Tarea 7 Redes: Inyectando paquetes UDP en un socket UDP

Nombre: Patricio Espinoza A. **Sección**: 1

1. Este ejercicio muestra lo trivial que es la seguridad en UDP. Si uno quisiera tener un sistema seguro en UDP, ¿cómo podríamos protegernos de este tipo de falsificaciones?

Ya que UDP no cuenta con garantías de seguridad, las opciones son utilizar

- **Firewalls (o filtros por IP y PORT)** para filtrar los ataques masivos como los realizados en el experimento.
  Referencia
  [What Is a UDP Flood Attack?]

- **HMAC (Hash-based Message Authentication Code):** con una clave compartida entre emisor y receptor para verificar la autenticidad de cada paquete al integrar una función de hash y llaves secretas.

  Referencia

  [What is HMAC(Hash based Message Authentication Code)? - GeeksforGeeks]

- **DTLS (Datagram Transport Layer Security):** Permite a las aplicaciones cliente/servidor comunicarse evitando las escuchas no deseadas (eavesdropping), ataques de intermediario (MITM), o modificación de mensajes, lo que da tanto confidencialidad como autentificación mutua.

  Referencia

  "use the DTLS protocol within the context of the Windows authentication Security Support Provider Interface (SSPI) model to secure the communication between clients and servers." [Datagram Transport Layer Security protocol | Microsoft Learn]

- **Encriptar las secuencias (000-999):** A través de un valor seguro obtenido criptográficamente, obtener una secuencia aleatoria.

2. En TCP, ¿sería igual de trivial inyectar un paquete de datos?

No sería igual ya que TCP realiza un mantenimiento de estado y de control de sesión. Entre sus particularidades tiene que hay un "handshake" inicial para establecer la conexión que el atacante debería replicar correctamente para establecer una conexión falsa, también están los números de secuencia que son de 32 bits, añadiéndole una dificultad mayor que con respecto a UDP, finalmente hay reconocimiento de ACKs que deben ser válidos para que el receptor procese el paquete en conjunto con una ventana deslizante (si se implementa) que limita aquellos paquetes válidos en cierto tiempo.

3. Si ahora queremos que el pirata este en otro computador que el cliente, ¿se podría hacer lo mismo con scapy? ¿Cómo?

Si se podría llegar a hacer pero es mucho más difícil y dependiente de factores, en particular:

- Ruteo de paquetes
    - El router del receptor debe aceptar y enrutar aquellos paquetes provenientes del atacante con la IP de origen falsificada.
    - Existen ISPs que implementan filtros anti-spoofing (BCP 38), que bloquea aquellos paquetes con una IP de origen inválido. [Information for end-users - BCP38]
    - La red a la que este conectado el computador pirata debe permitir IP spoofing (suplantación de IPs), además de pasar por los posibles firewalls. [IP Spoofing: Cómo identificarlo y prevenirlo - Una guía completa - OpenSecurity]
- Sincronización
    - Al estar en otro computador, y posiblemente en otra red, habrá mayor latencia lo que afectaría la sincronización entre los paquetes enviados por el computador pirata y los que están siendo comunicados entre el cliente y el receptor.
        - En particular aquí entrará en juego que si existe un firewall, probablemente no se pueda sobrecargar con tantos paquetes piratas **Referencia** [What Is a UDP Flood Attack?]

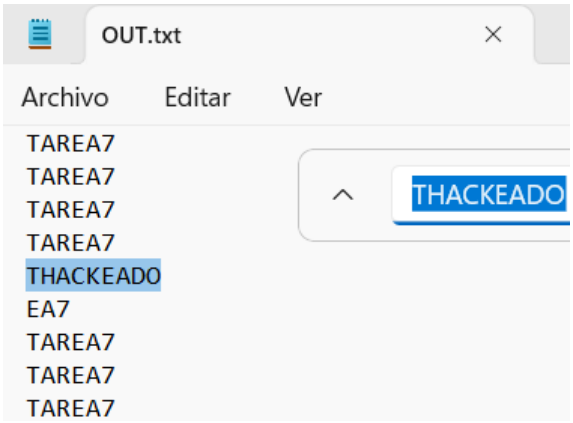
Para hacerlo hay 2 casos:

- Local
    - Requiere ver el tráfico en la red para conocer el puerto del cliente, en cuyo caso se podría necesitar usar ARP Spoofing.

      Referencia
      "Los ciberdelincuentes pueden enviar **mensajes falsificados ARP** a una LAN. Consigue vincular su dirección MAC con la dirección IP de un servidor, algo que como hemos visto anteriormente es necesario.
      A partir de ese momento empezaría a recibir cualquier tipo de información que ingrese a través de esa dirección IP y poder tomar el control del tráfico por completo." [Qué es y cómo evitar los ataques de ARP Spoofing]
- Internet
    - Para conocer el puerto del cliente remotamente se puede hacer por fuerza bruta, analizando si se obtiene una respuesta de "port unreachable"
    - Para evadir Firewalls se pueden utilizar túneles UDP, que además permite pivotear en una red interna.
      Referencia [Pivoting con Chisel - Túneles TCP/UDP sobre HTTP | Cristian Rebollo]
    - Para encontrar la secuencia nuevamente la opción sería por fuerza bruta, inundando la red con paquetes intentando coordinar el patrón utilizado.

4. Revise el archivo de salida una vez que logró inyectar al menos un paquete pirata. ¿qué pasa después del primer paquete inyectado? ¿Cómo se mezclan los datos correctos con los falsos?

Después del primer paquete inyectado es posible notar como se rompe el patrón en que se escribian los datos de los paquetes legítimos (se “corrompe” el archivo), en particular, los datos del pirata toman prioridad sobre los que tenia el cliente y se puede notar que quedan escritos de forma intercalada:



Es posible ver como para la secuencia en donde se escribiría “TAREA7” se escribió la “T” del paquete legítimo, y luego el paquete pirata se logró conectar y escribió “HACKEADO”. Una vez que terminó, se escribieron los datos que seguían llegando del servidor, dejando un “EA7” en la siguiente línea.

Posterior a ello el archivo pareciera seguir normal, lo que demuestra la importancia de como basta un paquete que se logre hackear para corromper el archivo.

5. Si invertimos los parámetros al programa pirata y enviamos paquetes falsos a anakena con origen el socket del cliente, ¿también funciona? ¿por qué?

Debería de ser posible enviar paquetes falsos con la premisa de enviarlos con origen un socket creado del cliente. Esto ya que UDP no autentica o realiza una simulación de handshake con el origen como tal. Por tanto, si los datos coinciden con un cliente en comunicación, deberían de pasar al servidor ya que su implementación solo hace echo.

Esto también se basa en la premisa de hacerse en el mismo computador local, simulando el pirata en la misma red, en caso contrario es mucho más probable que falle por las medidas de seguridad generales que hay implementadas.

Los principales obstáculos que podrían hacer que este ataque falle son un sistema de seguridad del router, firewalls en el servidor que bloqueen IPs ilegitimas/invalidas. La autenticación de protocolos y manejos de estados no son implementados con el código de la T3, por tanto, no se tomarían en cuenta como obstáculos.

En el caso de que se aceptará un paquete falso enviado a anakena el efecto sería una posible desincronización en el protocolo. Esto sucede ya que se hace un paquete falso atacando al servidor simulando que viene del cliente, si el servidor acepta el paquete, entonces hace echo al cliente reenviándolo. Sin embargo, como el paquete no fue enviado por el cliente legítimo, la secuencia podría no coincidir, en cuyo caso será descartado. En caso contrario, si acierta en la secuencia entonces el cliente podría llegar a escribir datos incorrectos o desincronizarse al intentar retransmitir el paquete ya que no era lo esperado, interfiriendo en el protocolo.