**Módulo 3 - Sesión Laboratorio**

|  |  |
| --- | --- |
| **Nombre:** Univ. Mamani Chavez Carla Vanesa | **CI:** 9124602 LP  **Paralelo:** Martes |
| **Docente:** Lic. Gallardo Portanda Franz Ramiro | **Fecha :** 28/06/2020 |

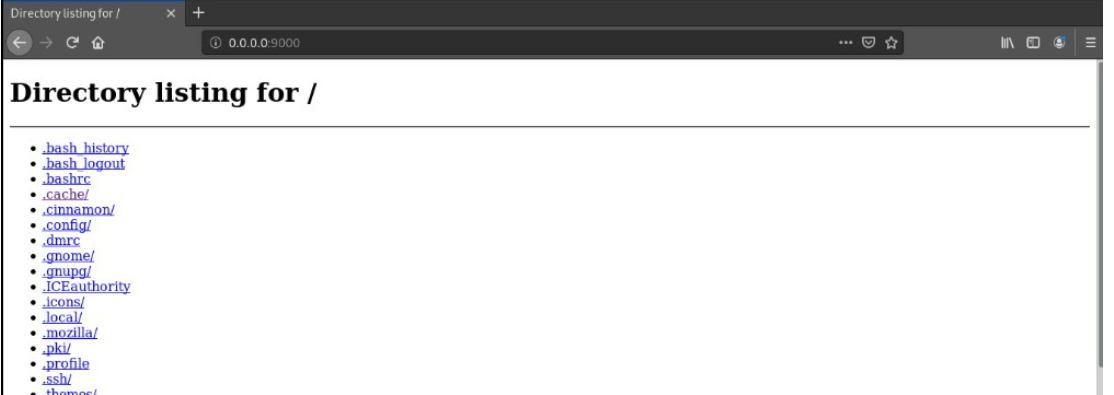
1. **Analisis de trafico TCP**
2. **En su máquina active un servidor Web, podemos usar Python para este propósito ejecutando el siguiente comando en la consola: python3 –m http.server 9000**

Una vez iniciado ejecutamos el comando

*python3 –m http.server 9000*

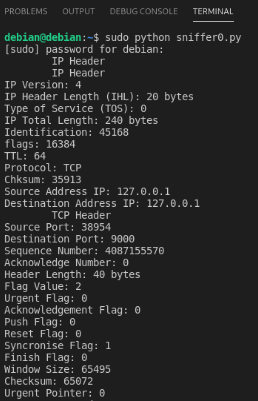


**El comando iniciara un servidor que escuche peticiones en el puerto 9000, a continuación, ejecutemos en el navegador y solicite un archivo al servidor. Con la página o el archivo desplegado en su navegador, habremos capturado suficiente tráfico para el análisis.**

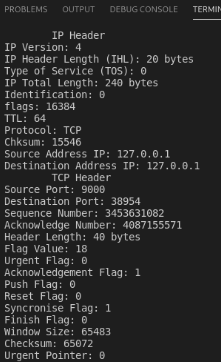


1. **Analice la información que arroja su programa e identifique los paquetes intercambiados para el establecimiento de la conexión TCP entre el servidor y el navegador. Recuerde que en TCP, el establecimiento de la conexión suele denominarse: proceso de acuerdo en tres, y que hace referencia al intercambio de tres paquetes específicos para establecer los parámetros de la conexión. Para estos tres paquetes proporcione toda la información de capa 3 y capa 4 que proporciona su programa.**

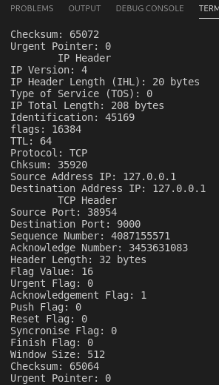
**1er camino, Flag Syn activado (Origen: PC – Destino: 9000)**



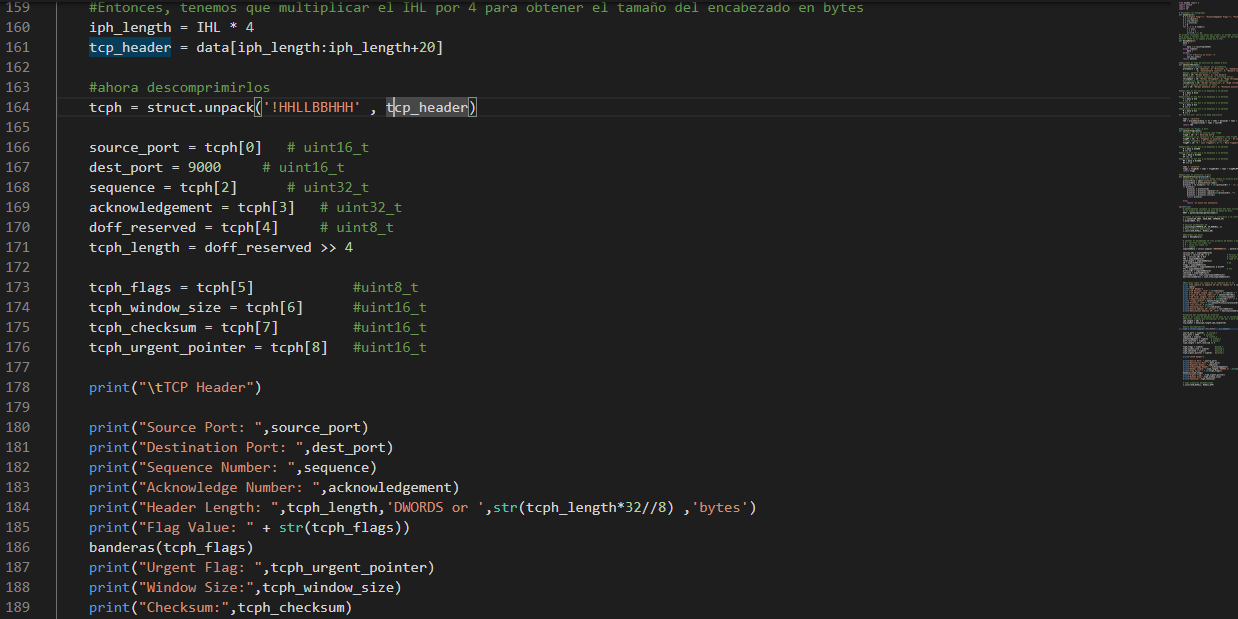
**2do camino, Flags Ack - Syn activado (Origen: 9000 – Destino: PC)**



**3er camino, Flag Ack activado (Origen: PC – Destino: 9000)**



**Donde:**



**Source Port:** Identifica el número de puerto de un programa de aplicación de origen.

**Destination Port:** Identifica el número de puerto de un programa de aplicación de destino.

**Sequence Number:** Especifica el número de secuencia del primer byte de datos de este segmento.

**Acknowledgment Number:** Identifica la posición del byte más alto recibido.

**Doff reserved:** Reservado para uso futuro.

**Windows Size:** Especifica la cantidad de datos que el destino está dispuesto a aceptar.

**Checksum:** Verifica la integridad de la cabecera y los datos de segmento.

**Urgent Pointer:** Indica datos que se deben entregar lo más rápidamente posible. Este puntero especifica la posición donde finalizan los datos urgentes.

|  |  |
| --- | --- |
| Code  Bits de control para identificar la finalidad del segmento: | |
| URG | El campo de puntero urgente es válido. |
| ACK | El campo de reconocimiento es válido. |
| PSH | El segmento solicita un PUSH. |
| RTS | Restablece la conexión. |
| SYN | Sincroniza los números de secuencia. |
| FIN | El remitente ha alcanzado el final de la corriente de bytes. |

1. **Conclusión**

Construimos un sniffer de la primera tarea que interpretaba los campos de la cabecera IP asi generando el trafico local o conectado a la red,

Ahora extendiendo las capacidades analizaremos el trafico TCP. Donde adquirimos e identificamos los paquetes establecidos por la conexión TCP todo esto con la ayuda de BaseSniffer que contuviese los mecanismos en común entre los tipos de captura para procesar los paquetes que se capturen, cada uno con diferentes características.Así mismo las banderas (Flags) son las encargadas de especificar los diferentes estados de la comunicación. También validan los valores de los distintos campos de la cabecera de control. Puede haber simultáneamente varios flags activados.