Subject: Saluti\r\n
\r\n
Saluti dalla Sardegna,\r\n
Bob\n
.\r\n
MAIL FROM:<presidente@acme.it>\r\n
RCPT TO:<alice@receiver.org>\r\n
DATA\r\n
From: presidente@acme.it\r\n                  BODY
To: alice@receiver.org\r\n
Subject: Comunicazione\r\n
\r\n
Cara Alice,\r\n
ho deciso che … blah .. blah.\r\n
\r\n
ACME President\r\n
```

Nell'esempio viene rappresentato sia l'header che il testo contenuto nella email (body) che Bob invia ad Alice.
In assenza della vulnerabilità SMTP smuggling, Alice riceverà una email, con mittente bob@acme.it, contenente l'intero testo (body), cioè, escludendo l'header per semplificare, Alice potrà leggere esattamente questo testo:

```
Saluti dalla Sardegna,
Bob
.
MAIL FROM:<presidente@acme.it>
RCPT TO:<alice@receiver.org>
DATA
From: presidente@acme.it
To: alice@receiver.org
Subject: Comunicazione

Cara Alice,
ho deciso che … blah .. blah.

ACME President
```

al punto che Alice potrebbe rispondere a Bob “Caro Bob, grazie per i saluti dalla Sardegna, ma non ho capito cosa mi volevi dire nel resto della email che mi hai inviato”.

Osserviamo però la particolare sequenza **\n.\r\n** (evidenziata in rosso nel precedente esempio).
Consideriamo adesso una coppia di MTA (outbound e inbound) ognuno con le proprie non conformità rispetto agli standard email, in particolare:
Outbound MTA:

- non riconosce la sequenza \n.\r\n come riga contenente un solo punto (probabilmente a causa dell’assenza del primo <CR>), perciò non applica il dot stuffing, inoltrando l’email, inalterata, verso l’inbound MTA.

Inbound MTA:

- accetta come terminatore di una riga di comando SMTP anche il solo carattere <LF> (non conforme allo standard, ma a favore di una massima interoperabilità).
- non rileva un errore nel PIPELINING dovuto alla ricezione di un comando DATA prima che abbia risposto ai precedenti comandi SMTP.

Durante la sessione SMTP l’inbound MTA riceve la seguente email (header + body) dall’outbound MTA (per brevità, tralascio la parte iniziale della sessione SMTP, riportando il colloquio SMTP a partire dal comando DATA):

```
[...]
DATA
From: Bob <bob@acme.it>\r\n
To: <alice@receiver.org>\r\n
Subject: Saluti\r\n
\r\n
Saluti dalla Sardegna,\r\n
Bob\n          riconoscimento del punto come
.\r\n          terminatore della prima email
MAIL FROM:<presidente@acme.it>\r\n   primo commando della seconda email
RCPT TO:<alice@receiver.org>\r\n
DATA\r\n                                  di colore verde è la sequenza
From: presidente@acme.it\r\n               di comandi/email ricevuti ed
To: alice@receiver.org\r\n               accettati in modalità PIPELINING
Subject: Comunicazione\r\n
\r\n
Cara Alice,\r\n
ho deciso che … blah .. blah.\r\n
\r\n
ACME President\r\n
\r\n.\r\n
QUIT
```

Quello che vediamo tra il comando DATA e il punto (prima del comando QUIT) corrisponde per l’oubound MTA alla trasmissione di una normale email, che **si trasforma per l’inbound MTA nella ricezione di due distinte email.**
La prima email è indirizzata ad alice@receiver.org e proviene da bob@acme.it.
La seconda email è sempre indirizzata ad alice@receiver.org ma **proviene da presidente@acme.it.**

La sessione SMTP che ha originariamente trasmesso l'email al MSA dell'outbound MTA (quella rappresentata qui è la successiva sessione SMTP tra outbound MTA e inbound MTA) è stata autenticata mediante le credenziali di Bob (bob@acme.it).
Entrambe le email hanno una valida autenticazione SPF, infatti entrambe provengono dall'indirizzo IP registrato nel record TXT (SPF) per il dominio acme.it (indirizzo IP dell'oubound MTA) ed entrambe hanno l'allineamento del dominio presente nei rispettivi MAIL FROM e header FROM.
Ne consegue che anche una eventuale validazione DMARC, nel caso sia presente, può avere successo (l'autenticazione SPF ha avuto successo).
In conclusione alice@receiver.org riceverà due email, quella con i saluti dalla Sardegna da parte di Bob, senza la parte successiva alla sequenza \n.\r\n e quella che ha nel campo From dell'header un valido indirizzo email istituzionale: **presidente@acme.it!** con il resto della email originale, cioè:

```
From: presidente@acme.it
To: alice@receiver.org
Subject: Comunicazione

Cara Alice,
ho deciso che … blah .. blah.

ACME President
```

Richiamando quanto descritto nel paragrafo *Invio di email di phishing nello scenario fin qui descritto,* lo spammer potrà avvalersi di questa vulnerabilità, per inviare email di phishing con un campo From dell'header che contiene validi indirizzi istituzionali!

C'è di peggio!

Nell'esempio riportato qui le due email che si generano con l'SMTP smuggling, hanno entrambe un indirizzo con lo stesso dominio (acme.it) nel campo From:
bob@acme.it, la prima e presidente@acme.it, la seconda.
Prendiamo ora in considerazione provider come Microsoft o GMX.de, che gestiscono milioni di domini. Entrambi sono risultati "compatibili" con la vulnerabilità (che hanno già risolto).
I due messaggi potrebbero avere domini diversi nei rispettivi campi From, purché siano domini gestiti dallo stesso provider, quindi purché abbiano in comune l'autenticazione SPF (stesso indirizzo IP dell'outbound MTA condiviso da entrami i domini).
Infatti, anche in questo caso le due email che si generano con l'SMTP smuggling provenendo dal "loro" provider comune, avranno l'autenticazione SPF valida.
Generalizzando, ciò significa che un utente appartenente ad un qualsiasi dominio presente su un provider, con il rispetto di determinate condizioni, potrebbe inviare valide email con header From di un altro utente di altro dominio gestito dallo stesso provider. Tra le condizioni vi è la probabile condivisione degli stessi outbound MTA del provider, quindi del relativo record SPF.

Esempio:

```
[...]
DATA
From: Bob <bob@acme.it>\r\n
To: <alice@receiver.org>\r\n
Subject: Saluti\r\n
\r\n
Saluti dalla Sardegna,\r\n
Bob\n          riconoscimento del punto come
.\r\n          terminatore della prima email
MAIL FROM:<president@vertex.com>\r\n primo commando della seconda email
RCPT TO:<mark@receiver.org>\r\n
DATA\r\n                                  di colore verde è la sequenza
From: president@vertex.com\r\n             di comandi/email ricevuti ed
To: alice@receiver.org\r\n               accettati in modalità PIPELINING
Subject: Comunicazione\r\n
\r\n
Cara Alice,\r\n
ho deciso che … blah .. blah.\r\n
\r\n
ACME President\r\n
\r\n.\r\n
QUIT
```

In questo esempio i domini acme.it e vertex.com sono gestiti dallo stesso provider, tramite il proprio MTA e i relativi record nel DNS.
alice@receiver.org riceverà l'email di saluti dalla Sardegna, con mittente Bob mentre *mark@receiver.org* riceverà l'email con mittente il presidente di Vertex (From *president@vertex.com*), comunque inviata con la sessione MUA-MSA, autenticata per l'utente Bob (*bob@acme.it*), lo spammer!

Conclusioni.

Al link https://sec-consult.com/blog/detail/smtp-smuggling-spoofing-e-mails-worldwide/ della società SEC Consult, società che il 18 dicembre 2023 ha reso pubblica la vulnerabilità SMTP smuggling, si trovano maggiori dettagli e i risultati di alcuni test effettuati.
Microsoft e GMX.de hanno già provveduto ad effettuare le necessarie correzioni sui propri mailer, in quanto sono tra i provider utilizzati durante i mesi di analisi e verifica della vulnerabilità da parte della SEC Consult.
In ambiente IETF (Internet Engineering Task Force) c'è stata qualche polemica nei confronti della SEC Consult per non aver dato un sufficiente tempo di preavviso, prima di rendere pubblica la vulnerabilità, necessario per le correzioni dei vari mailer e per il particolare momento scelto, poco prima del periodo natalizio.

La società si è giustificata dichiarando che in data il 13 settembre 2023 ha segnalato la vulnerabilità al centro di coordinamento dei CSIRT USA, questo ha risposto il 29 novembre autorizzando la diffusione dell'informazione in quanto (secondo loro) non si tratta di vulnerabilità, ma di una “feature” del protocollo SMTP.
I test della SEC Consult sono stati effettuati solo su alcuni grandi provider, ma nel mondo ci sono milioni di piccoli provider i cui mailer (basati su Postfix, Sendmail, Exim, etc..) necessitano ora di un urgente aggiornamento.
La vulnerabilità non è nelle specifiche dei protocolli SMTP, Internet Message Format, estensione Pipelining, che sono estremamente chiare, anche in questi elementi, ma nelle loro implementazioni, che, come visto, vuoi per ragioni di interoperabilità verso mailer non conformi, vuoi per interpretazione non conformi allo standard, hanno portato a questa vulnerabilità del sistema email Internet.
Gli sviluppatori di Postfix, Wietse Venema e Viktor Dukhovni, che conta più di 1.400.000 installazioni nel mondo, hanno avuto "speciali" festività di Natale 2023, dato che il 26 dicembre, ad appena 8 giorni dell’annuncio della vulnerabilità, hanno rilasciato alcuni short-term workarounds che coprono tutte le versioni di Postfix e annunciato che le recenti (dopo il 18 dicembre) release di Postfix sono esenti dal problema.
Maggiori dettagli: https://www.postfix.org/smtp-smuggling.html
Infine, al link https://github.com/hannob/smtpsmug si trova lo script smtpsmug per testare se un sistema risulta vulnerabile (nota: script non testato).