Implementación de Cifrado Híbrido (RSA-AES) y Análisis de Tráfico de Red con Wireshark

Josue Romero[1,*]

[1]**Instituto Tecnologico Superior Sudamericano, Cuenca, Ecuador**
[*]**Autores correspondiente: jaromero.3@sudamericano.edu.ec**

Resumen

En la presente práctica se desarrolla la implementación de un sistema de cifrado híbrido empleando el lenguaje de programación Python junto con la librería cryptography. Para ello, se diseñó una arquitectura cliente-servidor basada en sockets TCP que permite el intercambio seguro de información. El sistema hace uso del algoritmo asimétrico RSA para el intercambio de claves y del algoritmo simétrico AES-GCM para el cifrado de los mensajes. Adicionalmente, se llevó a cabo un análisis del tráfico de red mediante Wireshark, aplicando filtros por dirección IP, con el objetivo de comprobar que la información transmitida no se expone en texto plano dentro de la red local y que los datos sensibles permanecen protegidos.

RSA; AES-GCM; Wireshark; Cifrado Híbrido; Python; Sockets:

Abstract

This laboratory work presents the design and implementation of a hybrid encryption system using the Python programming language and the cryptography library. A client-server model was implemented through TCP sockets to enable secure data exchange. The system relies on the RSA asymmetric algorithm for secure key exchange and the AES-GCM symmetric algorithm for encrypting transmitted messages. Furthermore, network traffic analysis was performed using Wireshark by filtering packets according to IP addresses, ensuring that confidential information is not transmitted in plain text across the local network.

RSA; AES-GCM; Wireshark; Hybrid Encryption; Python; Sockets:

1. Introducción

La protección de la información durante su transmisión es un aspecto esencial en las redes actuales. El cifrado híbrido surge como una solución eficiente al combinar la rapidez del cifrado simétrico, como AES, con la seguridad en la distribución de claves que ofrece el cifrado asimétrico, como RSA. En esta práctica se recrea un escenario real de comunicación segura entre un cliente y un servidor, donde ambas partes intercambian credenciales criptográficas de manera controlada. El propósito principal es demostrar la correcta implementación del sistema mediante código y validar, a través del análisis de paquetes de red, que un posible atacante no puede acceder al contenido original sin disponer de la clave privada correspondiente.

2. Objetivos

Objetivo general:

- Diseñar y poner en funcionamiento un modelo de intercambio de información entre un cliente y un servidor que incorpore mecanismos criptográficos combinados, evaluando su nivel de protección mediante la observación y estudio del tráfico de red generado..

Objetivos específicos:

- Configurar una arquitectura cliente-servidor que permita la comunicación segura a través de sockets TCP..
- Aplicar técnicas de criptografía asimétrica para el intercambio seguro de claves de sesión.
- Utilizar algoritmos de cifrado simétrico autenticado para proteger la confidencialidad e integridad de los mensajes transmitidos.
- Analizar los paquetes de red capturados con herramientas de monitoreo para comprobar que la información intercambiada no sea comprensible para terceros no autorizados.

ç

3. Marco Teórico

El cifrado híbrido es una técnica criptográfica que combina dos enfoques complementarios. La criptografía asimétrica, representada por RSA, se utiliza exclusivamente para proteger la distribución de la clave simétrica, mientras que la criptografía simétrica, como AES, se encarga del cifrado del resto de la comunicación, optimizando el rendimiento del sistema.

- RSA (Rivest-Shamir-Adleman): Es un algoritmo de cifrado asimétrico que emplea un par de claves, una pública y una privada. Su seguridad se fundamenta en la complejidad computacional de factorizar números enteros de gran tamaño.
- AES-GCM (Advanced Encryption Standard - Galois/Counter Mode): Es un algoritmo de cifrado simétrico eficiente y seguro. El modo GCM no solo cifra la información, sino que también proporciona autenticación e integridad mediante una etiqueta de verificación (authentication tag).
- Wireshark: Es una herramienta de análisis de protocolos de red que permite capturar y examinar paquetes de datos, facilitando el estudio del comportamiento del tráfico y la verificación de mecanismos de seguridad.
- Python: Es un lenguaje de programación de alto nivel, reconocido por su claridad, flexibilidad y amplio ecosistema de librerías. Gracias a su facilidad de uso y a su carácter de código abierto, es ampliamente utilizado en proyectos de seguridad informática y redes.

4. Materiales y métodos

4.1. Entorno de laboratorio

- Equipo Servidor (IP: 192.168.20.223): Python 3.x, Librería cryptography.
- Equipo Cliente (IP: 192.168.20.63): Python 3.x, Librería cryptography.
- Herramienta de análisis: Wireshark.
- Red de conexión: Wi-Fi local.

4.2. Procedimiento

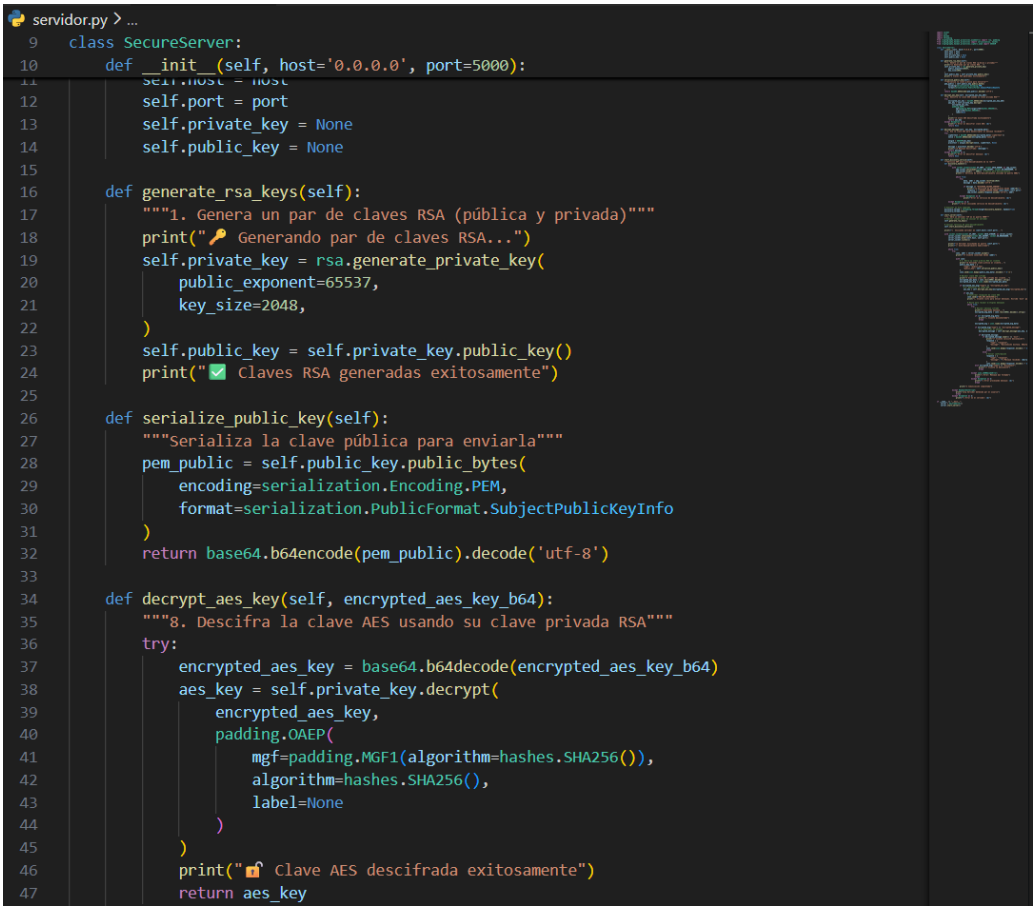


Figura 1. Ejecución del script del Servidor esperando conexión y generando claves RSA.

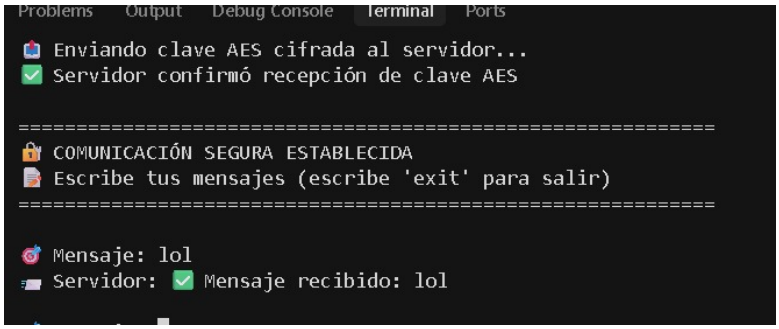


Figura 2. Ejecución del script del Cliente conectándose a la IP del servidor.

- Se configuró el script del servidor para escuchar en el puerto 5000 y generar un par de claves RSA de 2048 bits. Posteriormente, el cliente se conectó a la IP designada, recibió la clave pública, generó una clave AES aleatoria, la cifró y la envió de vuelta. Finalmente, se transmitió un mensaje secreto cifrado con AES-GCM.

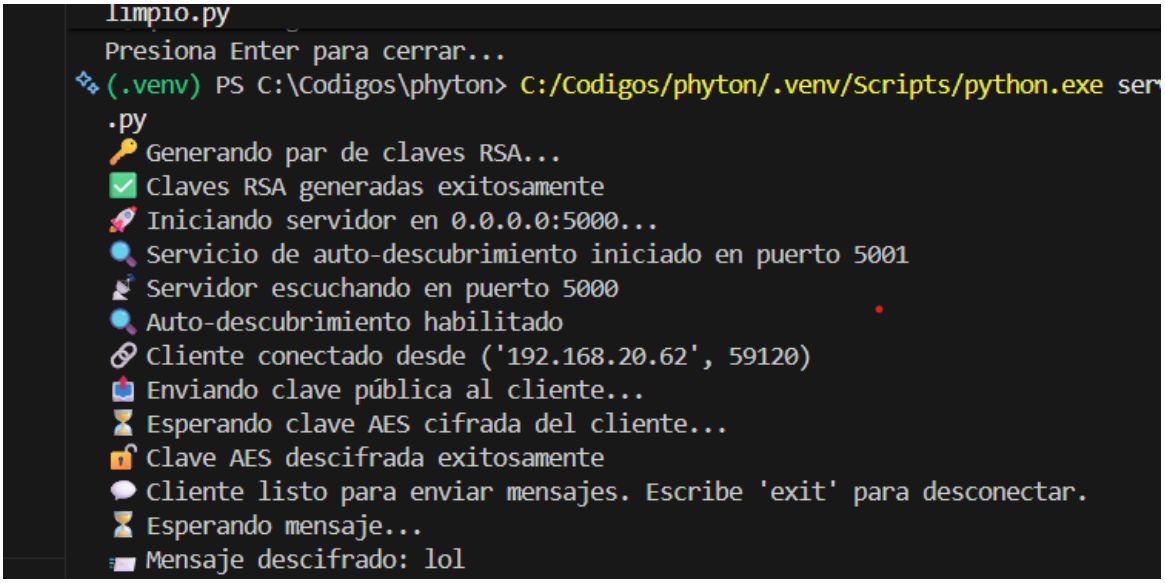


Figura 3. Mensaje descifrado exitosamente en la consola del servidor.

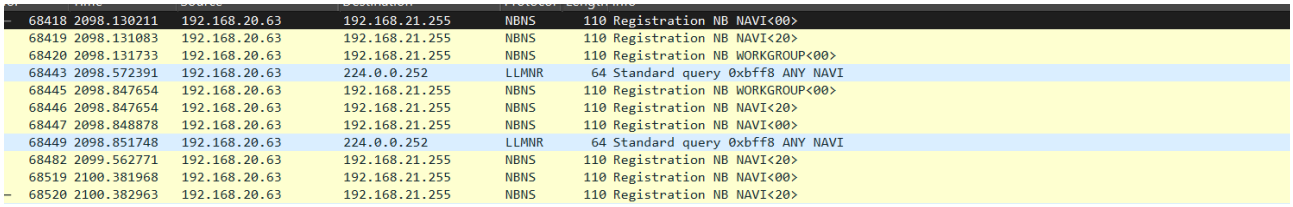
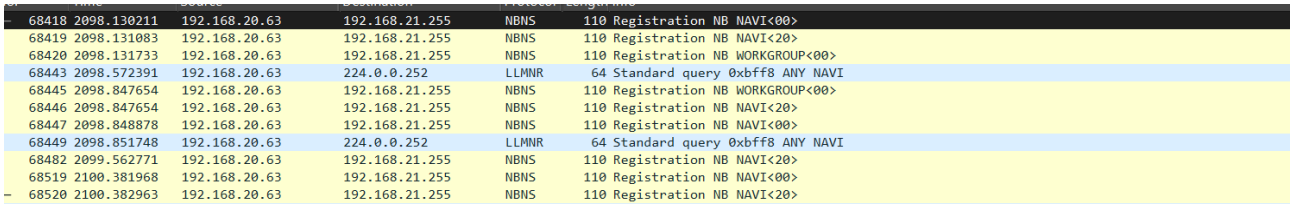


Figura 4. Trafico en wireshark

5. Análisis de resultados

Con el propósito de comprobar la robustez del sistema de seguridad implementado, se utilizó la herramienta de análisis de tráfico Wireshark en el equipo que cumple el rol de servidor. En lugar de aplicar un filtrado básico por número de puerto, se empleó un filtro específico por dirección IP, lo que permitió aislar de manera precisa la comunicación establecida entre los dos dispositivos participantes en la práctica.

Filtro aplicado: ip.addr == 192.168.20.63



68418	2098.130211	192.168.20.63	192.168.21.255	NBNS	110	Registration NB NAVI<00>
68419	2098.131083	192.168.20.63	192.168.21.255	NBNS	110	Registration NB NAVI<20>
68420	2098.131733	192.168.20.63	192.168.21.255	NBNS	110	Registration NB WORKGROUP<00>
68443	2098.572391	192.168.20.63	224.0.0.252	LLMNR	64	Standard query 0xbff8 ANY NAVI
68445	2098.847654	192.168.20.63	192.168.21.255	NBNS	110	Registration NB WORKGROUP<00>
68446	2098.847654	192.168.20.63	192.168.21.255	NBNS	110	Registration NB NAVI<20>
68447	2098.848878	192.168.20.63	192.168.21.255	NBNS	110	Registration NB NAVI<00>
68449	2098.851748	192.168.20.63	224.0.0.252	LLMNR	64	Standard query 0xbff8 ANY NAVI
68482	2099.562771	192.168.20.63	192.168.21.255	NBNS	110	Registration NB NAVI<20>
68519	2100.381968	192.168.20.63	192.168.21.255	NBNS	110	Registration NB NAVI<00>
68520	2100.382963	192.168.20.63	192.168.21.255	NBNS	110	Registration NB NAVI<20>

Figura 5. Captura de tráfico en Wireshark filtrada por la dirección IP del cliente.

Este método facilitó la visualización exclusiva de los paquetes intercambiados entre el cliente y el servidor, eliminando tráfico irrelevante de la red local.

Al examinar el flujo de comunicación mediante la opción Follow TCP Stream, se identificaron los siguientes comportamientos relevantes:

Distribución de la clave pública: El primer bloque de información enviado por el servidor se presenta en formato legible y contiene la cabecera ——BEGIN PUBLIC KEY——. Este comportamiento es esperado, dado que la clave pública está diseñada para ser transmitida sin restricciones de confidencialidad.

Protección de la clave de sesión AES: La respuesta generada por el cliente, con un tamaño aproximado de 256 bytes, aparece como una secuencia de datos no interpretables. Esto evidencia que la clave simétrica fue cifrada correctamente utilizando criptografía asimétrica, evitando su exposición durante la transmisión.

Cifrado del mensaje intercambiado: Los paquetes correspondientes al envío del mensaje final no revelan el contenido original. En su lugar, se observan valores aleatorios, lo que confirma la aplicación efectiva del cifrado simétrico mediante AES en modo GCM y la correcta protección de la información transmitida.

6. Conclusiones

El sistema de cifrado híbrido desarrollado cumplió satisfactoriamente su objetivo, permitiendo establecer una comunicación segura y confiable entre dos nodos de la red. La utilización de sockets en Python contribuyó a una comprensión clara del manejo de datos a nivel de transporte, facilitando la implementación del esquema criptográfico.

El análisis del tráfico realizado con Wireshark resultó clave para validar la seguridad del sistema, ya que al aplicar filtros por dirección IP se comprobó que tanto la clave de sesión como los mensajes intercambiados se transmiten de forma cifrada. Esto garantiza el cumplimiento del principio de confidencialidad y demuestra que un tercero sin las credenciales criptográficas adecuadas no podría interpretar la información capturada.

Referencias

Python Software Foundation. (2024). Cryptography Library Documentation. Recuperado de https://cryptography.io/en/latest/

Stallings, W. (2023). Cryptography and Network Security: Principles and Practice. Pearson.

Comer, D. E. (2022). Internetworking with TCP/IP. Addison-Wesley.

Wireshark Foundation. (2024). Wireshark User's Guide. Recuperado de https://www.wireshark.org/docs/wsug$_{h}tml_{c}hunked/$