Progetto Cyber Security & Ethical Hacking

Team: SecureSentinels

Modulo: Cybersecurity & Ethical Hacking (Epicode)

Data: 24 Febbraio 2026

Esercizio 6: Estrarre un Eseguibile da un PCAP

Obiettivi

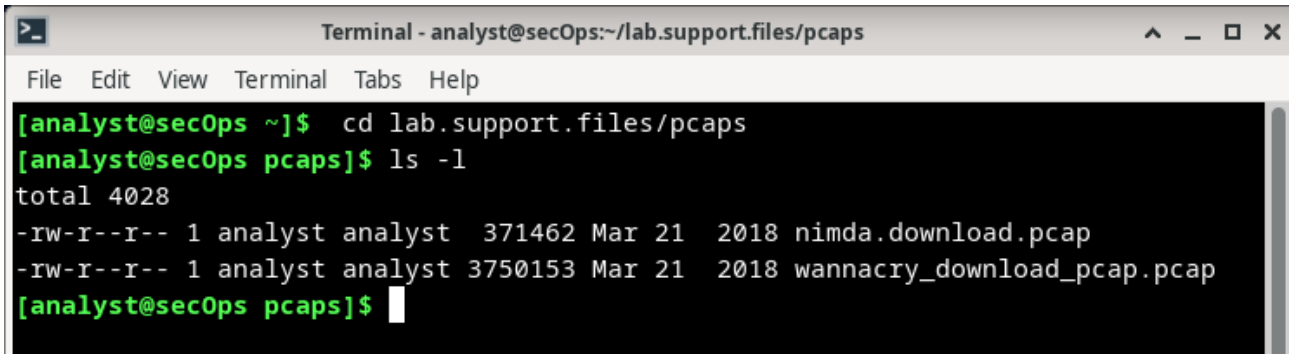
- **Parte 1: Analizzare Log e Catture di Traffico Pre-catturati**
- **Parte 2: Estrarre File Scaricati dal PCAP**

1. Attraverso l'analisi dei log si osserverà come avvengono le transazioni di rete a livello di pacchetto. In questo laboratorio, che vede in azione la Macchina Virtuale CyberOps Workstation, si effettuerà l'analisi del traffico in un file pcap attraverso lo strumento Wireshark (utilizzato per l'analisi e cattura del traffico dei dati di rete) e sarà estratto un eseguibile dal file.

Parte 1: Analizzare Log e Catture di Traffico Pre-catturati

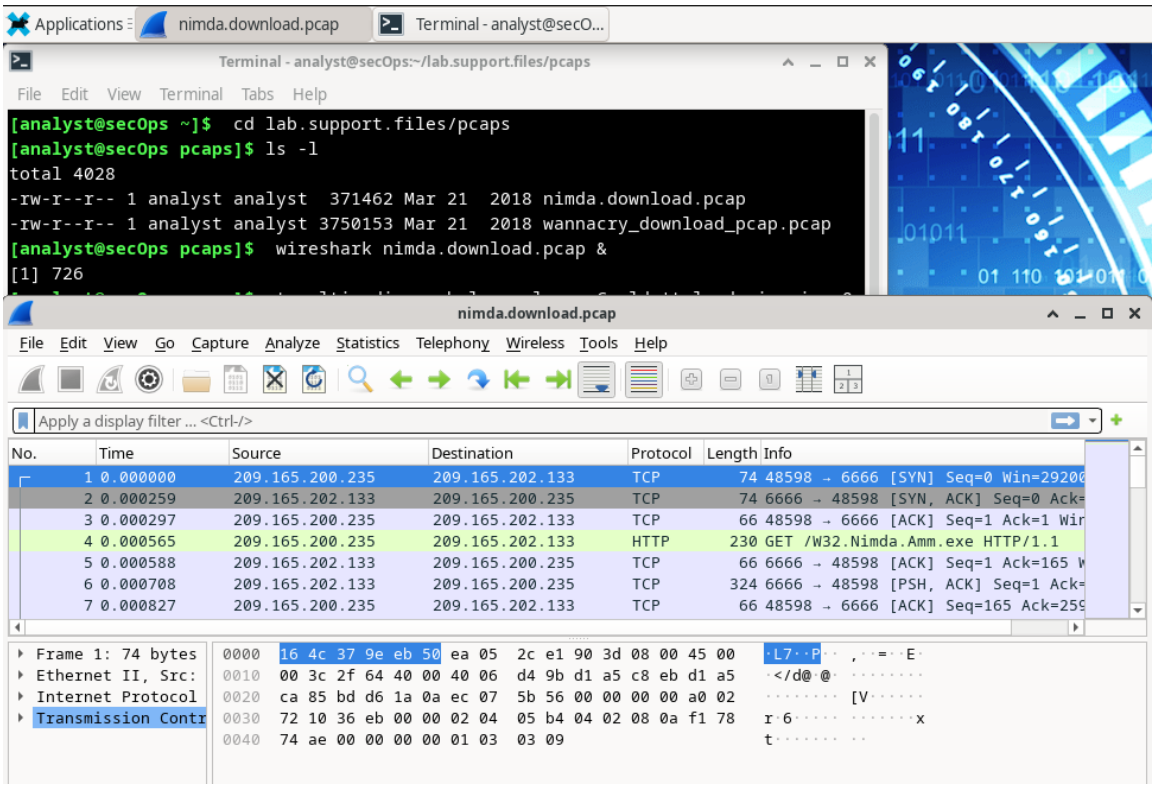
Istruzioni

a. Cambia directory nella cartella **support.files/pcaps**, e ottieni un elenco dei file usando il comando **ls -l**

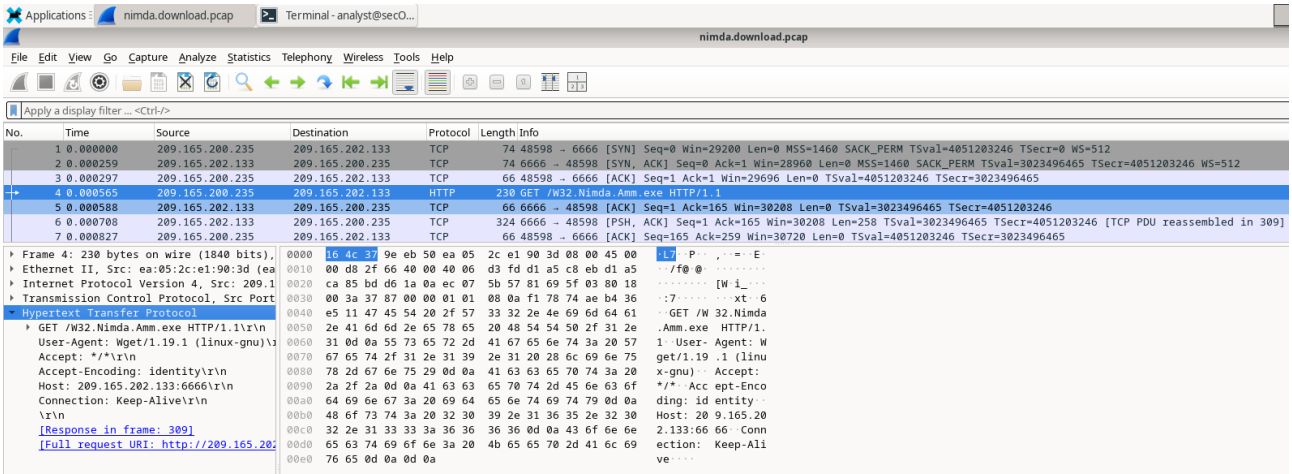


b. Esegui il comando sottostante per aprire il file **nimda.download.pcap** in Wireshark.

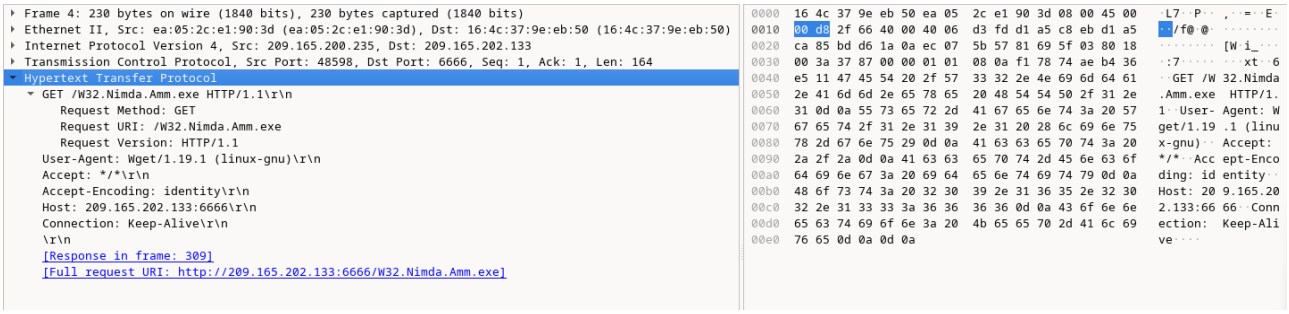
wireshark nimda.download.pcap &



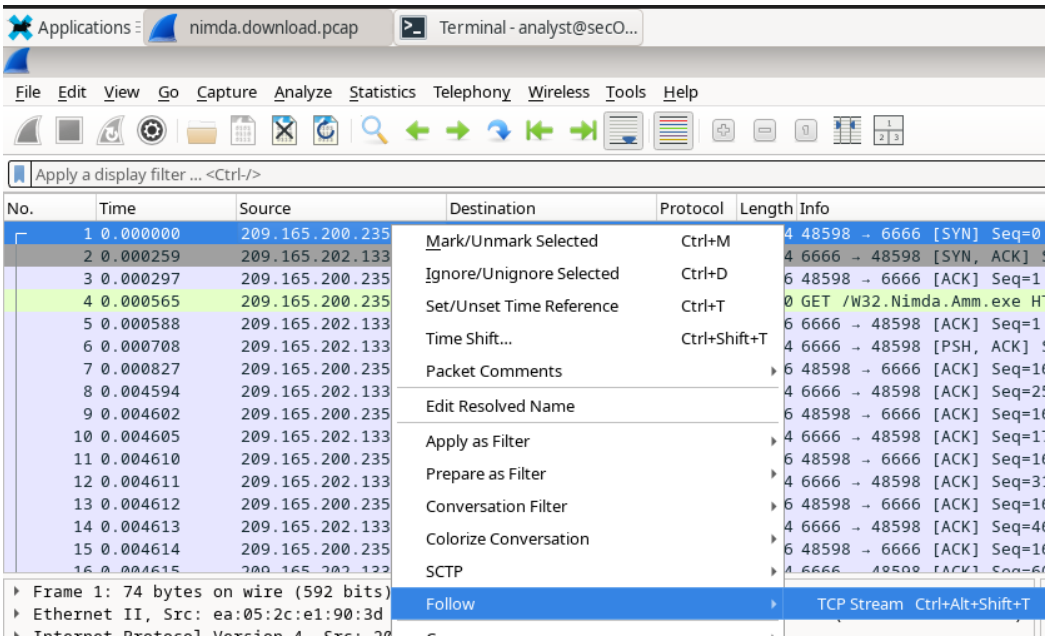
c. Il file **nimda.download.pcap** contiene la cattura dei pacchetti relativa al download del malware eseguito in un laboratorio precedente. Il **pcap** contiene tutti i pacchetti inviati e ricevuti mentre **tcpdump** era in esecuzione. Seleziona il quarto pacchetto nella cattura ed espandi **l'Hypertext Transfer Protocol**



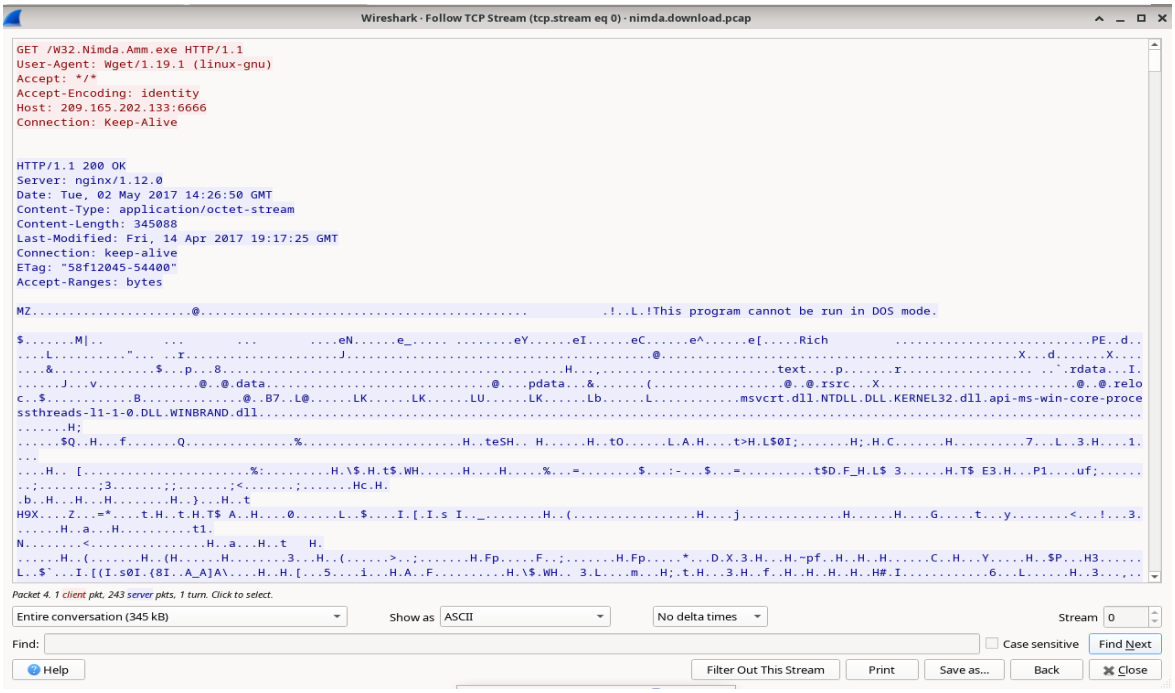
d. I pacchetti da uno a tre sono l'handshake TCP. Il quarto pacchetto mostra la richiesta per il file malware. Confermando ciò che era già noto, la richiesta è stata fatta tramite **HTTP**, inviata come richiesta **GET**



e. Poiché **HTTP** viene eseguito su **TCP**, è possibile utilizzare la funzione **Follow TCP Stream** di Wireshark per ricostruire la transazione TCP. Seleziona il primo pacchetto **TCP** nella cattura, un pacchetto **SYN**. Fai clic con il pulsante destro del mouse su di esso e scegli **Follow > TCP Stream**.



f. Wireshark visualizza un'altra finestra contenente i dettagli per l'intero flusso TCP selezionato.



Cosa sono tutti quei simboli mostrati nella finestra Follow TCP Stream? Sono rumore di connessione? Dati? Spiega. Ci sono alcune parole leggibili sparse tra i simboli. Perché sono lì?

- I simboli mostrati in figura riguardano il contenuto reale dei dati trasferiti sulla connessione TCP. Non sono rumore di connessione ma rappresentano i dati binari grezzi (il codice macchina) del file eseguibile scaricato, che Wireshark tenta forzatamente di tradurre in caratteri di testo ASCII. Le **parole leggibili** in mezzo ai simboli (come "This program cannot be run in DOS mode") sono stringhe di testo fisse inserite dai programmatori durante la compilazione del software, tipiche degli eseguibili di ambiente Windows.

Domanda Sfida:

Nonostante il nome W32.Nimda.Amm.exe, questo eseguibile non è il famoso worm. Per motivi di sicurezza, questo è un altro file eseguibile che è stato rinominato come W32.Nimda.Amm.exe. Usando i frammenti di parole visualizzati dalla finestra Follow TCP Stream di Wireshark, puoi dire quale eseguibile sia realmente

- Analizzando più in profondità i frammenti di testo all'interno del flusso TCP, è stato individuato un blocco XML (manifest) incorporato nel file. Questo passaggio ha permesso di confermare che il file intercettato **non è** il vero worm Nimda. Leggendo le stringhe interne (name="Microsoft.Windows.FileSystem.CMD" e <description>Windows Command Processor</description>), si evince chiaramente che, per motivi di sicurezza didattica, è stato utilizzato l'innocuo file di sistema Prompt dei Comandi di Windows (cmd.exe) opportunamente rinominato in W32.Nimda.Amm.exe.

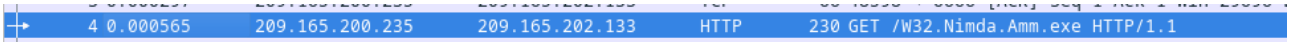
```
g="UTF-8" standalone="yes"?>
<!-- Copyright (c) Microsoft Corporation -->
<assembly xmlns="urn:schemas-microsoft-com:asm.v1" manifestVersion="1.0">
<assemblyIdentity
    version="5.1.0.0"
    processorArchitecture="amd64"
    name="Microsoft.Windows.FileSystem.CMD"
    type="win32"
/>
<description>Windows Command Processor</description>

<trustInfo xmlns="urn:schemas-microsoft-com:asm.v3">
    <security>
        <requestedPrivileges>
            <requestedExecutionLevel
                level="asInvoker"
                uiAccess="false"
            />
        </requestedPrivileges>
    </security>
</trustInfo>
<application  xmlns="urn:schemas-microsoft-com:asm.v3">
    <windowsSettings>
        <dpiAware  xmlns="http://schemas.microsoft.com/SMI/2005/WindowsSettings">true</dpiAware>
    </windowsSettings>
</application>
</assembly>
```

Parte 2: Estrazione e Verifica del File

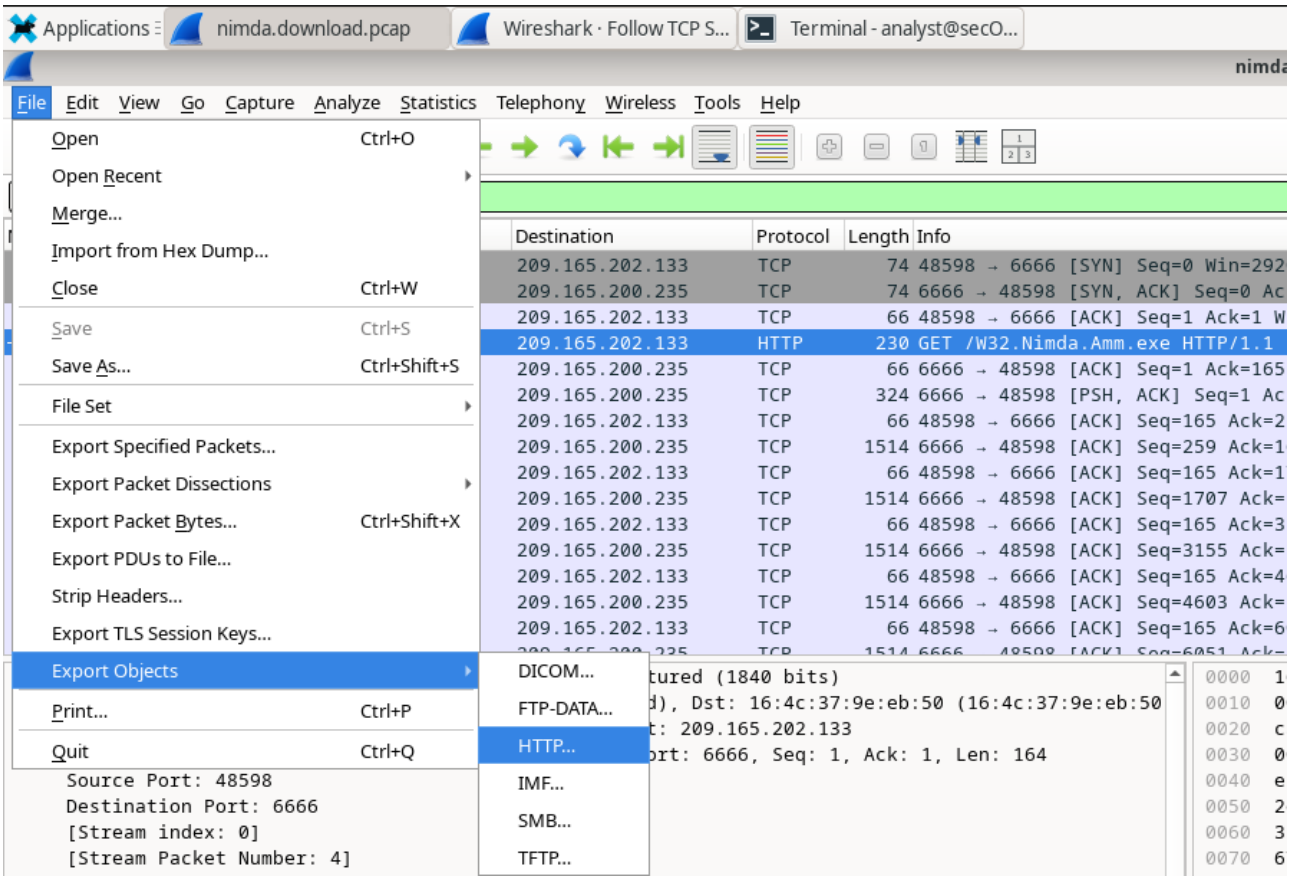
Istruzioni:

Poiché i file di cattura contengono tutti i pacchetti relativi al traffico, un **PCAP** di un download può essere utilizzato per recuperare un file scaricato in precedenza.

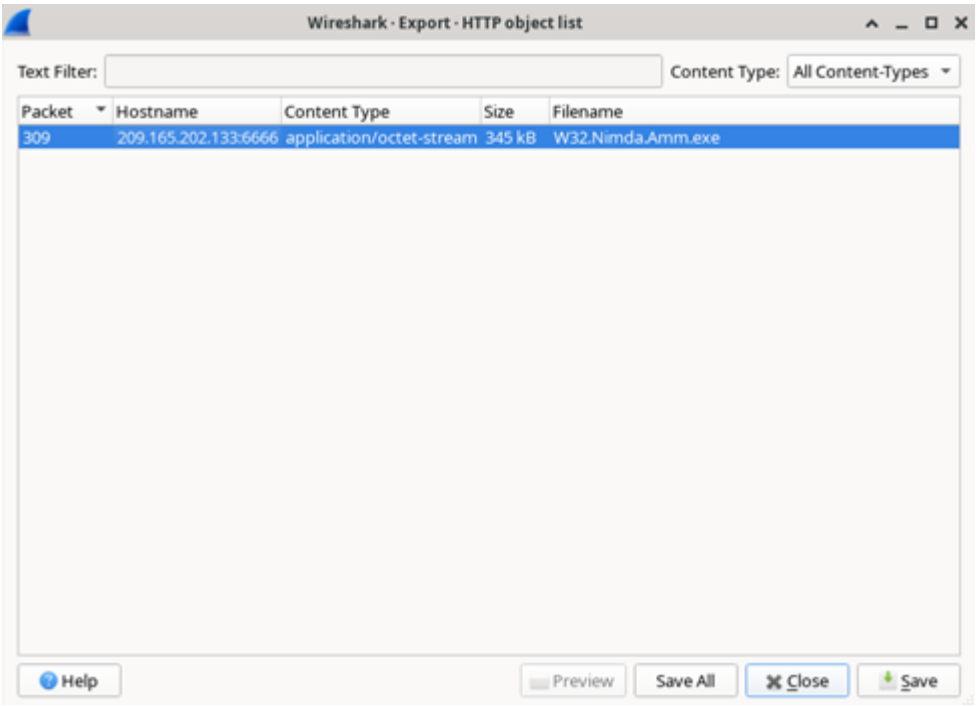
a. In quel quarto pacchetto nel file **nimda.download.pcap**, nota che la richiesta **HTTP GET** è stata generata da **209.165.200.235** a **209.165.202.133.**



b. Con il pacchetto della richiesta **GET** selezionato, naviga su **File > Export Objects > HTTP,** dal menu di **Wireshark.**



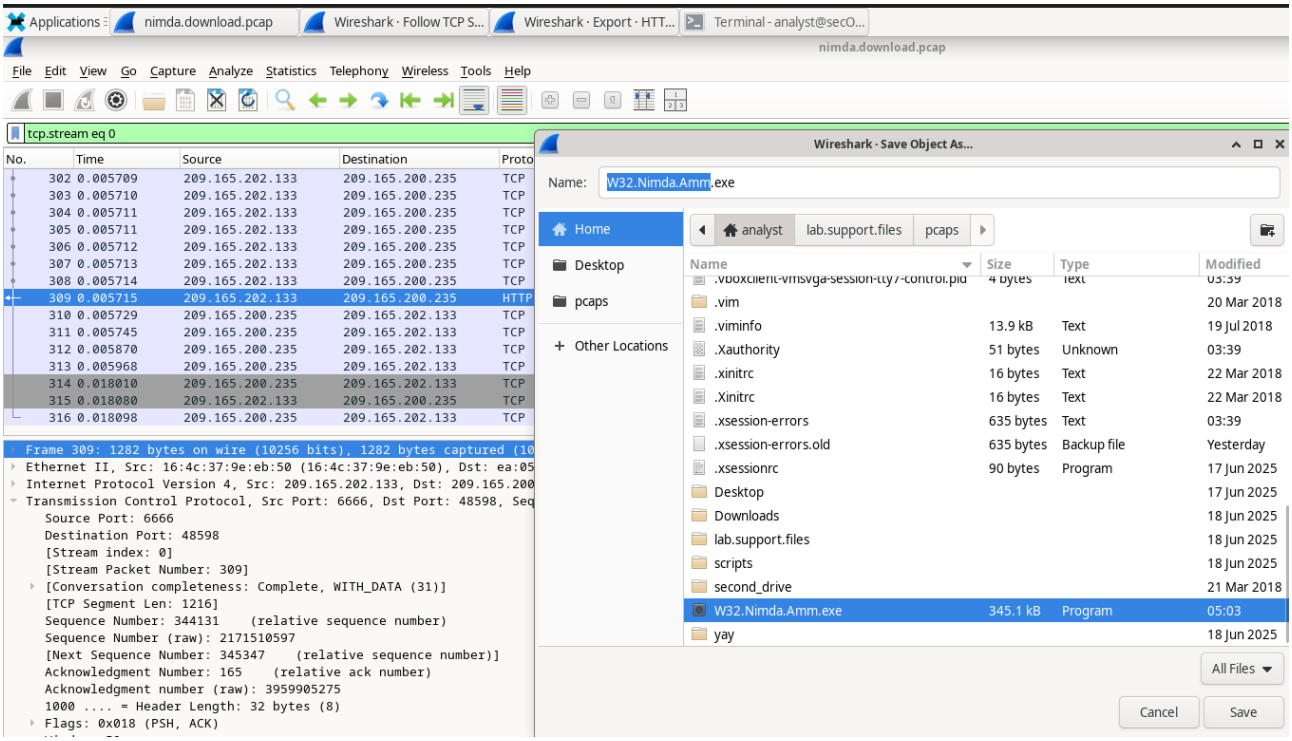
c. Wireshark visualizzerà tutti gli oggetti **HTTP** presenti nel flusso **TCP** che contiene la richiesta **GET.** In questo caso, solo il file **W32.Nimda.Amm.exe** è presente nella cattura.



Perché W32.Nimda.Amm.exe è l'unico file nella cattura?

- Dal menu principale di **Wireshark**, è stata utilizzata la funzione "**File > Export Objects > HTTP...**". Dalla lista degli oggetti intercettati è stato selezionato e salvato il file W32.Nimda.Amm.exe nella directory locale /home/analyst.
  I file **PCAP** contengono una copia esatta di tutto il traffico transitato. **Wireshark** è in grado di riconoscere i bit che compongono un file trasferito via **HTTP** e di riassemblarli in un file fisico sul disco, permettendone così l'estrazione sicura per un'analisi successiva. Nella finestra compare solo questo file poiché la cattura **PCAP** in esame è stata filtrata alla fonte per contenere unicamente questa specifica

d. Nella finestra **HTTP object list**, seleziona il file **W32.Nimda.Amm.exe** e fai clic su **Save As** nella parte inferiore dello schermo.
e. Fai clic sulla freccia sinistra finché non vedi **Home**. Fai clic su **Home** e poi sulla cartella **analyst** (non la scheda analyst). Salva il file lì



f. Cambia directory nella cartella **/home/analyst** ed elenca i file nella cartella usando **ls -l.**

```
[analyst@secOps ~]$ ls -l
total 6752
drwxr-xr-x 2 analyst analyst    4096 Jun 17  2025 Desktop
drwxr-xr-x 3 analyst analyst    4096 Jun 18  2025 Downloads
-rw-r--r-- 1 root    root    6538024 Feb 23 08:43 httpdump.pcap
drwxr-xr-x 9 analyst analyst    4096 Jun 18  2025 lab.support.files
drwxr-xr-x 3 analyst analyst    4096 Jun 18  2025 scripts
drwxr-xr-x 2 analyst analyst    4096 Mar 21  2018 second_drive
-rw-r--r-- 1 analyst analyst  345088 Feb 24 03:51 W32.Nimda.Amm.exe
drwxr-xr-x 5 analyst analyst    4096 Jun 18  2025 yay
```

g. Il comando file fornisce informazioni sul tipo di file. Usa il comando `**file**` per saperne di più sul malware

file W32.Nimda.Amm.exe

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

Nel processo di analisi del malware, quale sarebbe un probabile passo successivo per un analista di sicurezza?

- L'esercitazione ha dimostrato con successo come il traffico di rete non cifrato (HTTP) permetta la ricostruzione e l'estrazione totale dei file in transito.

  Nel processo reale di **Incident Response**, una volta estratto il file in questo modo, i probabili passi successivi per un analista di sicurezza sarebbero:

1. Calcolare l'hash crittografico del file (es. tramite il comando sha256sum) per ricercarlo all'interno di database di *Threat Intelligence* (come VirusTotal) e verificarne l'eventuale firma virale già nota.
2. Eseguire il malware in un ambiente *Sandbox* sicuro e isolato (Analisi Dinamica) per studiarne il comportamento, quali file tenta di modificare e quali connessioni di rete tenta di stabilire, senza mettere a rischio l'infrastruttura reale.